COMBINING DEEP LEARNING WITH REASONING: FROM MAPPING SPECIES TO SOLVING GAMES AND CRYSTAL STRUCTURES

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Di Chen

December 2021

COMBINING DEEP LEARNING WITH REASONING: FROM MAPPING SPECIES TO SOLVING GAMES AND CRYSTAL STRUCTURES

Di Chen, Ph.D.

Cornell University 2021

Artificial Intelligence (AI) aims to develop intelligent systems, inspired in part by human intelligence. AI systems are now performing at human and even superhuman levels on a range of tasks such as image identification, face, and speech recognition. These major recent AI achievements have been driven largely by advances in supervised deep learning, which requires large labeled datasets to supervise model training. In contrast, humans often solve complex tasks using far fewer data by amplifying intuitive pattern recognition with meticulous reasoning that uses prior knowledge, a hybrid strategy that is challenging for machines to emulate.

In this thesis, we focus on integrating prior knowledge reasoning into deep learning via an interpretable latent space. When the prior knowledge is sufficiently rich, as is common in many scientific applications, we can supersede traditional example-based supervised learning and compensate for a dearth of labeled data by exploiting prior knowledge and magnifying it with logical and constraint reasoning seamlessly integrated into neural network optimization.

We first illustrate this idea in the context of supervised learning on multi-label classification — in particular, on joint species distribution modeling, where we propose the Deep Multivariate Probit Model (DMVP) to uncover species interactions and habitat associations via the interpretable latent space for the entire North American avifauna and accelerate the learning by an order of magnitude

using prior knowledge of the low-rank structure of species interaction. Next, we demonstrate the capability of this approach on unsupervised tasks with rich prior knowledge via a novel framework called Deep Reasoning Networks (DRNets). For variants of visual Sudoku games, DRNets outperforms supervised state-of-the-art methods in an unsupervised manner. In materials science, DRNets surpasses previous approaches and the capability of experts on crystal-structure phase mapping, unraveling the Bi-Cu-V oxide phase diagram and enabling the discovery of solar-fuels materials. In the future, we plan to further develop, adapt, and customize Deep Reasoning Networks to effectively solve a variety of tasks that require a combination of pattern recognition, reasoning, and learning, which are pervasive in science and other application domains.

BIOGRAPHICAL SKETCH

Di Chen worked with Professor Carla P. Gomes for his Ph.D. research in the Department of Computer Science at Cornell University. Di Chen's research is centered on solving structured prediction, multi-entity modeling, covariate shift, task-based learning, and especially combining prior knowledge reasoning with deep learning for scientific discovery. Di Chen's work is highly motivated by key problems across multiple scientific domains, including artificial intelligence, machine learning, ecology, materials science, and citizen science. His work on automating crystal-structure phase mapping was featured as the cover article in the journal *Nature Machine Intelligence*. Prior to coming to Cornell, Di Chen completed his bachelor's degree in science in the ACM Honored Class at Zhiyuan College, Shanghai Jiao Tong University, China in 2017.

Dedicated to my parents, my advisor Carla, and my darling Nijia.

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation for and sincere gratitude to Professor Carla P. Gomes for continuous support on my Ph.D. journey and for her invaluable guidance, endless passion, immense knowledge, and grand vision. I am deeply indebted to her for the guidance she provided to me during the entire time I worked on the research for and writing of this thesis. I would like to thank my committee members, Professor Bart Selman, Professor Thorsten Joachims, and Professor Volodymyr Kuleshov for their helpful comments as well as for broadening my research from various perspectives. I would also like to thank Professor John Hopcroft for his dedication to the improvement of education in my home country, which eventually led to my Ph.D. journey at Cornell.

I would like to extend my sincere thanks to my colleagues and collaborators, John M. Gregoire, Daniel Fink, Yexiang Xue, R. Bruce van Dover, Yiwei Bai, Junwen Bai, Wenting Zhao, Shufeng Kong, Sebastian Ament, Johan Bjrck, Brendan Rappazzo, Rich Bernstein, Dan Guevarra, Lan Zhou, Shuo Chen, Yada Zhu, Xiaodong Cui, Miao Liu, and Jianbo Li, for our stimulating discussions, for the work we have done together, and all the fun we have had in the last four years. I am especially grateful to former senior Ph.D. student Yexiang Xue (who is now a professor at Purdue University), for mentoring me and enlightening me as I began my research.

Last but not least, I would like to express my appreciation for the support, both physical and spiritual, that I have received from my parents, my wife Nijia, and my friends.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Recent AI achievements have largely been driven by advances in deep learning, which has achieved tremendous success in areas such as image identification, face and speech recognition, autonomous driving, and game playing. Besides those common applications, AI also has great potential to dramatically accelerate scientific discovery[1, 40, 5, 145, 124, 140]. However, scientific discovery often requires combining data analysis with reasoning about prior knowledge, which is still a challenge for AI. In the quest for AI, researchers have embraced human cognition as an important source of inspiration. Kahneman [72] describes human thought as the combination of two processes: *thinking fast* in System 1, which executes highly automated and largely effortless pattern recognition tasks, and *thinking slow* in System 2, which performs complex reasoning. Both System 1 and System 2 have been emulated in AI approaches. Deep learning comprises one of the most successful artificial analogues of System 1, through its fast processing and pattern recognition. Artificial analogues of System 2 have been established in the complementary AI sub-fields of combinatorial and constraint reasoning, in which search and inference are used to tackle complex problems. Simultaneously thinking fast and slow is a hallmark of human intelligence and is inherent to the large breadth of problems solved by humans. To tackle challenges such as outperforming humans in the game of Go, AI systems integrate the two styles of processes, as exemplified by AlphaGo [130], which combines deep reinforcement learning with Monte Carlo tree search. Nevertheless, such hybrid approaches are not always possible with existing algorithms and may result in

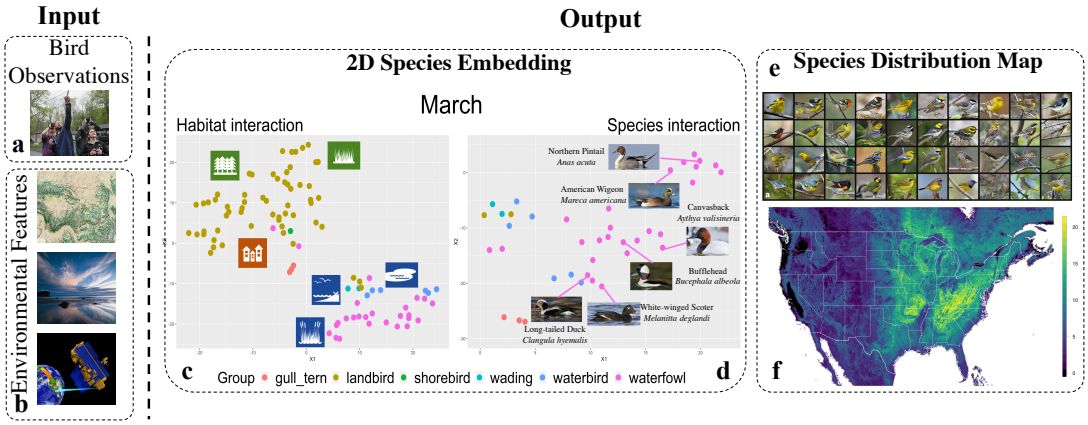


Figure 1.1: **Joint species distribution modeling.** In joint bird distribution modeling, given as input bird observations (a) and environmental features (b), the goal is to output a predictive JSDM (f), along with species environmental associations (c) and interactions (d). (See details in the following chapters.)

inferior performance due to coordination barriers between processing modules.

In this thesis, I introduce novel frameworks for integrating pattern recognition capabilities of deep learning with reasoning about prior knowledge based on an interpretable latent space. The idea of an interpretable latent space was first used in the context of supervised learning in multi-label classification, in particular in deep-learning-based joint species distribution models (JSDMs) [18, 19] that integrate prior ecological knowledge and species observational training data from the eBird citizen science program [139] (see Fig. 1.1). The need to capture and interpret interactions between species and their local environments as well as interactions among different species, which are core questions in ecology and conservation [142], was the initial motivation and inspiration for the semantically interpretable structured latent space. For this problem, we propose the Deep Multivariate Probit Model (DMVP) to uncover species interactions and habitat associations via the interpretable latent space for the entire North American avifauna and accelerate the learning by an order of magnitude using prior knowledge of the low-rank structure of species interaction. Later on, we

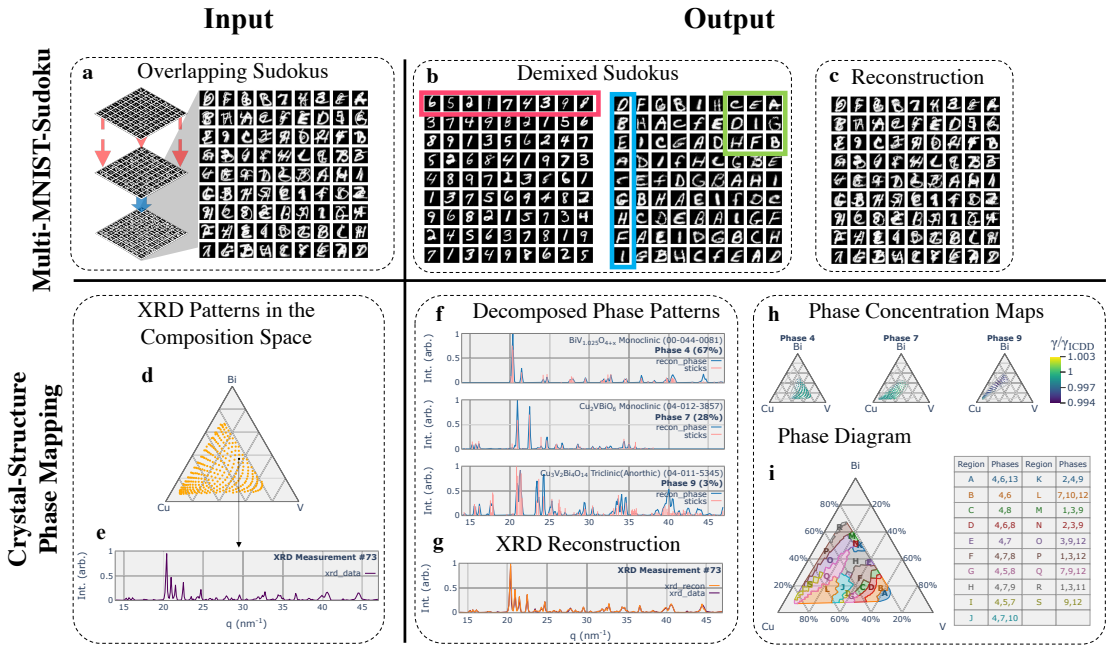


Figure 1.2: **Multi-MNIST-Sudoku (9×9) game and crystal-structure phase mapping.** In a 9×9 Sudoku game, the cells in each row (red rectangle), column (blue rectangle), and any of the nine non-overlapping 3x3 boxes (green square) have all-different digits. In Multi-MNIST-Sudoku, given images of mixed digit pairs, and prior knowledge that they form two overlapping Sudokus (**a**), the goal is to demix the digits into the two original Sudokus (**b**), by closely reconstructing the original input images (**c**). Phase mapping is a demixing task wherein a phase diagram **(i)** and the associated decomposed phase patterns (f–h) are inferred from a set of XRD patterns in a materials composition space **(d–e)**, requiring identification of pure-phase prototypes and their composition-dependent modification.

generalize this model to multi-target regression tasks for species abundance estimation [75], and the acceleration mechanism based on low-rank structure in DMVP is further applied to follow-up works based on the multivariate probit model for multi-label classification and multi-property prediction tasks [4, 76].

For unsupervised tasks with rich prior knowledge, I illustrate this idea via a novel approach called Deep Reasoning Networks (DRNets) [16, 15]. DRNets provides a general framework that integrates pattern recognition with prior knowledge reasoning for unsupervised demixing tasks. In Fig. 1.2, we

demonstrate the capabilities of DRNets on challenging visual Sudoku games (Multi-MNIST-Sudoku) and on high-throughput materials discovery problems (crystal-structure phase mapping). Both crystal-structure phase mapping and Multi-MNIST-Sudoku involve identification and demixing of the component signals in mixed-signal input data — specifically, crystal phases in the former, and handwritten digits and letters in the latter. Moreover, for scientific tasks such as crystal-structure phase mapping, researchers generally only have access to at most a few hundred (unlabeled) data samples, which greatly challenges classical data-hungry supervised deep learning models. Therefore, to tackle such unsupervised demixing tasks, supervision by constraint reasoning is required and supported by extensive prior knowledge from sources ranging from fundamental physical principles to the intuitive experience of scientists. Specifically, both demixing tasks (crystal-structure phase mapping and Multi-MNIST-Sudoku) involve two types of prior knowledge: prototypes of the component signals and rules that govern their mixtures. Both demixing tasks require constraint reasoning to interpret noisy and uncertain data while satisfying a set of rules: thermodynamic rules and Sudoku rules, respectively. When considering complex data instances with multiple composition degrees of freedom and many constituent phases, crystal structure phase mapping is substantially more complex than Multi-MNIST-Sudoku and can even surpass the analytical capabilities of human experts. As shown in the following chapters, by seamlessly integrating prior knowledge with deep learning via the interpretable latent space, DRNets outperforms supervised state-of-the-art methods in an unsupervised manner for variants of visual Sudoku games, and they surpass previous approaches and the capability of experts in crystal-structure phase mapping, unraveling the Bi-Cu-V

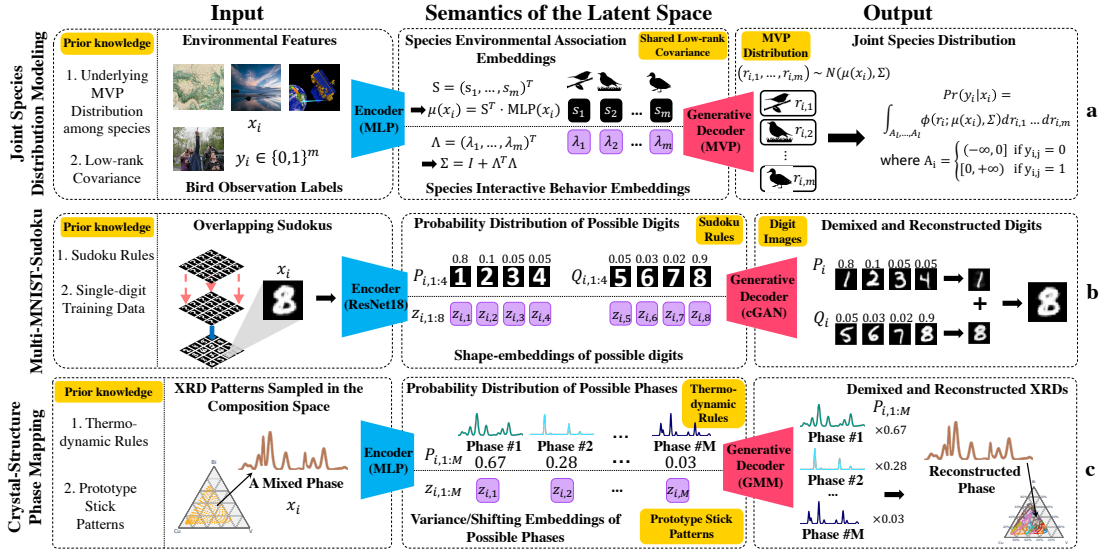


Figure 1.3: **Semantics of the interpretable latent space for different tasks. a.** In **joint species distribution modeling**, we extract the high-level environmental features $\text{MLP}(x_i)$ from the raw input x_i and encode the environmental association embeddings s_j and the interactive behavior embedding λ_j for each species j. The generative-decoder (MVP) uses s_j, $\text{MLP}(x_i)$, and λ_j to compute the corresponding $\mu(x_i)$ and Σ for the latent multivariate Gaussian random variables $r_{i,j}$ and maximizes the likelihood of the species observation y_i given x_i under the multivariate probit distribution. **b.** In **Multi-MNIST-Sudoku (4×4)**, we encode the input overlapping digits x_i into $P_{i,1:4}$, $Q_{i,1:4}$ and $z_{i,1:8}$, which denote the probability distribution and the shape embedding of possible digits (1–8). The generative-decoder (cGAN) uses $z_{i,1:8}$ to generate the demixed handwritten digits and reconstruct the original input with the expected overlapping image using $P_{i,1:4}$ and $Q_{i,1:4}$. **c.** In **crystal-structure phase mapping**, we encode the input XRD pattern x_i into $P_{i,1:M}$ and $z_{i,1:M}$, which denote the probability distribution and the variance/shifting-embedding of M possible phases. The generative-decoder (GMM) uses $z_{i,1:M}$ to generate the decomposed phases and reconstruct the original XRD using the phase probability distribution $P_{i,1:M}$.

oxide phase diagram and enabling the discovery of solar-fuels materials.

For all the aforementioned tasks, an interpretable structured latent space is key to incorporating prior knowledge, which is also the most challenging design of the model. The latent space is constructed using variables that have a specific interpretation and can be used in the formulation of the domain rules (see Fig. 1.3). For example, in the Sudoku domain, we will introduce a latent variable for each possible digit that gives the probability of that digit being

present in the cell associated with the input image. Consequently, we can use these variables in constraints on the allowed combinations of digits. Moreover, prior knowledge often involves complex constraints, such as combinatorial constraints to express valid Sudoku solutions or thermodynamic rules. For example, a digit cannot appear more than once in a row, and an X-ray diffraction pattern cannot be explained by more than 3 prototype phases, or by more than 2 prototype phases if there is alloying. To encode discrete variables that are involved in combinatorial constraints, we further propose a group of entropy-based continuous relaxations, where we model a probability distribution over all possible values for each discrete variable and gradually minimize the entropy of the distribution to mimic the original discrete variable.

The idea of combining deep learning with prior knowledge reasoning is highly motivated by real-world problems across multiple scientific domains. My research would not have been possible without my collaboration with the Cornell Lab of Ornithology and the Joint Center for Artificial Photosynthesis (JCAP) at Caltech. Through the collaboration with the Cornell Lab of Ornithology, we developed the largest-scale joint species distribution model, based on the *eBird* citizen science project [139], to make accurate prediction of the birds distribution on the entire North American continent and help design conservation plans. Our unsupervised model DRNets was motivated by the collaboration with the Joint Center for Artificial Photosynthesis (JCAP) at Caltech, to interpret X-ray diffraction data in the presence of physical constraints. Our work on automating crystal-structure phase mapping solved previously unsolved chemical systems, which led to the discovery of solar-fuels materials, and was featured as the cover article in the journal *Nature Machine Intelligence*.

In the future, the concept of combining deep learning with prior knowledge and the concept of the interpretable latent space will extend to other tasks. For example, in our on-going project, we incorporate the chemical rules of how molecular structure could possibly fragment into the graph neural network based deep learning framework to predict the experimental mass spectra from known chemical structures. This work has significantly outperformed existing rule-based methods and pure deep learning methods with respect to both the spectrum accuracy and the structure-level interpretability. More generally, research on incorporating neural network-based learning with symbolic knowledge representation and logical reasoning is an important next frontier in AI/ML research [26]. My thesis represents only one step in this direction but we expect that a variety of research projects will head in this direction in the future.

List of Publications

Some results in this thesis have been reported in the following peer-reviewed publications:

1. **Di Chen**, Yiwei Bai, Sebastian Ament, Wenting Zhao, Dan Guevarra, Lan Zhou, Bart Selman, R Bruce van Dover, John M Gregoire, and Carla P. Gomes. Automating crystal-structure phase mapping by combining deep learning with constraint reasoning. *Nature Machine Intelligence*, 3(9):812–822, 2021.
2. Yiwei Bai, **Di Chen**, and Carla P. Gomes. CLR-DRNets: Curriculum learning with restarts to solve visual combinatorial games. In *International Conference on Principles and Practice of Constraint Programming*. Springer, 2021.

3. **Di Chen**, Yada Zhu, Xiaodong Cui, and Carla P. Gomes. Task-based learning via task-oriented prediction network with applications in finance. *IJCAI*, 2020.

4. Shufeng Kong, Junwen Bai, Jae Hee Lee, **Di Chen**, Andrew Allyn, Michelle Stuart, Malin Pinsky, Katherine Mills, and Carla P. Gomes. Deep hurdle networks for zero-inflated multi-target regression: Application to multiple species abundance estimation. *IJCAI*, 2020.

5. **Di Chen**, Yiwei Bai, Wenting Zhao, Sebastian Ament, John Gregoire, and Carla P. Gomes. Deep reasoning networks for unsupervised pattern de-mixing with constraint reasoning. In *International Conference on Machine Learning*, pages 1500–1509. PMLR, 2020.

6. **Di Chen** and Carla P. Gomes. Bias reduction via end-to-end shift learning: Application to citizen science. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 493–500, 2019.

7. Johan Bjorck, Brendan H Rappazzo, **Di Chen**, Richard Bernstein, Peter H Wrege, and Carla P. Gomes. Automatic detection and compression for passive acoustic monitoring of the African forest elephant. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 476–484, 2019.

8. **Di Chen**, Yexiang Xue, and Carla Gomes. End-to-end learning for the deep multivariate probit model. In *International Conference on Machine Learning*, pages 932–941. PMLR, 2018.

9. Luming Tang, Yexiang Xue, **Di Chen**, and Carla P. Gomes. Multi-entity dependence learning with rich context via conditional variational auto-encoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

10. **Di Chen**, Yexiang Xue, Shuo Chen, Daniel Fink, and Carla P. Gomes. Deep multi-species embedding. In *IJCAI*, 2017.

CHAPTER 2

MULTI-LABEL CLASSIFICATION AND BEYOND

Understanding multi-entity interactions is a central question in many real-world applications. For example, in computational sustainability [42, 95], it is important to understand the spatial distribution of species and how species interact with each other and their environment, in order to develop conservation plans. In computer vision, detections of multiple objects are often correlated because of the shared background and scenario [152]. In natural language processing, a text often has several correlated labels in terms of its topic, emotion, and semantic meaning [108]. However, modeling the joint distribution of multiple entities is challenging, because the outcome space of the joint distribution is exponentially large w.r.t. the number of entities. As a result, most approaches can handle only a small number of entities, or each entity is modeled individually, ignoring the interactions among them.

In this chapter, we propose to model the joint distribution of multiple entities by combining the classical Multivariate Probit (MVP) model [3] with deep neural networks, which provides a scalable and flexible solution for modeling multi-entity distribution/interaction with a variety of contextual information. In the first part of this chapter, we illustrate the initial version of our method for joint species distribution modeling, where we jointly embeded multiple species as well as environmental covariates into a high-dimensional feature space to capture the inter-species interaction and the species-environment association and we learn the joint distribution via a Markov Chain Monte Carlo (MCMC) sampling method. Herein, the interpretable species embedding and the environmental covariates embedding comprise the prototype of the interpretable latent space

for our later works. In the second part of this chapter, we extend this method to a broad range of multi-label classification problems. Moreover, by incorporating prior knowledge of the low-rank covariance structure of the interaction among entities, we accelerate the learning by an order of magnitude via an efficient parallel sampling process. Furthermore, we provide theoretical and empirical analysis of the convergence behavior of our sampling process to illustrate its performance.

2.1 Joint Species Distribution Modeling

2.1.1 Introduction

Understanding the spatial distribution of species and how species interact with each other and their environment is essential for developing science-based conservation plans and ecological research. However, most species distribution models target only a single species at a time [113, 33, 36]. These single-species models ignore the role of species interactions, such as competition for shared resources (food, territory, etc.). For example, the American Robin and the Blue Jay are likely to be seen in the same place since the Blue Jay preys on Robin's eggs or fledglings and sometimes even steals its nest. Therefore, a model that predicts the occupancy of a collection of species instead of modeling each species individually is needed. The most straightforward formulation of a multi-species model [149] directly considers the probability of seeing a collection of species. However, this direct approach suffers from combinatorial intractability due to the large number of possible ways to form the collection. As a result, an efficient method of jointly modeling species distribution for a large number of species is

Figure 2.1: The ecological relationship between the American Robin and the Blue Jay.

still lacking.

We propose a novel method called Deep Multi-Species Embedding which can jointly model the distribution of hundreds of species as well as the correlations among species. DMSE jointly embeds multiple species as well as environmental covariates into a high-dimensional feature vector space via a deep neural network. Each embedded vector carries semantic meaning to the modeled entity, and the inner products of the embedded vectors for different species capture the relationships between entities (such as environmental preferences or correlations between species).

We apply this method to *eBird* bird observational data [105] and demonstrate how the DMSE model discovers inter-species relationships and outperforms the predictions of single-species distribution models (random forests and SVMs) as well as those of competing multi-label models. Moreover, as the number of species increases, the improvement in predictive performance of DMSE models over that of baseline models increases as well. Additionally, we demonstrate the benefit of using a deep neural network for feature extraction and show how the features improve the quality of species distribution modeling.

We also show a visualization of the embedding for hundreds of bird species in the northeastern US. That provides an intuitive picture of species' shared

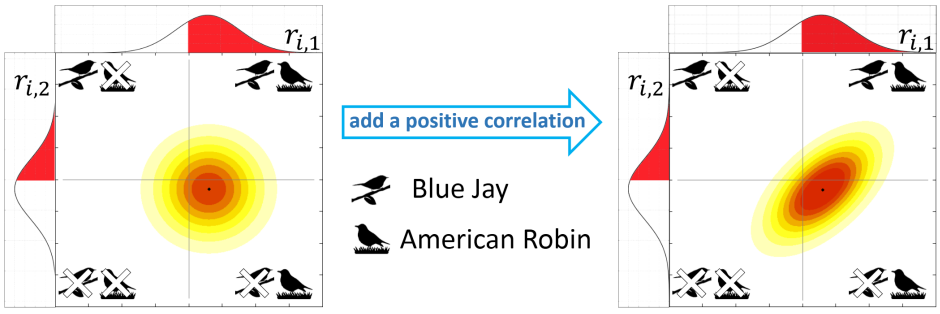


Figure 2.2: **Joint distribution of two species.** The left graph depicts the independent joint distribution of two species where the color from light yellow to red represents the probability from low to high. The probability mass in each quadrant represents the probability of each two-species co-occurrence. For example, the first quadrant represents the probability that two species occur together. The right graph is derived from the left one by adding a positive correlation between the two species. Here, we plot the marginal distributions for the two species on the upper and left sides of each graph. One can see, however, that if we change the correlation between the two species, the distribution of each species unconditional on the other remains the same.

environmental preferences and the correlations between species, after accounting for those explained by shared environmental preferences. Through this model, we are also able to quantitatively measure many species–species interaction which could be only qualitatively described by ecologists before.

2.1.2 Multi-Species Modeling

Our goal is to estimate the joint distribution of multiple species based on the observational data recording the presence or absence of each species at a site. More formally, given a collection of species $\{species_1, ..., species_n\}$ and the species observation data $D = \{(b_1, l_1), ..., (b_N, l_N)\}$, we would like to estimate the distribution $\Pr(b_i|l_i)$. Here $b_i \in \{0, 1\}^n$ is an indicator for the species co-occurrence of each observation, $b_{i,j} = 1$ if and only if $species_j$ was detected at site i, and $l_i = (f_1, ..., f_m)^T$ is an environmental feature vector that contains the values of m environmental covariates (or features) that describe site i. To simplify notation,

we also use l_i to denote the observation site i.

From One Species to More

Our DMSE method is based on the latent variable formulation of the probit model [21] which is widely used to model binary outcomes. The probit model also have been used by [114] on species distribution modeling. However, their setup is different from ours and can only handle a handful of species.

For the clarity of presentation, we start by describing how to model the distribution of single species using the probit model. For each $species_j$, we link the occurrence of $species_j$ at observation site l_i with a random variable $r_{i,j}$ where the probability that $species_j$ was detected at observation site l_i is equal to the probability that $r_{i,j} > 0$, i.e.

$$\Pr(b_{i,j} = 1|l_i) = \Pr(r_{i,j} > 0) \tag{2.1}$$

Here, $r_{i,j}$ follows a normal distribution $N(\mu_{i,j}, \sigma)$ where $\mu_{i,j}$ is a function of l_i and σ is fixed to be 1. According to the definition of normal distribution, a positive $\mu_{i,j}$ implies that the $species_j$ is more likely to be present than absent at site l_i and a negative $\mu_{i,j}$ implies the opposite. Therefore, we can model the distribution of each species by parameterizing $\mu_{i,j}$.

A general approach to model the joint distribution of multiple species is to simply join the distribution of each species assuming each species is independent. For the ease of presentation, we call this kind of joint distribution "independent joint distribution".

The left graph in the picture above (Fig.2.2) depicts the independent joint

distribution of two species (American Robin and Blue Jay) corresponding to random variable $r_{i,1}$ and $r_{i,2}$. In the graph, the color from light yellow to red represents the probability from low to high and the probability mass in each quadrant represents the probability of each co-occurrence. For example, the probability mass in the first quadrant shows the probability that American Robin and Blue Jay are present together at the observation site l_i. The one-dimensional distributions on the graphs' upper side and left side are the marginal distributions for American Robin and Blue Jay respectively. The red area in each one-dimensional distribution represents the probability of the presence of each species unconditional of the other.

Since the independent joint distribution can not model the correlation between species which widely exists in the real world, we extend the probit model by applying multivariate normal distribution over the n-dimensional random variables $r_i = (r_{i,1}, ..., r_{i,n})$ i.e.

$$r_i \sim N_n(\mu_i, \Sigma) \tag{2.2}$$

where $\mu_i = (\mu_{i,1}, ..., \mu_{i,n})^T$ and Σ is the covariance matrix. In this way, each random variable $r_{i,j}$ still follows a normal distribution, but we can capture interspecies correlation by parameterizing the covariance matrix Σ.

As shown in the right graph of Fig 2.2, we change the covariance between random variable $r_{i,1}$ and $r_{i,2}$ from 0 to a positive number ρ, then the joint distribution changes significantly. For example, the probability mass in the first quadrant becomes larger, which means these two species are more likely to be present together. Although we affect the joint distribution of two species by changing the covariance between $r_{i,1}$ and $r_{i,2}$, the marginal distribution of each random variable does not change. This means the probability of the presence of each

species, unconditional of the other, is unaffected by the covariance. This property ensures that our model can maintain the predictive capability derived from learning habitat preferences of each species. Meanwhile, it can outperform the independent version when the species distributions are correlated. In addition, if we restrict the variance of each species to be 1, the matrix Σ becomes a correlation matrix, a convenient and intuitive parameterization.

Deep Multi-Species Embedding

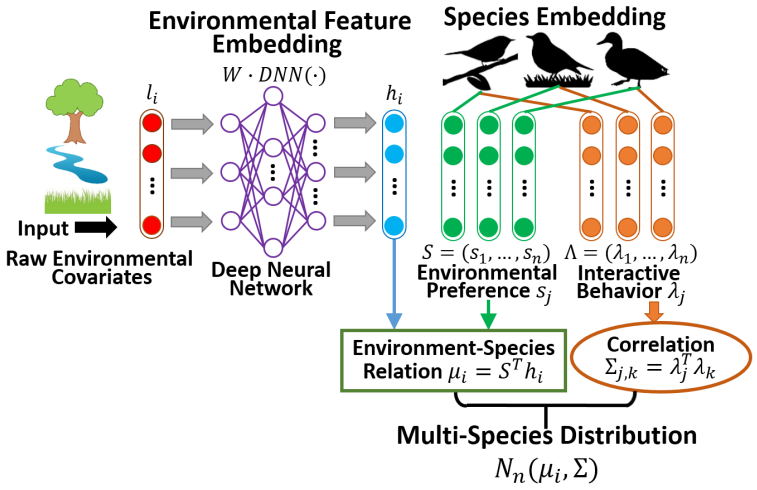


Figure 2.3: The intuitive visualization of DMSE framework.

In order to estimate the parameters μ and Σ, we need to first model the species-environment relationship as well as the correlation between species. To achieve this, we first embed each $species_j$ with two vectors $s_j \in R^{d_1}$ and $\lambda_j \in R^{d_2}$ representing its environmental preference and interactive behavior respectively. Here d_1, d_2 are the dimensionality of these two vector spaces which are manually set. In our experiments, we set $d_1 = d_2 = 100$. Because embedding methods are able to take advantage of high-dimensional representations, it is advantageous to set this parameters to be high, though the methods are not sensitive to the

exact values. We choose to model these two characteristics separately instead of embedding to the same vector, because it is not uncommon for groups of species to share similar environment-distribution relationships, but have very different inter-species associations. Thus, by modeling these characteristics separately, DMSE can capitalize on the shared environment-distribution relationships without biasing the inter-species correlation estimates. Moreover, because the environmental features used in the model describe habitat characteristics at a much coarser spatial resolution than that of the inter-species interactions, this model formulation can be seen as multi-scale approach that shares information at coarse scales at the habitat level while simultaneously allowing fine-scale variation between species.

When it comes to the environmental features, we apply a deep neural network and a projection matrix to embed the low-dimensional raw environmental data into the same d_1-dimensional feature space as the vectors s_j. For each observation (b_i, l_i),

$$l_i \xrightarrow{embed} h_i : h_i = W \cdot DNN(l_i), \tag{2.3}$$

here $DNN(\cdot)$, a function mapping from R^m to $R^{n_{output}}$, represents a deep neural network. In our experiment, we empirically found that a 3-hidden-layer fully connected neural network using tanh as the activation function worked the best. The number of neurons in each hidden layer was 256, 256, 64. W, a d_1-by-n_{output} projection matrix, is used for modulating the data range and mapping the DNN's output layer to the same high-dimensional feature space with s_j, and n_{output} is the dimension of output layer. We will include more discussion about the performance of the neural network in the experimental section.

With these embeddings for each species and the environmental features at each observation site, we use the inner-product ${s_j}^T h_i$ to score the relationship between $species_j$ and environmental features in the observation site l_i. Similarly, we also use the inner-product $\lambda_j^T \lambda_k$ to score the interaction between $species_j$ and $species_k$.

In order to simplify the presentation, we concatenate the vectors s_j and λ_j as the columns into two matrices.

$$\begin{aligned} S &= (s_1, s_2, ..., s_n) \in R^{d_1 \times n}, \\ \Lambda &= (\lambda_1, \lambda_2, ..., \lambda_n) \in R^{d_2 \times n} \end{aligned} \tag{2.4}$$

Using the notations in equation(2.2), (2.3) and (2.4) , we can formulate our DMSE model as follows,

$$\begin{aligned} &\Pr(b_{i,j} = 1 | l_i) = \Pr(r_{i,j} > 0), \\ &r_i \sim N_n(\mu_i, \Sigma), \\ &\text{where } \mu_i = S^T h_i = S^T (W \cdot DNN(l_i)) \text{ and } \Sigma = \Lambda^T \Lambda. \end{aligned} \tag{2.5}$$

Here $\mu_{i,j} = {s_j}^T h_i$ scores the habitat suitability of $species_j$ at observation site l_i and $\Sigma_{j,k} = \lambda_j^T \lambda_k$ represents the correlation between $species_j$ and $species_k$. According to the definition of multivariate normal distribution, we derive that

$$\begin{aligned} &\Pr(b_i | l_i) = \int_{L_1}^{R_1} ... \int_{L_n}^{R_n} f(x) dx_1 ... dx_n \\ &\text{where } f(x) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)), \\ &\text{and } L_j = \begin{cases} 0 & \text{if } \quad b_{i,j} = 1 \\ -\infty & \text{if } \quad b_{i,j} = 0 \end{cases}, R_j = \begin{cases} +\infty & \text{if } \quad b_{i,j} = 1 \\ 0 & \text{if } \quad b_{i,j} = 0 \end{cases} \end{aligned} \tag{2.6}$$

Training and Testing

We train our model by maximizing the log-likelihood on the observational data. The parameters that should be trained are the matrix S, Λ, W and the parameters in the deep neural network denoted by θ_{DNN}:

$$S, \Lambda, W, \theta_{DNN} = \text{argmax}_{S,\Lambda,W,\theta_{DNN}} \sum_{i=1}^{N} \log \Pr(b_i|l_i) \tag{2.7}$$

We use the stochastic gradient descent algorithm as proposed in [30, 13] to optimize the log-likelihood function in equation (2.7).

In order to train and test our DMSE model, we need to be able to compute the integration in equation (2.6) and its derivatives with respect to each parameter. For the integration, we use an adaptive algorithm proposed in [39], which can calculate the cumulative distribution function (CDF) on multivariate normal distribution with a high accuracy (relative error $< 10^{-6}$).

To compute the derivative of $\Pr(b_i|l_i)$, one key observation is that if we can compute the derivative of $\Pr(b_i|l_i)$ with respect to μ and Σ, we can easily obtain other derivatives we want by simply applying the chain rule. Since the multivariate normal distribution is uniformly continuous, we first transform the derivative of the integration into the integration of the derivative of density function as follows.

$$\begin{aligned}
\frac{\partial \log \Pr(b_i|l_i)}{\partial \mu} &= \frac{1}{\Pr(b_i|l_i)} \int_{L_1}^{R_1} \dots \int_{L_n}^{R_n} \frac{\partial f(x)}{\partial \mu} dx_1 \dots dx_n \\
\frac{\partial \log \Pr(b_i|l_i)}{\partial \Sigma} &= \frac{1}{\Pr(b_i|l_i)} \int_{L_1}^{R_1} \dots \int_{L_n}^{R_n} \frac{\partial f(x)}{\partial \Sigma} dx_1 \dots dx_n
\end{aligned} \tag{2.8}$$

Using the definition of multivariate normal distribution, we derive the following

equations:

$$\frac{\partial f(x)}{\partial \mu} = f(x) \cdot F(\Sigma, \mu, x), \frac{\partial f(x)}{\partial \Sigma} = f(x) \cdot G(\Sigma, \mu, x)$$

$$\text{where } F(\Sigma, \mu, x) = \Sigma^{-1}(x - \mu),$$

$$G(\Sigma, \mu, x) = -\frac{1}{2}(\Sigma^{-1} - \Sigma^{-1}(x - \mu)(x - \mu)^T \Sigma^{-1}) \tag{2.9}$$

According to equation (2.6), we know that

$$\int_{L_1}^{R_1} ... \int_{L_n}^{R_n} \frac{f(x)}{\Pr(b_i|l_i)} dx_1 ... dx_n = 1 \tag{2.10}$$

Thus, we can consider $\frac{f(x)}{\Pr(b_i|l_i)}$ as the density function of a distribution over a hypercube $Q \subseteq R^n$ corresponding to the integration range of equation(2.10). Thus, we can employ the Markov Chain Monte Carlo sampling method to estimate the derivative of $\log \Pr(b_i|l_i)$ with respect to μ and Σ as follows:

$$\begin{aligned} \frac{\partial \log \Pr(b_i|l_i)}{\partial \mu} &= \int_{L_1}^{R_1} ... \int_{L_n}^{R_n} \frac{f(x)}{\Pr(b_i|l_i)} F(\Sigma, \mu, x) dx_1 ... dx_n \\ &= E\Big[F(\Sigma, \mu, x)\Big]_{x \in Q} \approx \frac{1}{M} \sum_{k=1}^{M} F(\Sigma, \mu, x_k) \end{aligned} \tag{2.11}$$

$$\begin{aligned} \frac{\partial \log \Pr(b_i|l_i)}{\partial \Sigma} &= \int_{L_1}^{R_1} ... \int_{L_n}^{R_n} \frac{f(x)}{\Pr(b_i|l_i)} G(\Sigma, \Sigma, x) dx_1 ... dx_n \\ &= E\Big[G(\Sigma, \mu, x)\Big]_{x \in Q} \approx \frac{1}{M} \sum_{k=1}^{M} G(\Sigma, \mu, x_k) \end{aligned} \tag{2.12}$$

To make our model more efficient, we apply an enhancement for our model. Using the property of normal distributions, we know that $\Pr(|r_{i,j} - \mu_j| > k\Sigma_{j,j}) < \frac{e^{-k^2/2}}{k\sqrt{2\pi}}$. As the result, we can make a cut-off on L_i and R_i which significantly reduces our sample range and increases the convergence rate in our sampling process.

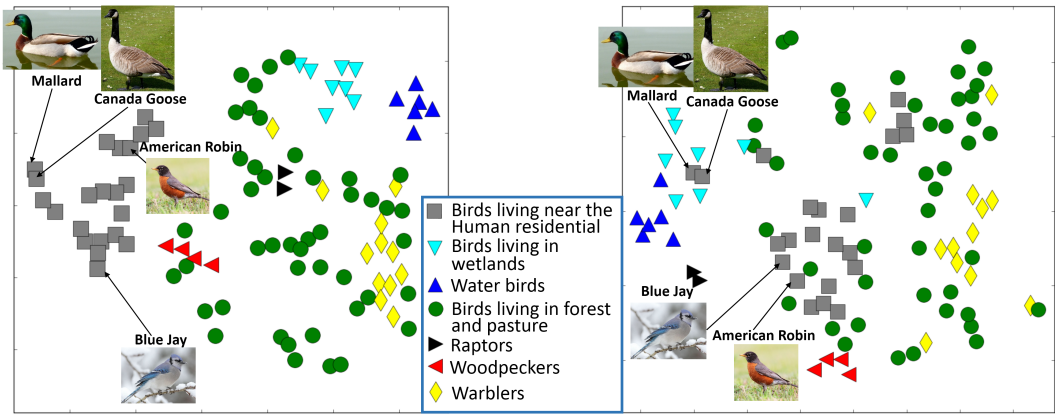


Figure 2.4: The left map visualizes the embeddings s_j representing the environmental preference of each species and the right graph depicts the embeddings λ_j corresponding to the correlation among species. One can see (the left map), birds of the same category cluster tightly and birds of the same breed also have a similar environmental preference. Compared with the right graph, one can find that the birds living in similar habitat have relative high correlation, but there are still some birds with high correlation that have a different environmental preference.

2.1.3 Related Works

We refer the reader to [33] for a survey of general techniques used in species distribution modeling. Modeling approaches in this area vary depending on the type of observational data and application objectives. The most commonly available observational data records only where species have been detected and identified, known as presence-only data. The authors of [113] developed the popular MaxEnt model using maximum entropy to estimate the population intensity. More recently, the connections between Poisson point processes and MaxEnt have been used to develop presence-only data models [38]. Other data collection protocols, like *eBird*, record both when species are and are not detected. These "presence-absence" datasets are typically modeled using a variety of statistical and machine learning methods including additive logistic regression, random forests, and boosted regression trees. Occupancy models [96] account for imperfect detection of species by explicitly modeling hierarchically

linked observation and occupancy processes, resulting in stronger ecological inferences [51, 63]. Species distribution models have also been extended to capture population dynamics using cascading models [127], Brownian Bridges [59], circuit theory [102], and non-stationary predictor response relationships [36]. Recent extensions to joint species distribution models focus on modeling the unobserved environmental factors which potentially drive the correlated distribution [52], and for spatiotemporal dynamics based on Gaussian processes [148]. In machine learning literature, multi-label classification [163, 67, 166, 116, 70, 71] is also related to our work and can be applied to multi-species modeling. Most research in multi-label classification is based on ensemble of classifier chains (ECC), which is different from our approach and cannot provide direct information about the species correlation matrix. Among these previous work, [52], which also uses the latent random variables to model correlations, is most closely related to ours. However, their model can only handle a few (no more than 10) random variables to infer the unobserved factors which potentially drive the correlated distribution, and it ignores the interaction between species. In contrast, our DMSE method can handle hundreds of latent random variables for each species and can quantitatively measure the interaction among species. In the experiments section, we show that our DMSE model outperforms many aforementioned models.

Our model is also inspired by embedding methods which are widely applied to many areas, including music [20], language [9, 103], online purchase behavior [117] etc. The core idea is to learn a vector (or other structure) to represent each of the data points, so that the interaction in the vector space reflects the semantic meaning in the original data. Embedding methods have been proven to have better generalization performance and to provide a better data visualization as

well. [122] presents an embedding model that assumes an exponential family of conditional distributions, similar to Generalized Linear Models [100], to link observed quantities to latent embeddings that capture the semantic relationships of interest. Our DMSE model was developed independently of [122][1]. While the probit model used in DMSE is in the exponential family, DMSE differs fundamentally from the work in [122]: The DMSE framework considers two heterogeneous contextual information feature sets (environmental features and interspecies relationships), it uses a deep neural network at the latent quantity level to extract high-level feature from environmental covariates and it couples the environmental and species embeddings into a predictive multi-species distribution model. It would be interesting to adapt the embeddings proposed in [122] and incorporate them into our DMSE setting. To our knowledge, we are the first ones to apply embedding methods with deep neural network structure to multi-species modeling.

2.1.4 Experiments

We work with crowd-sourced bird observation data collected from the successful citizen science project eBird [105]. One record in this dataset is often referred to as a "checklist" in which the bird observer reports all the species he/she detects as well as the time and geographical location of the observation site. Crossed with the National Land Cover Database for the U.S. [58], we can estimate the landscape composition of each observation site l_i with 15 different land types such as the percentage of the water, forest, grass, etc. For the use of training and testing, we transform all this data into the form (b_i, l_i) as described in the first

[1]We thank Liping Liu and David M. Blei for bringing up to our attention and discussing the Exponential Family Embedding in personal communications.

paragraph of the Multi-Species Modeling section. The dataset for this experiment is formed by picking all the observation checklists from the Bird Conservation Region [25] (BCR) 13 in the last two weeks of May from the 2002-2012 which contains 39154 observations. May is a migration period for BCR 13, therefore a lot of non-native birds pass over this region, which gives us excellent opportunities to observe their habitat choice during the migration. Here we choose the top 100 most frequently observed birds as the species collection which covers 97.6% of the records in our dataset. In the experiments, we use a 5-fold cross validation to validate the multiple choices of hyper-parameters as well as evaluate the stability of models and we observe no overfitting between the loss on the validation vs test set during cross-validation.

What do embeddings look like?

We start by giving a qualitative impression of the embeddings produced by our method and visualized by t-SNE algorithm [94], which is a popular visualization method that can visualizes high-dimensional data by giving each datapoint a location in a two-dimensional map while, to a large extent, maintaining their original proximity.

Fig.2.4 visualizes the embeddings of environmental preference and interactive behavior (s_j and λ_j) of each species. In the picture, we manually assign the species into four main categories according to their habitat preference[2]: **(1) Birds living near residential areas**, such as House Sparrow, Common Grackle, American Robin, Blue Jay, Mallard, Canada Goose, etc. Most of them are easy to find in the backyards, city parks, parking lots and agricultural fields. The presence

[2]We get the habitat preference of birds from the website www.allaboutbirds.org

ratio of these species are more than 25% of the records since they are easy for bird-watchers to find. **(2) Birds living in wetlands**, such as Swamp Sparrow, Northern Rough-winged Swallow, Killdeer, etc. that live near the water but mainly feed on insects. **(3) Water birds**, such as gulls, herons and cormorants which need a large amount of open water. **(4) Birds living in forest and pasture**, such as warblers, woodpeckers, nuthatches, thrushes, hawks, etc. These kinds of birds always live in the forest, grassland, pasture, shrubs, or near forest edges. These are the four categories that do not overlap with each other.

One can see in the left map of Fig.2.4, the birds of the same category cluster tightly. For example, the birds living near the human settlements are all on the left, the birds living in wetlands and the water birds are on the right-top corner. Since the birds living in forest and pasture have a large habitat range, we further highlight three taxa in this category: the warblers, the woodpeckers and the raptors. It is interesting to note that the birds within a taxon have a high similarity of habitat preference which coincides with the field observation.

When it comes to the embeddings of correlation (the right map), it can be observed that in most cases, the species living in similar places have a relative higher correlation. However, one can find some interesting cases comparing the left map and the right map. For example, although the Mallard and Canada Goose are more common to see near human habitation, the occurrence of these two birds still has a high correlation with other water birds. What is more, in the left map, we find that the locations of Blue Jay and American Robin are not very close, but from the right map, we know that they have a very high correlation which coincides with the ecological relationship as we described in the introduction section.

Predictive Performance of DMSE

While the visualization provides interesting qualitative insights, we now provide a quantitative evaluation of the model quality based on predictive power. In our experiment, we analyze the performance of DMSE for modeling both single-species and multi-species. Here we use two metrics to analyze the predictive performance: **(1) Area under Curve (AUC)**, i.e. the area under receiver operating characteristic (ROC) curve which is used to describe the ability of the model to rank outcomes, and **(2) the log-likelihood**, i.e. $\sum_{i=1}^{N} \log \Pr(b_i|l_i)$. which measures the calibration of the predicted probabilities of species occurrence.

DMSE's performance on a single-species model

We compare the single-species predictive performance of DMSE with random forest (RF) model and SVM in terms of AUC. Random forest is one of the standard techniques used to model single species distribution in a wide variety of ecological and conservation applications (e.g. [55, 33]) and SVM is also a popular and robust model which has been successfully used for numerous applications. We implemented RFs and SVMs using python-sklearn. The number of trees in RFs was 1000, which saturated the predictive performance. The kernel of SVMs was RBFs, which perform well across a range of applications. Here we also analyze the effect of the deep neural network in the DMSE model by analyzing the performance of a DMSE model, in which we only use projection matrix W to embed the environmental features. We test these four models on different species from very common to rarely seen. As shown in Fig.2.5, the deep neural network gives us a significant boost on the predictive power of DMSE. We expect

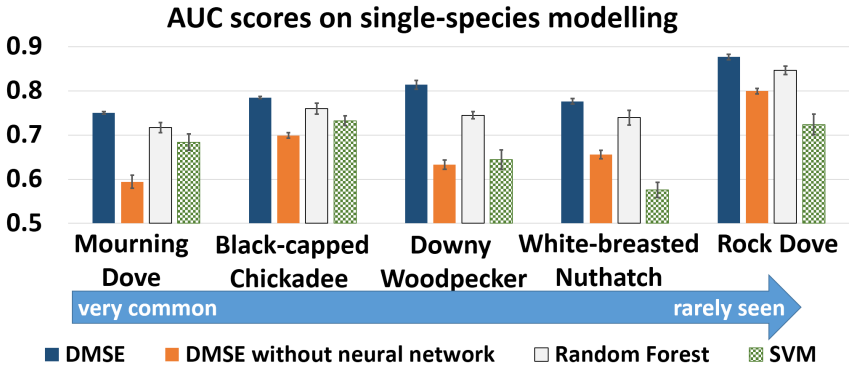


Figure 2.5: With the help of neural network, our single species version DMSE outperforms other models in terms of AUC.

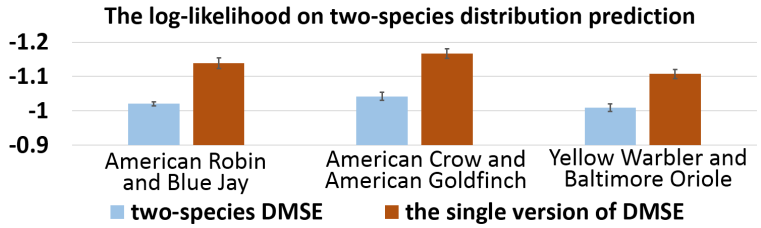


Figure 2.6: By modeling the correlation, the two-species DMSE outperforms the single version.

a similar performance boost when we incorporate deep structures into other relevant models, such as the exponential family embedding model in [122]. With the help of deep neural network, our DMSE model outperforms other models.

What are the effects of correlation?

We now explore whether the correlation plays an important role in multi-species modeling. We start by comparing the performance of multi-species DMSE and the single version of DMSE on modeling two-species distribution. The single version of DMSE can be thought of as the original DMSE model but with an identity correlation matrix , which means we model the multi-species distribution by modeling the distribution of each species independently without their correlation. Here we use **log-likelihood** instead of **AUC** to analyze models' per-

Species Name	Species Name	correlation
Red-eyed Vireo	Eastern Wood Pewee	0.607
Common Grackle	Red-winged Blackbird	0.604
European Herring Gull	Great Black-backed Gull	0.580
Yellow Warbler	Common Yellowthroat	0.567
Blue Jay	American Robin	0.535
Common Grackle	American Robin	0.510
Blue Jay	Northern Cardinal	0.504
American Crow	American Robin	0.493
Common Grackle	European Starling	0.475
European Starling	Red-winged Blackbird	0.474

Table 2.1: The list for species pairs with high correlation. The correlation here is derived from covariance matrix Σ.

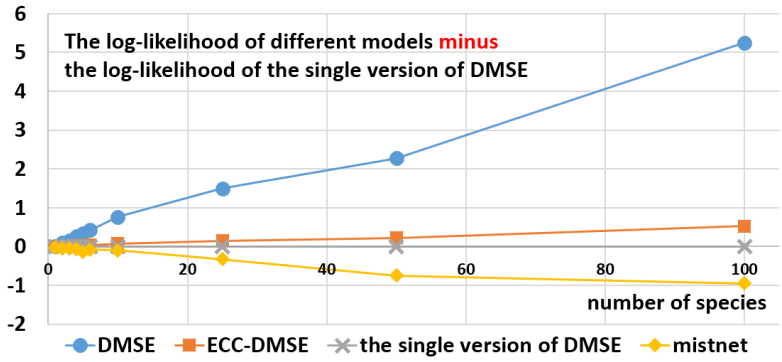


Figure 2.7: As the number of species becomes larger, the performance of multi-species DMSE becomes better and better compared with single species DMSE and other models. This figure shows the performance difference of all models against the single species DMSE model.

formance because the AUC averaged across species still values the distribution of each species separately, which does not fully reflect the benefit of modeling correlation. According to our experimental results, the multi-species DMSE outperforms the single version on all the species pairs that we have tried. Because of space limitation, we only show the performance on 3 pairs of species.

As shown in Fig.2.6, multi-species DMSE has a substantial improvement compared with the single version of DMSE, which reflects the important role of inter-species correlation. Furthermore, we provide Table. 2.1, which quantita-

tively measures the interaction between some species pairs with relatively high correlation. These pairs have been validated by domain experts.

In addition, we compare our DMSE model with **(1) the single version of DMSE**, **(2) the ensemble of classifier chains (ECC) method** [70] which is a standard class of models used for multi-label prediction within the discipline of machine learning (We use single version of DMSE as building blocks of the classifier chain) and **(3) MISTNET model** proposed in [52], a recent extension of species distribution model that models unobserved environmental factors which potentially drive the correlated distribution of multiple species.

During the training and testing, all these models use the same environmental feature as their input, which is the landscape composition of each observation site l_i with 15 different land types. When predicting species, we do not use any information about other species occurring at a given location. The hyperparameters of DMSE are the same as the single version of DMSE , which have been introduced in section 2.2. The hyperparameters of other models follow the setting of original literatures with some reasonable adjustment, which saturated the predictive performance. Finally, we compare all models against the single version of DMSE to show their difference.

In Fig.2.7, as the number of species goes up, the predictive performance of our multi-species DMSE keeps improving and it outperforms other models. We believe that the ensemble of classifier chain method does not perform well mainly because of errors keep cumulating further down the classifier chain. This experiment not only highlights the importance of modeling the correlation between species, but also shows the improvement of DMSE model over previous approaches.

2.1.5 Discussion

In this section, we presented a novel Deep Multi-Species Embedding model that can quantitatively capture inter-species correlations of hundreds of species simultaneously, by jointly embedding vectors corresponding to multiple species as well as vectors representing environmental covariates into a common high-dimensional feature space via a deep neural network. Our DMSE model significantly outperforms existing models on multi-species distribution modeling. Additionally, we demonstrated the benefit of using a deep neural network for feature extraction and show how that improves the predictive performance of species distribution modeling. The ability to visualize the learned embeddings is also a key feature for easy interpretability and open-ended exploratory data analysis. However, DMSE has not fully explored prior knowledge in terms of the structure of the interactions between entities. As a result, the current learning method relies on a sequential MCMC based sampling, which is the bottleneck of the training process. In the following section, we improve the model by exploiting the low-rank covariance structure of those interactions, which results in an efficient parallel learning process.

2.2 Boosting Multi-label Classification via Incorporating Low-rank Covariance Structure

2.2.1 Introduction

Understanding multi-entity interactions is a central question in many real-world applications. For example, in computational sustainability [41, 95], it is important to understand the spatial distribution of species and how species interact with each other and their environment, for developing conservation plans. In computer vision, the detections of multiple objects are often correlated because of the shared background and scenario [152]. In natural language processing, a text often has several correlated labels in terms of its topic, emotion, and semantic meaning [108]. The multivariate probit model (MVP) [3] is a popular classic model for studying interactions of multiple entities. Nevertheless, learning the multivariate probit model is challenging because it involves the integration of a multivariate normal distribution over a constrained space.

A classic approach for optimizing the MVP model is Bayesian Inference [21, 144], where the posterior distribution is simulated by Markov Chain Monte Carlo (MCMC) methods [66] and the maximum likelihood estimates are obtained by a Monte Carlo version of the Expectation Maximization (EM) algorithm. These approaches require the simulation of observations from a multivariate truncated normal distribution involving an arbitrary covariance matrix. Although observations from a multivariate truncated normal distribution can be sampled from a sequence of univariate truncated normal distributions [39], the computational effort is rather heavy for high-dimensional problems. Extensions of the classic

MVP in specific domains have been proposed under specific assumptions of the covariance matrix (see, for example, [132, 162]). Recent approaches for computing the maximum likelihood of MVP have been proposed using the first-order gradients and the second-order information [18, 98]. Those approaches integrate MCMC methods and the numerical estimation of the multivariate probit model [39], which is based on an importance sampling using the truncated normal distribution.

The accessibility of massive contextual data, as well as the success of deep learning, provide additional opportunities and challenges for boosting MVP. On the one hand, massive contextual data, such as millions of high-resolution images, create the possibility of improving predictive performance, particularly when integrated with deep neural networks, which are remarkably powerful for extracting high-level features from raw data. On the other hand, a scalable learning scheme, which integrates well with parallelized infrastructure such as graphics processing units (GPUs) is needed to take advantage of various deep neural networks as well as the massive contextual data. Unfortunately, the classical approaches such as Bayesian inference or previous gradient-based methods, inevitably contain sequential inferences, such as MCMC simulations, which are typically not easy to implement on GPUs. A recent approach called Multi-Entity Dependency Learning via Conditional Variational Auto-encoder (`MEDL_CVAE`) [147] is compatible with deep neural networks and exploits GPUs, with competitive wall-clock training time, but suffers from two key limitations. On one hand, `MEDL_CVAE` learns the joint likelihood by optimizing the variational lower bound of the joint likelihood but has no guarantee concerning the gap between the lower bound and the true likelihood. A second limitation is that the empirical optimization of the variational lower bound in `MEDL_CVAE` suffers from

the KL-vanishment problem, which is a known problem in applications based on a variational auto-encoder. As a result, when integrating it with powerful deep neural networks such as Convolutional Neural Networks, the KL term decreases dramatically to zero, which causes serious overfitting problems that restrict its performance.

We propose **a novel end-to-end learning scheme for the Deep Multivariate Probit Model (DMVP), which is scalable and flexible with various deep neural networks**. Specifically: **(1)** We introduce the *Deep Multivariate Probit Model* (DMVP), a deep generalization of classic MVP, in which a flexible deep neural network is embedded to extract the high-level features from the raw data. **(2)** We propose an efficient parallel sampling process based on the low-rank covariance structure of the interactions among entities, which transforms the integration over a high-dimensional constrained space into an expectation over the residual multivariate normal distribution with a variance strictly lower than that of the rejection sampling, tightly integrates with various deep neural networks, and can be implemented end-to-end on GPUs. **(3)** We provide both a theoretical and an empirical analysis of the convergence behavior of the sampling process embedded in DMVP. We provide theoretical convergence guarantees for DMVP as well as a numerical analysis of the convergence behavior based on a tighter bound, which is much closer to the empirical results. Our theoretical bound also sheds light on the trade-offs between performance and convergence. **(4)** We apply DMVP to three multi-entity modelling problems. In the first application, we use the crowd-sourced *eBird* dataset combined with the National Land Cover Database for the U.S. (NLCD) [58] and satellite images to study the interactions among multiple species. In the second application, we study the deforestation and human encroachment in the Amazon rainforest with high-resolution satellite

images. In the last application, we study the associated concepts of real-world web images using the NUS-WIDE-LITE web image dataset collected from Flickr [22].

Preview of results: We show that our DMVP (a) trains significantly faster than classic MVP models using the end-to-end learning scheme fully implemented on GPUs; (b) captures correlations among multiple entities in all applications; and (c) outperforms the approaches that assume independence among entities conditioned on contextual data, classic MVP models, recent gradient-based MVP methods, and the recent variational approach `MEDL_CVAE`.

2.2.2 Preliminaries

Notations

We use $\phi(x;\mu,\Sigma)$ and $\Phi(x;\mu,\Sigma)$ to denote the density function and the cumulative distribution function of the multivariate normal distribution with mean $\mu \in \mathbb{R}^l$ and covariance $\Sigma \in \mathbb{R}^{l\times l}$, i.e,

$$\phi(x;\mu,\Sigma) = \frac{1}{(2\pi)^{l/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} \tag{2.13}$$

$$\Phi(x;\mu,\Sigma) = \int_{-\infty}^{x_1} ... \int_{-\infty}^{x_l} \phi(s;\mu,\Sigma).\mathrm{d}s_1...\mathrm{d}s_l \tag{2.14}$$

where $|\cdot|$ denotes the determinant of a matrix.

For the sake of simplicity, we use $\Phi(x)$ to denote the CDF of one-dimensional standard normal distribution.

For the comparison between vectors, we use "$\preccurlyeq$" to denote the "element-wise

less or equal to", i.e,

$$a \preccurlyeq b \text{ iff } \forall i, a_i \leq b_i \tag{2.15}$$

Deep Multivariate Probit Model

The multivariate probit model (MVP) is described in terms of a multivariate normal distribution of the underlying latent variables that are manifested as binary responses through a threshold specification. More specifically, given the dataset $D = \{(x_i, y_i)|i = 1, ..., N\}$, where $x_i \in \mathbb{R}^m$ is the m-dimensional contextual data and $y_i \in \{0, 1\}^l$ is the l-dimensional binary label, MVP maps the Bernoulli distribution of each binary label $y_{i,j}$ to a sequence of latent variables $r_i = (r_{i,1}, ..., r_{i,l})$ through the threshold 0, where r_i is subject to a multivariate normal distribution, i.e,

$$\begin{aligned} &\Pr(y_{i,j} = 1|x_i) = \Pr(r_{i,j} > 0) \\ &\Pr(y_{i,j} = 0|x_i) = \Pr(r_{i,j} \leq 0) \\ &\text{where } r_i \sim N(\mu(x_i), \Sigma). \end{aligned} \tag{2.16}$$

More specifically, the marginal likelihood is,

$$\begin{aligned} \Pr(y_{i,j} = 1|x_i) &= \Phi\left(\frac{\mu(x_i)_j}{\sqrt{\Sigma_{j,j}}}\right) \\ \Pr(y_{i,j} = 0|x_i) &= \Phi\left(-\frac{\mu(x_i)_j}{\sqrt{\Sigma_{j,j}}}\right), \end{aligned}$$

and the joint likelihood is,

$$\begin{aligned} \Pr(y_i|x_i) &= \int_{A_1} ... \int_{A_l} \phi(s; \mu(x_i), \Sigma) \mathrm{d}s_1 ... \mathrm{d}s_l \\ \text{Here } A_j &= \begin{cases} (-\infty, 0] & \text{if} \quad y_{i,j} = 0 \\ [0, +\infty) & \text{if} \quad y_{i,j} = 1 \end{cases} \end{aligned} \tag{2.17}$$

Let $D^i = \text{diag}(2y_i - 1) \in \{-1, 0, 1\}^{l \times l}$ which is a diagonal matrix using vector $2y_i - 1$ as its diagonal. Then, we can further reduce formula (2.17) into the CDF of a

multivariate normal distribution, using the affine transformation, i.e.,

$$\Pr(y_i|x_i) = \Phi(0; -\mu_i', \Sigma_i'), \tag{2.18}$$
$$\text{where } \mu_i' = D^i\mu(x_i) \text{ and } \Sigma_i' = D^i\Sigma D^i.$$

Learning the classic MVP model involves estimating the coefficient W of the linear function $\mu(x_i) = Wx_i$ and the covariance matrix Σ. Usually, both the coefficient matrix W and the covariance matrix Σ are learnt from data, but in some cases the variance matrix Σ can be computed directly from data. For example, the model in [98] used linear kernel for the covariance matrix, where the covariance matrix is the sum of a linear kernel matrix and a diagonal noise matrix computed from the raw input data.

Taking advantage of the successful development of deep learning, we introduce the Deep Multivariate Probit Model (DMVP), which is a deep generalization of the classic MVP, where $\mu(x_i)$ changes from the linear function Wx_i to a non-linear function $\theta(x_i)$, learnt via a deep neural network, and the covariance matrix Σ is always learnt from the data. In this way, the DMVP obtains the flexibility as well as the predictive power of various deep neural networks while modelling the correlations of multiple entities by fitting the correlations of the latent variables.

2.2.3 End-to-End Learning for DMVP

The generic learning methods of MVP are maximum-a-posteriori estimation and maximum likelihood estimation. Because we have introduced the deep neural networks into the DMVP, we train DMVP by maximizing the log-likelihood,

which is the most commonly used method for the training of neural networks, i.e.,

$$\underset{\theta,\Sigma}{\operatorname{argmax}} \sum_i \log \Pr(y_i|x_i) = \underset{\theta,\Sigma}{\operatorname{argmax}} \sum_i \log \Phi(0; -\mu_i', \Sigma_i').$$

The difficulties with respect to learning the DMVP are mainly due to the computation of equation (2.18) as well as its gradients, which are obtained by integrating over a high-dimensional constrained space of latent variables. As pointed out by [97], there is no closed form solution for equation (2.18), and to date can only be estimated via sampling methods.

End-to-End Sampling Process for DMVP

The vanilla rejection sampling estimates $\Phi(0; -\mu_i', \Sigma_i')$ by counting the rate that a sample r from $N(-\mu_i', \Sigma_i')$ satisfies $r \leqslant 0$. However, because the value of $\Phi(0; -\mu_i', \Sigma_i')$ could be exponentially small, on average, it could take exponentially many trials to get merely one trial that satisfies the condition. One straightforward solution for this estimation, which has been adopted in [18], is to use the MCMC approaches to estimate the distribution over the truncated high-dimensional space. Another importance sampling method proposed by [39] uses Cholesky factorization to compute the equation (2.18). This method transforms the sampling of a truncated multivariate normal distribution into the sampling of a sequence of univariate truncated normal distributions, where the truncation of each univariate normal distribution depends on the samples of all preceding random variables. Because both the MCMC method and the importance sampling require a sequentially dependent sampling, they cannot easily integrate with parallelized deep learning infrastructure such as GPUs. Therefore, we propose a novel parallel sampling method to address this approximation problem.

Though there is no closed form for computing the CDF of a general multivariate normal distribution, the one-dimensional CDF $\Phi(x)$ has very accurate analytical estimation [23], which has been implemented in almost all machine learning tools. Inspired by this fact, we decompose the covariance matrix Σ into $V + \Sigma_r$, where V is a diagonal positive definite matrix and Σ_r is the residual covariance matrix, so that a random variable $r \sim N(0, \Sigma)$ can be decomposed as the subtraction of two random variable $z \sim N(0, V)$ and $w \sim N(0, \Sigma_r)$, i.e., $r = z - w$. Thus, the estimation of $\Phi(0; -\mu, \Sigma)$ in equation (2.18) can be transformed into the expectation of the product of l one-dimensional CDF's, conditioned on the residual multivariate normal distribution w, i.e.,

$$
\begin{aligned}
&\Phi(0; -\mu, \Sigma) = \Pr(r - \mu \preccurlyeq 0) \quad r \sim N(0, \Sigma) \\
&= \Pr(z - w \preccurlyeq \mu) \quad z \sim N(0, V),\ w \sim N(0, \Sigma_r) \\
&= E_{w \sim N(0, \Sigma_r)}[\Pr(z \preccurlyeq (w + \mu) | w)] \quad z \sim N(0, V) \\
&= E_{w \sim N(\mu, \Sigma_r)} \left[\prod_{j=1}^{l} \Phi\left(\frac{w_j}{\sqrt{V_{j,j}}} \right) \right] \\
&= E_{w \sim N(V^{-1/2}\mu, V^{-1/2}\Sigma_r V^{-1/2})} \left[\prod_{j=1}^{l} \Phi\left(w_j \right) \right] \\
&\approx \frac{1}{M} \sum_{k=1}^{M} \left(\prod_{j=1}^{l} \Phi\left(w_j^{(k)} \right) \right).
\end{aligned} \tag{2.19}
$$

Knowing that the role of the parameter V is to rescale the sample w_j, without loss of generality, we can assume that V is an identity matrix and directly learn the "rescaled" residual multivariate normal distribution. That is, **in the rest of the paper as well as the Figure (2.8), we use the identity matrix I to replace V.**

Main idea: The high-level idea of our end-to-end learning scheme for DMVP is based on the transformation shown in equation (2.19). The intuition behind our transformation is similar to the *Rao-Blackwell theorem* [11], which improves an estimator by computing its expectation, conditioned on a sufficient statistic. In our case, instead of using a sufficient statistic, we use the residual distribution

w, which fully captures the correlations of the original multivariate distribution. Conceptually, given input features x_i and labels y_i, DMVP first learns the mean and the covariance of the residual multivariate normal distribution via a deep neural network. Then, DMVP samples batches of independent samples $w^{(k)}$ from the residual multivariate normal distribution and uses equation (2.19) to compute the estimation of the joint likelihood. **This sampling process outperforms previous estimation methods in several aspects. First**, the process samples from an explicit distribution, which is significantly more efficient than MCMC-based methods, which need to burn a lot of intermediate samples to reach the implicit distribution. (We show the experimental results in the subsection 5.) **Second**, the variance of this sampling process is strictly smaller than the vanilla rejection sampling (See Appendix A.1), therefore DMVP requires fewer samples, since every sample from this process provides non-trivial information. **Third**, this sampling process can be implemented in parallel on GPUs, which is not the case for MCMC.

Figure (2.8) depicts the detailed learning framework implemented in DMVP. The feature network, composed of multi-layer convolutional networks or fully connected networks, extracts high-level features from the contextual data source to learn the μ for each data point. The choice of the feature network depends on the type of contextual data and the problem, but is flexible enough to be any structure that could be boosted by GPUs. In DMVP, the residual covariance matrix Σ_r is a global parameter, which is learned from random initialization and shared by all data points. To ensure that Σ_r is a semi-positive definite matrix, we actually form the residual covariance matrix by the product of one matrix and its transpose, i.e., $\Sigma_r = \Sigma_r^{1/2}(\Sigma_r^{1/2})^T$. The random variable generator generates batches of the standard normal distributed random variable $z^{(k)}$ in parallel on GPUs. Then,

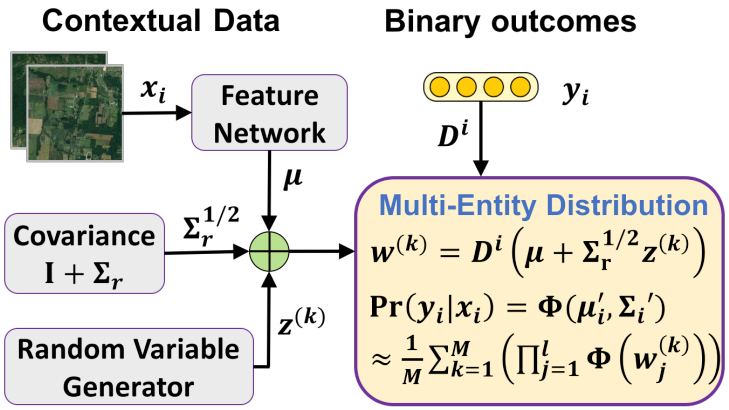


Figure 2.8: The overview of the parallelized learning framework of the Deep Multivariate Probit Model.

using $\Sigma_r^{1/2}$, μ, and the diagonal matrix D^i corresponding to y_i, DMVP computes batches of samples $w^{(k)} = D^i(\mu + \Sigma_r^{1/2} z^{(k)})$. According to the affine transformation of the normal distribution, the samples $\{w^{(k)}\}$ are subject to the multivariate normal distribution $N(D^i\mu, D^i\Sigma_r D^i)$, which is the desired residual multivariate normal distribution as derived in equation (2.18) and (2.19). Because there is no dependency among those samples, all the operations described above could be computed in parallel using tensor operations. Therefore, we can integrate DMVP with various deep neural networks and implement it end-to-end on GPUs using popular machine learning packages (such as Tensorflow or PyTorch).

Theoretical Analysis of the DMVP's Convergence Behavior

In terms of the convergence behavior of this sampling process, we provide a theoretical analysis with respect to the estimation error. Since the estimate $\prod_{j=1}^{l}\Phi\left(w_j^{(k)}\right)$ is bounded between 0 and 1, Hoeffding's inequality guarantees exponentially fast convergence in M between the r.h.s of equation (2.19) and

$\Pr(y_i|x_i)$, i.e.,

$$\Pr\left[\left|\frac{1}{M}\sum_{k=1}^{M}\prod_{j=1}^{n}\Phi(w_{i,j}^{(k)}) - \Pr(y_i|x_i)\right| \geq \epsilon \Pr(y_i|x_i)\right]$$
$$\leq 2e^{-M\epsilon^2 \Pr^2(y_i|x_i)}. \tag{2.20}$$

Though equation (2.20) converges exponentially fast, the value of $\Pr(y_i|x_i)$ could be the magnitude of 2^{-l}. That is, we may need to sample $O(2^{2l})$ many times to have a reasonable multiplicative error bound. To address this issue, another assertion can be proven for this sampling process using Chebyshev's inequality:

Theorem 1 *Let $\mu \in \mathbb{R}^l$ and $\Sigma \in \mathbb{R}^{l\times l}$ be the rescaled mean and rescaled residual covariance matrix of the random variable $w^{(k)}$ in equation (2.19), then we have*

$$\Pr\left[\left|\frac{1}{M}\sum_{k=1}^{M}\prod_{j=1}^{l}\Phi(w_{i,j}^{k}) - \Pr(y_i|x_i)\right| \geq \epsilon \Pr(y_i|x_i)\right]$$
$$\leq \frac{\Phi\left(0; \begin{bmatrix}-\mu\\ -\mu\end{bmatrix}, \begin{bmatrix}\Sigma + I & \Sigma\\ \Sigma & \Sigma + I\end{bmatrix}\right)/\Phi^2(0; -\mu, \Sigma + I) - 1}{M\epsilon^2} \tag{2.21}$$
$$\leq \frac{\left(\frac{\Phi(0;-\mu,2\Sigma+I)}{\Phi(0;-\mu,\Sigma+I)}\right)^2 |2\Sigma + I|^{1/2} - 1}{M\epsilon^2} \tag{2.22}$$
$$\leq \frac{\prod_{i=1}^{l} g(\mu_i)^2 |2\Sigma + I|^{1/2} - 1}{M\epsilon^2} \tag{2.23}$$

where $g(\mu_i) = \max_x \frac{\Phi(\sqrt{2}x+\mu_i)}{\Phi(x+\mu_i)}$. See Appendix A.1 for a more detailed proof.

The function $g(\mu_i)$ in the theorem (1) does not have a closed form but it is a monotonous decreasing function, which converges to 1 as μ_i increases. Figure 2.9 is the visualization of function $g(\mu_i)$. As can be seen, the function $g(\mu_i)$ is very close to 1 when μ_i is positive. Though $g(\mu_i)$ increases exponentially with an upper bound $\sqrt{2}e^{\frac{3-2\sqrt{2}}{2}\mu_i^2}$ when μ_i is a very small negative number, the training method - maximum likelihood estimation - ensures that most μ_i are positive. In

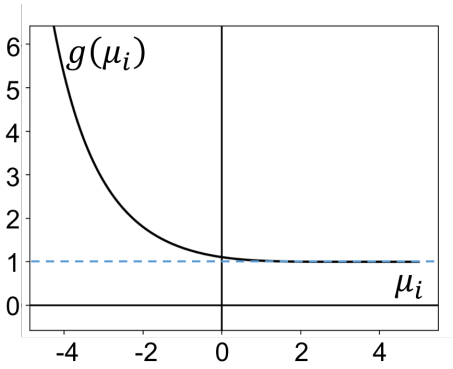


Figure 2.9: The visualization of function $g(\mu_i)$.

theorem (1), the equation (2.21) is the upper bound derived by exact analysis of the second moment of the random variable $\prod_{j=1}^{l} \Phi(w_{i,j}^{k})$. Knowing there is no general closed form for the CDF of multivariate normal distribution, we further derive the equation (2.23) to provide an analytical upper bound.

Though, in the worst case, the upper bounds could be exponentially large with respect to the dimensionality, this still sheds light on the convergence behavior of our sampling process. For example, if the distribution of entities is independent, then the rescaled residual covariance Σ is a zero matrix. In that case, the variance of our sampling process is zero, so that we only need to sample once to get the exact likelihood. In more general cases, if the rescaled residual covariance Σ is a low-rank matrix, most eigenvalues of the matrix $2\Sigma + I$ are 1, which indicates a small $|2\Sigma + I|$. According to our experiments, most eigenvalues of the rescaled residual covariance matrix Σ are very close to 0, which supports the empirical convergence behavior of our DMVP. In the experimental subsection, we provide more detailed analysis in terms of the performance as well as the convergence behavior of DMVP with a low-rank residual covariance matrix, showing that the DMVP's performance only degrades significantly when the rank of the residual covariance matrix is extremely small.

Since our learning scheme is based on stochastic gradient descent, we also use

the derivatives of equation (2.19) as the estimation of the true derivatives. The variance analysis of the derivatives of $\Pr(y_i|x_i)$, which has a similar convergence bound as theorem (1), could also be derived using the similar method. Because of space limitations, see Appendix A.1 for a more detailed proof.

2.2.4 Other Related Work

Multi-entity modelling problems are studied extensively under the names of multi-label classification, multi-entity embedding and structured prediction. The simplest approach is to model the distribution of each entity independently, given the contextual data, known as the *binary relevance model*. This approach is quite popular in multi-label image classification because of its simplicity and flexibility. (We chose it as a baseline for this problem.) However, this could perform poorly, particularly when certain labels are rare or some are highly correlated. Therefore, max-margin [125], ranking losses [32] and embedding methods [122] have been used to address the correlations. Along this line of research, recent approaches [7, 6] use SSVM minimizer to optimize energy-based structured models. Those approaches mainly focus on the classification problem, in which the correlation among entities is implicit and therefore it is hard to derive the structured probabilistic distribution of entities. Our applications of DMVP, on the other hand, focus more on probabilistic modeling rather than classification. Another classic approach related to MVP is the Conditional Random Field (CRF) [80], which offers a general framework for structured prediction based on undirected graphical models. Instead of using correlated latent variables, CRF models the correlation among entities directly, where the joint probability of multiple outcomes is proportional to an energy function. However, optimizing CRF models

suffers from the computational intractability of the partition function. To remedy this issue, [157] applied ensemble methods and [27] proposed a special CRF for problems involving specific hierarchical relations. Nevertheless, optimizing CRF models still inevitably depends on gibbs sampling for approximate inference, and has the same problem as the MCMC-based MVP models. A newly proposed ecological model, the Deep Multi-Species Embedding model (DMSE) [18], introduces deep neural networks into the classic MVP. Nevertheless, the learning methods of DMSE are also based on sequential inference such as the MCMC simulations, so that they are not easily boosted using GPUs.

The mixed-logit model [101] is another statistical model for analyzing discrete outcomes, whose marginal likelihood is similar to the formula of the transformation step in DMVP. However, the mixed-logit model is a general way to inject random variables into the logistic regression while the transformation in DMVP uses the auxiliary residual covariance to estimate the likelihood. Multi-Entity Dependency Learning via Conditional Variational Learning (`MEDL_CVAE`) uses a conditional variational auto-encoder to handle correlation between multiple entities, and is also compatible with parallelized deep structures. Despite its limitations, as discussed in the introduction, `MEDL_CVAE` is a state-of-the-art multi-entity modelling method and is also closely related to our DMVP model. Therefore, we chose `MEDL_CVAE` as the representative approach among those competitive multi-entity modelling methods and compare its performance to DMVP, in the experimental subsection.

2.2.5 Experiments

Datasets and Implementation Details

We evaluate our DMVP[3]. on three datasets of multi-entity modelling problems.

eBird is a crowd-sourced bird observation dataset collected from the successful citizen science project *eBird* [106]. One record in this dataset is referred to as a checklist in which the bird observer records all the species he/she detects as well as the time and the geographical location of the observational site. Crossed with the National Land Cover Dataset for the U.S. (NLCD) [58], we obtain a 15-dimensional feature vector for each observational site which describes the landscape composition with respect to 15 different land types such as water, forest, etc. We also collect the satellite images for each observation site by matching the geographical location of the observational site to Google Earth[4]. Each satellite image covers an area of 12.3km^2 near the observation site and has 256×256 pixels. The dataset for this experiment is formed by picking all the observation checklists from the Bird Conservation Region (BCR) 13 [25] in the last two weeks of May from 2004 to 2014, which contains 50,949 observations. We choose the top 100 most frequently observed birds as the target species which cover over 95% of the records in our dataset.

Amazon is the Amazon rainforest landscape satellite image dataset collected for Amazon rainforest landscape analysis,[5] in which raw images were derived from Planet's full-frame analytic scene products using 4-band satellites in sun-

[3]Code to reproduce the experiments can be found at https://bitbucket.org/DiChen9412/icml2018_dmvp

[4]https://www.google.com/earth/. Google Earth has already conducted preprocessing including cloud removing on the satellite images.

[5]https://www.kaggle.com/c/planet-understanding-the-amazon-from-space.

Dataset	eBird	Amazon	NUS
#Training Set	40759	27545	44493
#Validation Set	5095	3443	5561
#Test Set	5095	3443	5561
#Entities	100	17	81

Table 2.2: the statistics of the *eBird* and the *Amazon* dataset

synchronous orbit and International Space Station orbit. The organizers used Planet's visual product processor to transform raw images into 3-band 256x256-pixel jpg format. The *Amazon* contains a total of 34,431 samples and each sample in this dataset contains a satellite image chip covering an area of 0.9 km^2 in Amazon rainforest. The chips were analyzed using the Crowd Flower[6] platform to obtain a ground-truth composition of the landscape. There are 17 different labels for each satellite image chip, which represent a reasonable subset of phenomena of interest in the Amazon basin such as atmospheric conditions, common land cover phenomena, and land use phenomena.

NUS-WIDE-LITE is a light version of the *NUS-WIDE* datasets collected by the National University of Singapore [22], which contains 55,615 samples and each sample is the low-level features (such as wavelet texture, histogram, correlogram, etc) of the real-world web image associated with tags from Flicker. The 81 tags represent 81 different concepts related to the web images, such as the concepts related to the objects in the image (dog, cat, building, etc) and the concepts of the background (clouds, sunset, etc). For the ease of presentation, we use **NUS** to denote this dataset.

We randomly split the datasets into three parts for training, validation, and testing. The details of the three datasets are listed in table 2.2.

[6]https://www.crowdflower.com/

Performance Analysis of the DMVP on Multi-Entity Modelling Problems

We compare the proposed DMVP with baseline models from three different groups. The first group, which we refer to as *conditional independent model* (CIM), assumes independence among entities, conditioned on the contextual data. Within this group, we chose different models based on the type of the input features. For example, when the input features are images, we choose to use convolutional neural networks (CNN), while we use the multi-layer fully connected neural network (MLP) for one-dimensional feature inputs. For the sake of fairness, the structure of CIM as well as the feature networks in other baseline models are always the same as the feature network of DMVP. More specifically, for the data resources of low-level features, such as the NLCD features of **eBird** dataset and the **NUS** dataset, we use a 4-layer fully connected neural network with hidden units of size 128, 256, 256, l, where the activation function of the first 3 layers is ReLU [107] and there is no activation function in the last layer. For the image data resources, we use a CNN similar to the Alexnet [77] with some minor adjustments. The second group is the *previously proposed Multivariate Probit Model*, which can also model correlations among entities, but uses different inference methods. Within this group, we chose the Deep Multi-Species Embedding (DMSE) model [18], a gradient-based MVP model, which uses the numerical computing method proposed by [39] to estimate the likelihood and a MCMC-based method to estimate the gradients. This model represents a wide class of MCMC-based multivariate probit models while further improving the classic MVP by taking advantages of the flexibility of deep neural networks to obtain useful feature extractions. Nevertheless, its training process involves MCMC approaches as well as the sequential importance sampling, and therefore cannot be integrated on GPUs. For the last group, we chose the `MEDL_CVAE`[147] model,

which is a *state-of-the-art multi-entity modelling approach* proposed recently. This model uses conditional variational auto-encoder to handle correlation between multiple entities, in which it approximates the joint likelihood by its variational lower bound.

Because we study multi-entity modelling problems, in our experiments, we use Negative Joint Distribution Log-likelihood (Neg.JLL) as the indicator of a model's performance: $-\frac{1}{N}\sum_{i=1}^{N}\log \Pr(y_i|x_i)$, where N is the number of samples in the test set. Based on the theorem (1) we obtain 1,000,000 samples from the residual multivariate normal distribution for testing DMVP's performance, which is sufficient to guarantee the accuracy of the estimation. However, for the training, DMVP empirically converges well with only 100 samples.

All the training and testing process of our DMVP and other baseline models, which are compatible with the GPUs, are performed on one NVIDIA Quadro P4000 GPU with 8GB memory. The training and testing process for the DMSE model is performed on Intel(R) Core(TM) i7-7700K CPU@4.20Gz with 8 cores. Since the bottleneck of the DMSE model is on the MCMC sampling, which could not be parallelized trivially, additional cores do not improve the wall-clock time significantly. The whole training process lasts 200 epochs, using the batch size of 128, Adam optimizer [74] with learning rate of 10^{-4} and utilizing batch normalization [64], 0.5 dropout rate [133] and early stopping to accelerate the training process and to prevent overfitting for not only DMVP but all baseline models.

Table 2.3 shows the average performance of DMVP as well as other baseline models on the 3 datasets (4 different type of input features) in terms of the nega-

eBird-NLCD				
Method	**CIM**	**DMSE**	MEDL_CVAE	**DMVP**
wall-clock time (mins)	**2**	1200	10	10
Neg.JLL	34.96	30.53	30.86	**29.68**
eBird-Images				
Method	**CIM**	**DMSE**	MEDL_CVAE	**DMVP**
wall-clock time (mins)	**820**	>3000	847	843
Neg.JLL	34.14	N/A	33.68	**28.26**
Amazon				
Method	**CIM**	**DMSE**	MEDL_CVAE	**DMVP**
wall-clock time (mins)	**484**	>3000	502	495
Neg.JLL	1.70	N/A	1.64	**1.50**
NUS-WIDE-LITE				
Method	**CIM**	**DMSE**	MEDL_CVAE	**DMVP**
wall-clock time (mins)	**4**	1410	12	12
Neg.JLL	6.17	5.76	5.82	**5.73**

Table 2.3: Comparison of various methods on 3 datasets (4 different input features) in terms of the Negative Joint Log-likelihood (the smaller the better) and the wall-clock time.

tive joint log-likelihood (Neg.JLL) and the wall-clock time of training.[7] There are multiple key results in Table 2.3: **(1)** By comparing the Neg.JLL of the conditional independent model (CIM) with other models, one can observe **significant advantages of modelling the correlations among entities**. **(2) DMVP trains more than 100 times faster than the MCMC-based DMSE model** in terms of the wall-clock time. This huge gap between DMVP and DMSE is due not only to the parallelization but also to the advantage of sampling from an explicit distribution. For DMVP, empirically we only need to sample 100 samples per data point to converge very well and every sample here is an unbiased estimation of the joint likelihood. However, in DMSE, we need to burn every 1000 intermediate samples to merely get one quasi-unbiased sample from the implicit distribution, which is not cost-efficient. What's more, the high-resolution image data resources are way beyond the capacity of the MCMC-based method, where the DMSE model cannot reach a reasonable performance after 2-day-long training. **(3)** In terms of the Neg.JLL, **DMVP outperforms all baseline models** including the

[7]We thank the authors of [147] and [18] for sharing the codes.

competitive MEDL_CVAE model, which is compatible with deep neural networks and also models the correlations among entities. There are two reasons of why DMVP outperforms MEDL_CVAE: (i) DMVP directly learns the joint likelihood while MEDL_CVAE approximates the joint likelihood by optimizing its variational lower bound. (2) there is a KL-vanishment issue, which is notorious in all applications based on variational autoencoder, in the training of variational lower bound that hampers the performance of the MEDL_CVAE model.

Empirical Analysis of the DMVP's Convergence Behavior

In subsection 2.2.3 we provide the theoretical upper bound of the DMVP's convergence behavior. Based on the theorem (1), one way to reduce the sampling variance is to assume the low-rank property of the residual covariance matrix. Therefore, we conducted the empirical analysis of the DMVP's performance as well as the convergence behavior on three datasets with residual covariance matrix of different rank. Based on the theorem (1), we use the numerators of both equation (2.21) and equation (2.23) to indicate the convergence rate, where the former is a tighter bound without an analytic form and the latter is the theoretical upper bound. Though the tighter bound (equation (2.21)) does not have a analytic form, we could show the value estimated using the numerical method proposed in [39]. What's more, because the value of the indicators derived from equation (2.23) and equation (2.21) vary across data points over time, we pick the median at the end of the training as the representative. We implement the constraint of $rank(\Sigma_r)$ by restricting the dimensionality of $\Sigma_r^{1/2}$, i.e., $\Sigma_r^{1/2} \in \mathbb{R}^{l \times k}$, where $k \leq l$. Figure (2.10) show the experimental results conducted on three datasets. Because of the similarity, for the **eBird** dataset, we only show

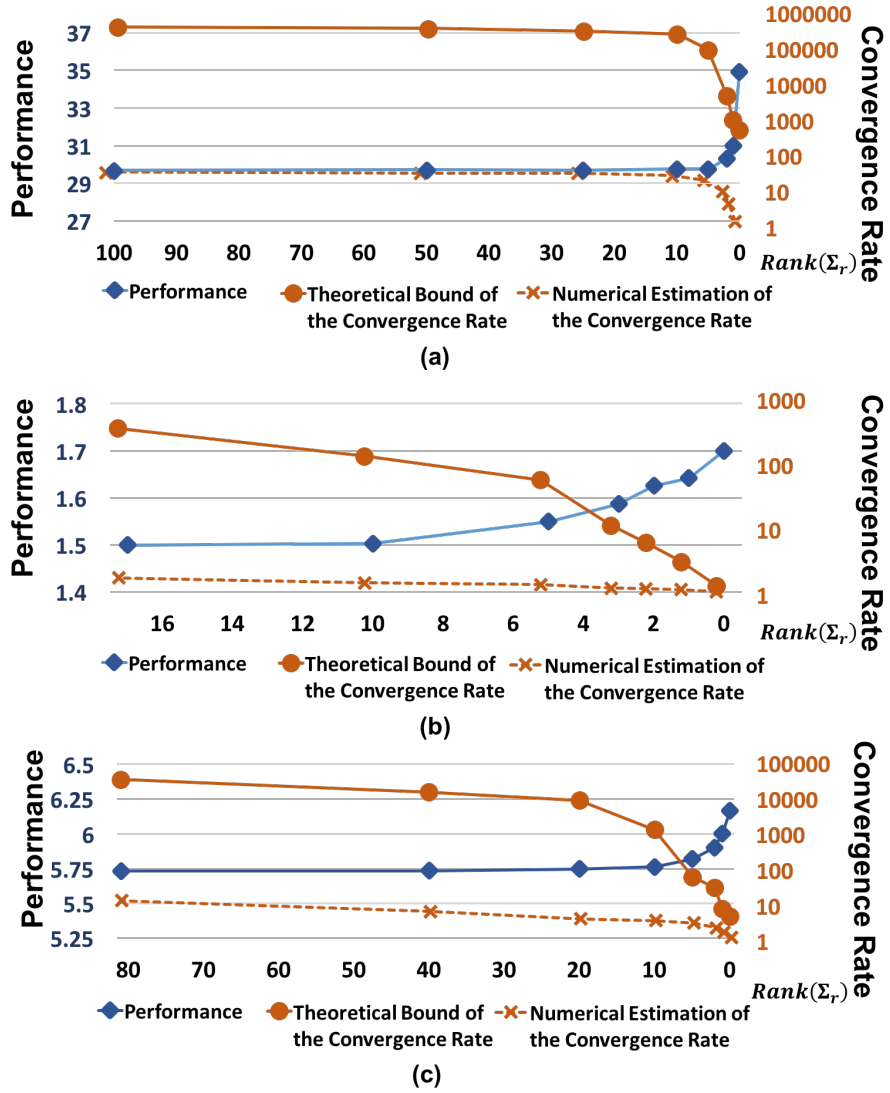


Figure 2.10: The analysis of DMVP's performance and the convergence behavior on three datasets with respect to low-rank residual covariance matrix. The performance is indicated by Neg.JLL and the convergence rate are measured using both the theoretical bound derived from equation (2.23) as well as the numerical estimation of the tighter bound derived from equation (2.21). (For both of them, the smaller the better.) As the rank of Σ_r goes lower, the DMVP converges better while the performance of DMVP only degrades significantly when the rank of Σ_r is extremely low. The subplots (a) (b) (c) correspond to **eBird**, **Amazon** and **NUS** respectively.

the analysis using NLCD features. One observation from Figure (2.10) is that the theoretical bound is way looser than the numerical estimation of the tighter bound, which is actually closer to the empirical results. In our experiments, DMVP converges well in all datasets using only 100 samples. Nevertheless, the

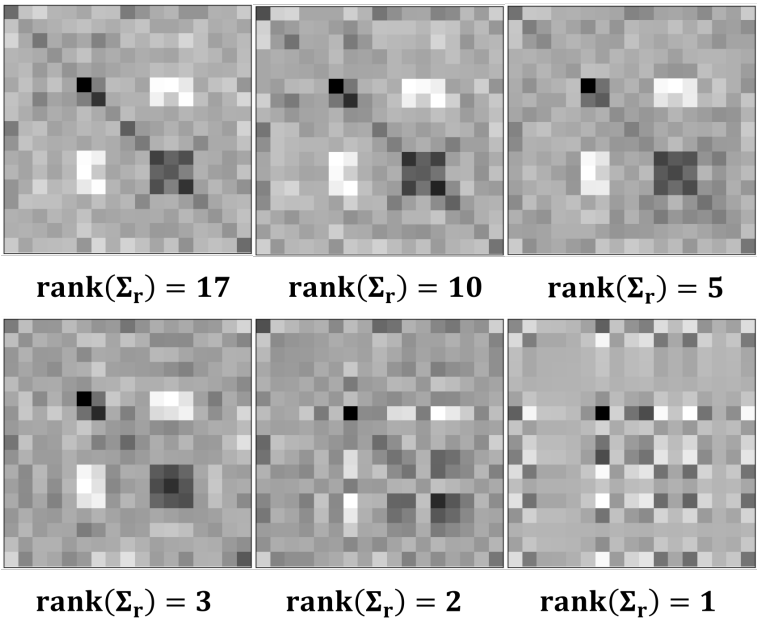


Figure 2.11: The visualization of the residual covariance matrix (Σ_r) on the **Amazon** dataset, with Σ_r of different ranks, which capture the correlations in different resolutions. The pattern of Σ_r only degenerates with extremely low ranks. (less or equal to 3)

theoretical bound still sheds light on the convergence behavior of DMVP. One can see, as we restrict the rank of the residual covariance matrix to be lower, both the theoretical bound and the numerical estimation of the convergence rate become better while the performance of DMVP only degrades significantly when the rank of Σ_r is extremely small. The reason behind this phenomenon is that the rank of Σ_r actually describes the the resolution of how fine-grained DMVP models the residual covariance. Therefore, it is possible to approximate a full-rank matrix by a low-rank matrix with minimal discrepancy. As an example, the Figure 2.11 is the heatmap of the residual covariance matrix on **Amazon** dataset with rank from full-rank to rank-1. (Because of the space limitation, we only show the covariance heatmap of **Amazon** datasets.) One can see, the pattern of residual covariance does not change too much until the resolution is extremely low. These facts are consistent with the empirical results of learning DMVP with full-rank residual covariance matrix, where most eigenvalues of Σ_r are very close to zero. Based on these observations, we can naturally balance the

computational complexity and the predictive performance of DMVP by tuning the resolution. This provides the potential benefits of using DMVP to analyze large scale multi-entity correlation with low-rank constraints.

2.2.6 Discussion

In this section, we proposed an end-to-end learning framework called the Deep Multivariate Probit Model (DMVP), in which we use an efficient parallel sampling process to exploit prior knowledge of the low-rank covariance structure of the interaction, to integrate DMVP with various GPU-boosted deep neural networks. Tested on three real-world applications of multi-entity modelling, we show that DMVP trains 100 times faster than previous MCMC-based methods (DMSE), captures rich correlations among entities, and consistently performs better than previous models. We further provide both theoretical and empirical analysis of DMVP's convergence behavior, revealing the benefits of balancing the computational complexity against the predictive performance by restricting the rank of the residual covariance matrix.

DMVP provides a solid foundation for the learning of multi-entity interactions. Later on, we generalize DMVP to multi-target regression tasks for species abundance estimation [75], and the low-rank structure based acceleration mechanism in DMVP is further applied to follow-up works based on the multivariate probit model for multi-label classification and multi-property prediction tasks [4, 76]. Moreover, the interpretability of the entity embedding in DMVP inspires the design of more complicated interpretable latent spaces for other works.

CHAPTER 3

INCORPORATING PRIOR KNOWLEDGE INTO DEEP LEARNING FOR UNSUPERVISED DEMIXING TASKS

In this chapter, we introduce Deep Reasoning Networks (DRNets), which is a general framework that integrates pattern recognition with prior knowledge reasoning for unsupervised demixing tasks. DRNets are greatly motivated by a complex scientific discovery task that concerns inferring crystal structures of materials from X-ray diffraction data (crystal-structure phase mapping). Given the scientific complexity of this domain, we start by introducing DRNets on an analogous but much simpler task: disentangling two overlapping hand-written Sudokus (**Multi-MNIST-Sudoku**) (see Fig. 3.1). Both de-mixing tasks require probabilistic reasoning to interpret noisy and uncertain data while satisfying a set of rules: thermodynamic rules and Sudoku rules, respectively. For example, de-mixing handwritten digits is challenging, but it is more feasible when we reason about the rules concerning the two overlapping Sudokus. Crystal structure phase mapping is substantially more complex. In fact, crystal structure phase mapping easily becomes too complex for experts to solve and is a major bottleneck in high-throughput materials discovery. Moreover, unlike the Multi-MNIST-Sudoku, where we can generate massive instances from the MNIST dataset, scientific tasks such as crystal-structure phase mapping often have only hundreds of data points and no labeled training data, which greatly challenges classical data-hungry deep learning models. Therefore, supervision by constraint reasoning is strongly desired, and strongly motivated by extensive prior knowledge from sources ranging from fundamental principles to the intuitive experience of scientists. As shown in the following sections, when the prior knowledge is sufficiently rich, as is common in many scientific applications, DRNets outperform traditional

supervised learning, and can even compensate for a dearth of labeled data, by exploiting prior knowledge and magnifying it with reasoning seamlessly integrated into neural network optimization.

3.1 Introduction

Artificial Intelligence (AI)[134] aims to develop intelligent systems, inspired in part by human intelligence. AI systems are now performing at human and even superhuman levels on a range of tasks such as image identification [143], face, [146] and speech recognition [47]. AI also has the potential to accelerate scientific discovery dramatically.[40, 5, 145, 124, 140, 78] Recent AI achievements have been driven mainly by advances in supervised deep learning[85], which requires large labeled datasets to supervise model training. However, in general, scientists do not have large amounts of labeled data for scientific discovery. They often solve complex tasks using only a few data samples by amplifying intuitive pattern recognition with detailed reasoning about prior knowledge to make sense of the data. Such a hybrid strategy has been difficult to automate. Herein we consider crystal-structure phase mapping, a long standing challenge in materials science that is emblematic of the class of scientific problems whose automation constitutes a substantial advancement with respect to the grand challenge of high-throughput unsupervised scientific data interpretation.

Crystal-structure phase mapping involves separating noisy mixtures of X-ray diffraction (XRD) patterns into the source XRD signals of the corresponding crystal structures, a task for which labeled training data are typically not available. Furthermore, a valid phase diagram of the crystal structures of a given chemical system must satisfy thermodynamics rules (Fig. 3.1a-f). Herein we provide a

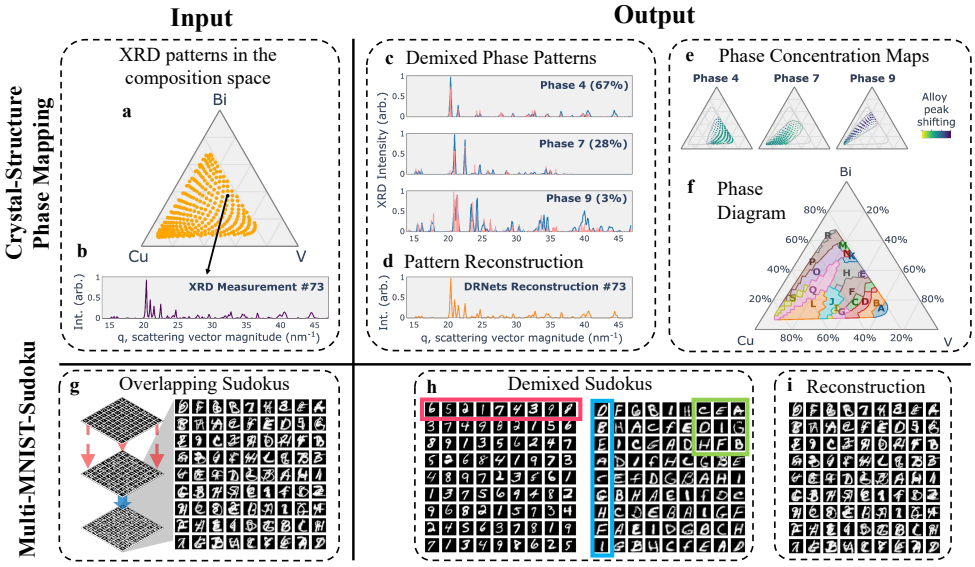


Figure 3.1: **Crystal-Structure Phase Mapping and Multi-MNIST-Sudoku** Phase mapping is a demixing task wherein a phase diagram is inferred from a set of XRD patterns in a materials composition space **(a)**, requiring identification of pure-phase prototypes and their composition-dependent modification. The input **(a-b)** and output **(c-f)** are illustrated for pattern #73 where the DRNets-modified prototypes are shown as sticks in **(c)** for each demixed pattern. For each phase, DRNets output includes the composition map of activation and alloying-based modification from the prototype, shown in **(e)** for 3 phases. The composition regions corresponding to each unique combination of phases is the most salient aspect of the underlying phase diagram **(f)**. In a 9x9 Sudoku, the cells in each row (red rectangle), column (blue rectangle), and any of the nine non-overlapping 3x3 boxes (green square) have all-different digits. In Multi-MNIST-Sudoku, given images of mixed digit pairs, and prior knowledge that they form two overlapping Sudokus (**g**), the goal is to demix the digits into the two original Sudokus (**h**), closely reconstructing the original input images (**i**).

detailed description of how to formulate phase mapping as an unsupervised pattern demixing problem and how to solve it using Deep Reasoning Networks (DRNets) [17]. DRNets is a general framework for combining deep learning with constraint reasoning for incorporating scientific prior knowledge. DRNets is designed with an interpretable latent space for encoding the prior-knowledge domain constraints, enabling seamless integration of constraint reasoning into neural network optimization. Constraint reasoning is a particular type of AI reasoning in which axioms and rules are expressed as constraints and the infer-

ence procedure is a search method. The axioms and rules pertaining to a given task comprise the prior knowledge needed to identify valid solutions. In this manuscript, we show how DRNets requires only a modest amount of (unlabeled) data and compensate for the limited data by exploiting and magnifying the rich scientific prior knowledge about the thermodynamic rules that govern the mixtures of crystals. We further provide insights concerning the interpretability and scalability of DRNets, as well as the role of data and the different DRNets' modules, through a series of ablation studies. DRNets makes this crystal-structure phase mapping advancement by combining learning with constraint reasoning, emulating the analysis of expert scientists and enabling interpretation of complex systems in high-dimensional composition spaces.

Given the scientific complexity of Crystal-Structure Phase Mapping, we provide an initial intuitive explanation of DRNets framework based on Multi-MNIST-Sudoku, [17] a variant of the Sudoku game that involves demixing two completed overlapping hand-written Sudokus (Fig. 3.1g-i). To demonstrate the scalability of DRNets, we also consider 9x9 Sudoku instances combining *both digits and letters*, beyond the 4x4 Multi-MNIST-Sudoku instances involving only digits, used in the original Multi-MNIST-Sudoku variant[17]. We note that, in addition to its intuitive allure, Sudoku represents a logical reasoning task and is a computationally hard combinatorial problem [160]. Thus, Multi-MNIST-Sudoku, with hand-written digits and letters, encapsultates a hybrid reasoning-learning task and provides a tangible demonstration of the value of integrating learning and reasoning for noisy data. The availability of ground truth data also facilitates algorithm comparisons and ablation studies.

Deep Reasoning Networks (DRNets) provides a general framework that

integrates pattern recognition with reasoning about prior knowledge. Both Crystal-Structure Phase Mapping and Multi-MNIST-Sudoku involve identification and demixing of the component signals in mixed-signal input data, specifically crystal phases or handwritten digits and letters. Moreover, for scientific tasks such as crystal structure phase mapping, researchers generally only have access to at most a few hundred (unlabeled) data samples, which greatly challenges classical data-hungry supervised deep learning models. Therefore, to tackle such unsupervised demixing tasks, supervision by constraint reasoning is required and supported by extensive prior knowledge from sources ranging from fundamental physical principles to the intuitive experience of scientists. More specifically, both demixing tasks involve two types of prior knowledge: prototypes of the component signals and rules that govern their mixtures. Both demixing tasks require constraint reasoning to interpret noisy and uncertain data, while satisfying a set of rules: thermodynamic rules, and Sudoku rules, respectively. When considering complex data instances with multiple composition degrees of freedom and many constituent phases, crystal structure phase mapping is substantially more complex than Multi-MNIST-Sudoku and can even surpass the analytical capabilities of human experts.

Complex constraints, such as the thermodynamic rules of phase mapping, are ubiquitous in the physical sciences. Constraint satisfaction and optimization is an impactful approach for domains ranging from satisfiability to sphere packing and protein folding,[120, 46, 35] and is an approach we have explored for phase mapping.[84] The lack of labeled data combined with the realities of experimental data, such as noise and deviations of measured patterns from their idealized prototypes, require simultaneous learning of the de-mixed signals and reasoning about their mixtures, making constraint satisfaction necessary but insufficient for

phase mapping solvers. DRNets encodes complex constraints via a meaningful and interpretable latent representation coupled with a fixed generative decoder that captures the prior knowledge about the domain patterns in an end-to-end deep net framework. Furthermore, the constraint reasoning of DRNets enhances the learning of the shared parameters that govern pattern mixing across multiple (unlabeled) input instances, which in turn facilitates demixing of each pattern in the source dataset. The goal of the present work is to demonstrate the impact of this seamless integration of reasoning and learning for unsupervised pattern demixing tasks, Multi-MNIST-Sudoku as an illustrative example and ultimately crystal structure phase mapping, which are solved by the same general DRNets framework customized with task-specific component models. DRNets for phase mapping tackle a core long standing problem in materials science, outperforming prior methods, which is demonstrated on a benchmark system and by solving the previously unsolved Bi-Cu-V oxide phase diagram. The results contribute to the broader goal of establishing DRNets as a modular end-to-end framework for tasks that require integrating pattern recognition capabilities with reasoning about prior knowledge, which are pervasive in scientific areas as diverse as biology, materials science, and medicine.

3.2 DRNets framework

At a high level, the goal of unsupervised pattern demixing of crystal structures or digits and letters is to infer the base patterns underlying the mixtures observed in unlabeled data [1]. The demixing task is therefore to invert the pattern mixing

[1]We refer to a setting as unsupervised when no labeled training data are available for the target output. For example, Multi-MNIST-Sudoku DRNets are unsupervised since they don't have access to labeled mixed-digit training data.

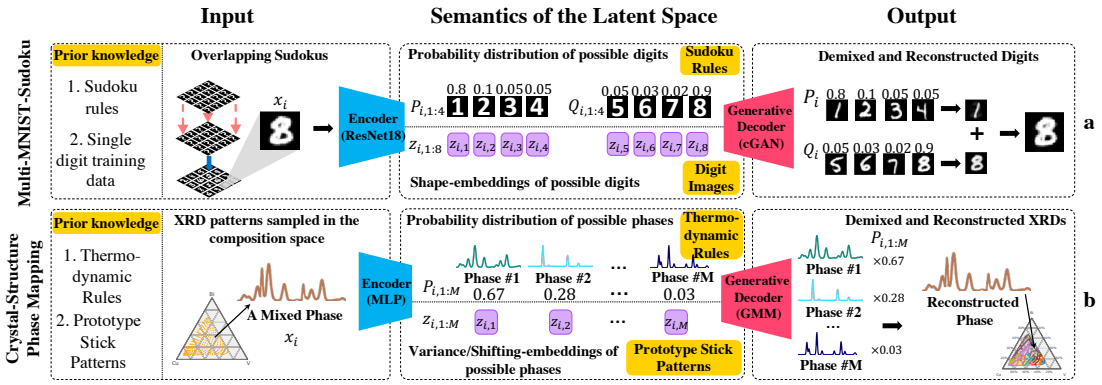


Figure 3.2: **DRNets framework and the semantics of the latent space for different tasks.** In the DRNets framework, an *interpretable* structured latent space is key to incorporating prior knowledge, through the interplay of the encoder, the generative decoder (cGAN trained on single digits or a Gaussian Mixture Model (GMM) based on prototype stick patterns), and the reasoning constraints (Sudoku rules or thermodynamic rules). **a.** In **Multi-MNIST-Sudoku**, DRNets encode the input overlapping digits x_i into $P_{i,1:4}$, $Q_{i,1:4}$ and $z_{i,1:8}$, which denote the probability distribution and the shape embedding of possible digits (1-8). The generative-decoder (cGAN) uses $z_{i,1:8}$ to generate the demixed hand-written digits and reconstruct the original input with the expected overlapping image using $P_{i,1:4}$ and $Q_{i,1:4}$. **b.** In **crystal-structure phase mapping**, DRNets encode the input XRD pattern x_i into $P_{i,1:M}$ and $z_{i,1:M}$, which denote the probability distribution and the variance/shifting-embedding of M possible phases. The generative-decoder (GMM) uses $z_{i,1:M}$ to generate the decomposed phases and reconstruct the original XRD using the phase probability distribution $P_{i,1:M}$.

processes from the data, i.e., the generative processes for each pattern and the way the patterns are combined. However, often unlabeled data do not provide a strong enough pattern signal, motivating reasoning about prior knowledge. DRNets enhance standard unsupervised pattern discovery approaches with prior knowledge about the constraints that govern the patterns, via constraint reasoning, and prior knowledge about the patterns' *shape*, via a fixed generative decoder (see Fig. 3.2). More specifically, DRNets combine deep learning with constraint reasoning in an *end-to-end encoder-generative-decoder framework*, and formulate unsupervised pattern discovery as a *data-driven constrained optimization problem* that *(i)* minimizes a reconstruction loss of the input data, such that *(ii)* the inferred patterns adhere to a given generative model and *(iii)* satisfy domain

constraints. See mathematical details about DRNets' problem formulation flow in Extended Data Fig. 3.13 and Methods.

While standard machine learning approaches can easily handle *(i)*, enforcing *(ii)* and *(iii)* is challenging and emblematic of the limitations of traditional deep learning. The standard approach for incorporating domain knowledge into a deep net architecture is to add terms to the loss function such as various types of sparsity constraints. However, we need to encode more complex constraints, such as combinatorial constraints to express valid Sudoku solutions or thermodynamic rules. For example, a digit cannot appear more than once in a row or an X-ray diffraction pattern cannot be explained by more than 3 prototype phases, or 2 prototype phases if there is alloying. *The challenge of our demixing tasks is that the domain rules capturing prior knowledge involve variables that we do not have direct access to in our problem formulation. In fact, discovering those variables and their values is part of the interpretation task that we are trying to solve.* The challenge is further complicated by the fact that we are operating in an unsupervised setting (no labeled training data). In supervised learning, different strategies have been exploited to incorporate prior knowledge, ranging from placing constraints on the output variables of the deep net to hybrid approaches interleaving symbolic and neural processing. [26] However, in standard unsupervised deep learning approaches, e.g., neural networks autoencoders, the latent space is generally uninterpretable and therefore does not provide a means to express the domain constraints or enforce that the latent patterns conform to the generative model. Furthermore, domain rules are often captured by combinatorial constraints that are not differentiable, and therefore cannot be easily embedded in a deep learning framework. The strategy in DRNets to overcome these challenges is to *(1)* specify *an intended semantics* for the latent space, which means that the latent

space is constructed using variables that have a specific interpretation that can be used in the formulation of the domain rules. For example, in the Sudoku domain, we will introduce a latent variable for each possible digit that gives the probability of that digit being present in the cell associated with the input image. We can now use these variables in constraints on the allowed combinations of digits (see Fig. 3.2). As we will show, these latent variables take on the desired semantics using a small set of unlabeled examples combined with the encoded domain constraints; *(2)* express the domain rules that control the encoding of the latent space in a form amenable to continuous optimization using *entropy-based continuous relaxations; (3)* employ *an optimization formulation whose objective balances the dual needs of minimizing a reconstruction loss of the input data and a reasoning loss that captures the domain constraints* (local constraints involving a single data point or global constraints involving many data points); *(4) use a data-driven approach to jointly solve multiple related (unlabeled) data instances*; and *(5)* solve the data-driven constrained optimization problem using *constraint-aware stochastic gradient descent*, a variant of stochastic gradient descent developed for DRNets that batches together data points involved in the same constraints and *is aware* of the constraints, automatically adjusting the weights of the constraints as a function of their satisfiability.

In DRNets, learning is data driven and reasoning is knowledge driven. Two intertwined processes combine learning and reasoning to discover the values for the encoder's parameters that provide the best interpretation for the interpretable latent space, given the data and prior knowledge: input pattern reconstruction (digits and crystal phases) in conjunction with reasoning about the domain rules (Sudoku and thermodynamic rules). The input pattern reconstruction is performed through the reconstruction loss, with guidance from prototypical

domain patterns (single digits and crystal phases) provided via a fixed generative decoder. The reasoning is performed with ”self-supervision” from the domain rules prior knowledge, encoded as constraints using the interpretable latent space variables and added as entropy-based continuous functions to the loss via Lagrangean relaxation. The reconstruction and reasoning loss constrain the encoding of the latent space to adhere to the domain patterns and rules. We refer to DRNets' formulation as *data-driven constrained optimization* to highlight that even though we design an interpretable latent-space, its semantics are ultimately determined by the quality and quantity of the (unlabeled) data as well as the prior knowledge available, which are critical for conditioning the optimization process to discover the underlying patterns and pattern mixing process across instances. Since often the pattern mixing function is not invertible, the optimization process in DRNets is further conditioned by learning the shared mixing process parameters across instances (e.g. multiple Sudoku instances or related X-ray diffraction patterns). In contrast, standard direct optimization typically considers one instance at a time. Interestingly, as our ablation studies show, while multiple (unlabeled) data are required by DRNets to uncover the pattern mixing process, the amount of (unlabeled) data required is considerably more modest than in standard supervised deep learning settings (see Fig. 3.10 and Fig. 3.3).

Further details about DRNets' problem formulation flow, constraint relaxations, and DRNets' algorithms are given in Extended Data Fig. 3.13, Methods, and Supplementary Methods.

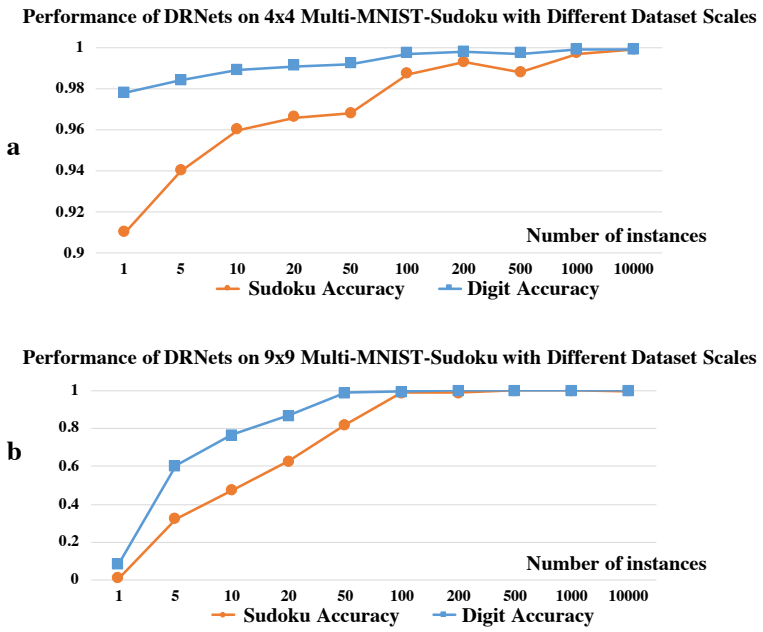


Figure 3.3: **The performance of DRNets on Multi-MNIST-Sudoku tasks with different dataset scales.** Learning over multiple instances significantly (especially, for the 9x9 cases) improves the performance of DRNets. Nevertheless, DRNets can reach 99% Sudoku accuracy with only 100 Multi-MNIST-Sudoku instances, a considerable smaller amount of data compared to standard deep learning approaches.

3.3 Sudoku: demixing handwritten digits and letters

Multi-MNIST-Sudoku consists of demixing digits from two completed overlapping hand-written Sudokus while satisfying Sudoku rules (Fig. 3.1a-c). The prior knowledge comprises the information that a set of images forms two overlapping handwritten Sudokus; each image corresponds to a Sudoku cell with two overlapping handwritten digits/letters; and the Sudoku rules (Fig. 3.1f). Humans tackle Multi-MNIST-Sudoku by interleaving vision clues with reasoning about Sudoku rules, which is emulated by DRNets as illustrated in Fig. 3.1d. DRNets combines deep learning with constraint reasoning and optimization to reason about Sudoku rules. In Multi-MNIST-Sudoku DRNets, a deep neural network encodes a structured latent representation of the input (digit images), which cap-

tures the probabilities and shapes of the possible digits under Sudoku constraints enforced by the reasoning module (Fig.3.2a). The reasoning module comprises the reasoning constraints as well as batch and constraint weight controllers. A fixed conditional generative adversarial network (cGAN) pre-trained on single digits, incorporating prior single digit prototype knowledge, decodes (generates) the individual digit images from their structured latent representation along with their probabilities to reconstruct the mixed input image. The overall objective function of DRNets combines responses from the generative decoder and the reasoning constraints, and is optimized using constraint-aware stochastic gradient descent. We apply entropy-based probabilistic continuous relaxations to encode discrete constraints, such as Sudoku rules, which can be seamlessly incorporated into the objective function. *A key distinguishing feature of DRNets is an interpretable latent space with semantics emerging by the coupling of the encoder, the generative decoder, and reasoning module,* in contrast to standard deep learning methods in which the latent space lacks semantics. Further details about DRNets' components, latent space semantics, problem formulation, and algorithms are given in Fig. 3.2, Fig. 3.4, and Methods.

To evaluate DRNets, we generated 32x32 images of overlapping digits/letters from the test set of MNIST [86] and EMNIST [24], such that every n^2 (n is 4 or 9) images form two n-by-n, overlapping Sudokus (see Supplementary Methods). For the 9x9 case, to distinguish the two overlapping Sudokus, we used letters A-I from EMNIST for the second Sudoku. For ease of presentation, we refer to these letters as "digits" in the following content. DRNets is unsupervised and therefore do not train on labeled mixed digit images; DRNets has only access to single hand-written digits from the MNIST/EMNIST training dataset, which are used to pre-train the generative decoder (cGAN). DRNets significantly outperform state-

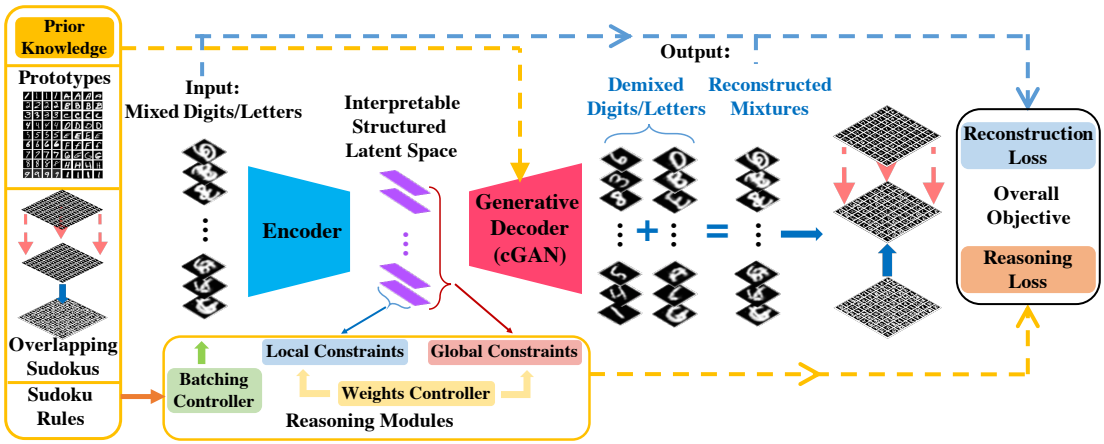


Figure 3.4: **DRNets for Multi-MNIST-Sudoku** DRNets perform end-to-end deep reasoning by using a convolutional neural network to encode an *interpretable* structured latent space that is used by a fixed generative decoder, a conditional generative adversarial network (cGAN), to reconstruct the input mixed digits. The interpretable structured latent space also allows the encoding of reasoning constraints, which enforce that the latent space adheres to prior knowledge about Sudoku rules. Prior knowledge also includes digit prototypes, which are used to pre-train and build the fixed decoder's generative module. An overall objective combines responses from the fixed generative decoder and the reasoning module and is optimized using constraint-aware stochastic gradient descent and backpropagation.

of-the-art supervised MNIST demixing models CapsuleNet [123] and ResNet [53], which in contrast to DRNets are supervised by training on labeled mixed images produced by overlapping digits from the MNIST/EMNIST training dataset (Fig. 3.5).

The intuitive nature of the Sudoku task provides an opportunity to gain insights regarding DRNets' structure and performance via ablation studies. For example, if we replace the fixed cGAN with a (weaker) standard learnable decoder, without prior knowledge about single digits, the optimization process can no longer find the right semantics for the latent space. Moreover, removing the reasoning module also deteriorates the digit accuracy, in addition to the Sudoku accuracy (Fig. 3.5 and Supplementary Note). The final ablation study shows the importance of the data-driven learning of the shared parameters of the

		Methods	Accuracy (%)		Time	Training Set Size	Test Set Size
			Digit	Sudoku			
4x4 Multi-MNIST Sudoku	unsupervised	DRNets	99.90	98.55	28min	N/A	16x10,000 overlapping digits
		DRNets w/o Reasoning **(ablation)**	88.83	15.01	110min		
		DRNets w/o cGAN **(ablation)**	16.02	0.0	57min		
		DRNets + exhaustive search	**100.0**	**100.0**	5.3hours + 28min		
	supervised	CapsuleNet	97.87	50.92	1min + 30min	16x10,000 overlapping digits	
		CapsuleNet + local search1	97.87	57.80	3hours + 30min		
		CapsuleNet + exhaustive search	**100.0**	**100.0**	5.3hours + 30min		
		ResNet-18	97.67	68.47	3min + 2.6hours		
		ResNet-18 + local search1	97.67	88.31	3hours + 2.6hours		
		ResNet-18 + exhaustive search	**100.0**	**100.0**	5.3hours + 2.6hours		
9x9 Multi-MNIST Sudoku	unsupervised	DRNets	**99.99**	**99.23**	7hours	N/A	81x10,000 overlapping digits
		DRNets + local search2	**99.99**	**99.84**	60s + 7hours		
		DRNets w/o Reasoning **(ablation)**	63.82	0.0	7hours		
		DRNets w/o cGAN **(ablation)**	1.24	0.0	7hours		
	supervised	CapsuleNet	99.10	23.33	150s + 41hours	81x10,000 overlapping digits	
		CapsuleNet + local search2	99.10	73.63	210s + 41hours		
		ResNet-18	98.73	35.39	80s + 4.3hours		
		ResNet-18 + local search2	98.73	80.72	140s + 4.3hours		

Figure 3.5: **Comparison of the performance of different methods for Multi-MNIST-Sudoku** We show the "solving time" for unsupervised DRNets and its ablation variants and "test time + training time" for supervised baselines. The test time for CapsuleNet/ResNet + local search includes the local search time. Note that we used two different local search algorithms for 4x4 cases and 9x9 cases. "local_search1" performs an enumeration for the top-2 likely digits in all 16 cells to try to satisfy Sudoku rules. For 9x9 cases, it is impossible to enumerate the top-2 likely digits for 81 cells (2^{81}). Therefore, "local_search2" conducts a depth-first search for digits in each cell from most likely to less likely until it finds a valid Sudoku combination, which is faster than "local_search1". For 4x4 cases, we also applied exhaustive search for all methods, where we enumerate all possible 4x4 Sudokus and return the one with the highest likelihood given our predictions. Note that such strategy is not feasible for 9x9 Sudokus, given there are around 6.67×10^{21} 9x9 Sudokus. The ablation study of removing the reasoning modules (DRNets w/o Reasoning) shows that not only does the Sudoku accuracy degrades, the digit accuracy also degrades, especially for 9x9 Sudokus. The ablation study of replacing the cGAN with a (weaker) standard learnable decoder, without prior knowledge about single digits (DRNets w/o cGAN) shows that both the Sudoku and digit accuracy degrades dramatically.

demixing task across multiple 9x9 Multi-MNIST-Sudoku instances. Nevertheless, DRNets can reach 99% accuracy with only 100 (unlabeled) 9x9 Multi-MNIST-Sudoku instances (Fig. 3.3), a considerable smaller amount of data compared to standard deep learning approaches. More generally, DRNets accomplishes

superior performance through *self-supervision by reasoning* about Sudoku rules, resulting in a better performance for digit accuracy and considerably better for Sudoku accuracy than the supervised systems. The broader implication is that the reasoning component enables DRNets to compensate for the lack of training data, which considerably broadens the purview of AI to problems that have a rich prior knowledge but may lack sufficient training data for traditional deep learning.

3.4 Crystal-structure phase mapping: demixing X-Ray diffraction patterns

Scientific discovery comprises a range of problems where *self-supervision by reasoning* is strongly desired due to the lack of large example datasets, and strongly motivated by extensive prior knowledge, from fundamental principles to the intuitive experience of scientists. In materials science, phase mapping is the problem of inferring the individual phases, i.e., crystal structures, from a collection of X-ray diffraction (XRD) patterns (Fig.3.1a-f), a major bottleneck in research due to the substantial expert analysis required for generating meaningful solutions [92, 48, 79, 135, 43]. As a demixing problem, phase mapping parallels the high level structure of Multi-MNIST-Sudoku where instead of overlapping handwritten digits, an XRD pattern contains a mixture of signals from so-called "pure" phases, and the solution includes demixed signals from each XRD pattern with rules based on a collection of input XRD patterns, as illustrated in Fig. 3.6. The prototypes in phase mapping are *stick patterns* that provide the locations and intensities of peaks in XRD patterns for each known phase (See Fig. 3.6 and

Fig. 3.9), and this set of peaks comprises the entirety of the XRD signal for a single crystal structure, i.e., a pure-phase pattern. Computationally, phase mapping is an NP-hard problem [84] whose sheer number of possible combinations of prototypes in each XRD pattern grows exponentially with data size, ($\sim$ 300 XRD patterns and $\sim$ 200 prototypes), rendering traditional methods computationally infeasible. Supervised methods [110, 140, 109, 87, 14] for phase identification in an XRD pattern and unsupervised methods for phase mapping[135, 43, 121] have been developed and perform well on some datasets, especially when input patterns are akin to simple mixtures of prototypes with mutually distinguishable features. Ternary composition spaces, i.e., mixtures of 3 elements from the periodic table, have great scientific value for discovery of materials with desired properties that is concomitant with complex phase behavior in which different compositions (proportions of the elements) form many unique mixtures of the prototype phases. The complexity is compounded by phenomena such as alloying, wherein each composition's XRD pattern may contain unique variants of the prototypes. The most typical alloying-based variation of a prototype includes altered peak intensities and systematically-shifted peak positions. Peak intensities can also vary from the prototypes due to various ways in which the materials and experiment conditions vary from those used to generate the prototypes. When the variants of prototypes contain strongly overlapped signals with unknown relative peak intensities compared to the prototypes, phase mapping can only be solved (by humans or AI) through reasoning about prior knowledge based on thermodynamics. These "rules" are more nuanced than those of Sudoku and are described in detail in Supplementary Methods. Briefly, when combining 3 elements, the number of phases that can appear in an input pattern is at most 3, and is at most 2 if the composition is in a region that exhibits alloying. This

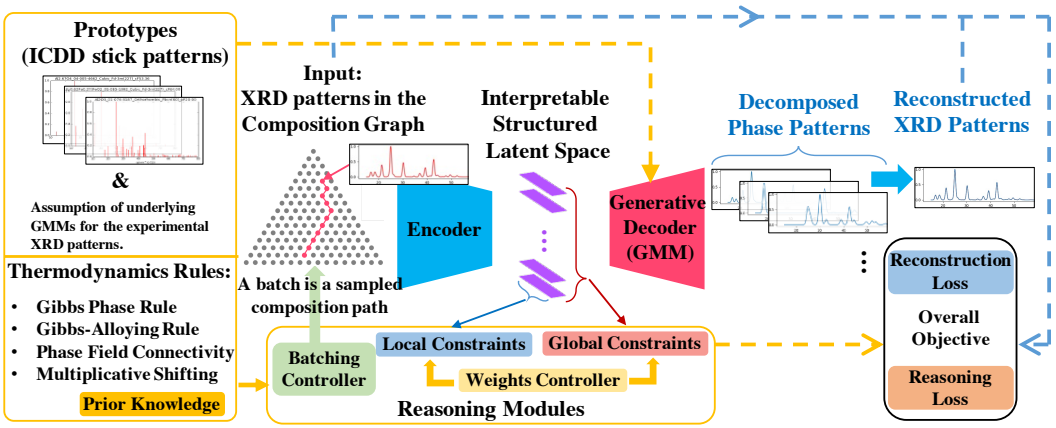


Figure 3.6: **DRNets for Crystal-Structure Phase Mapping** DRNets performs end-to-end deep reasoning by encoding a latent space that is used by a generative decoder to reconstruct the XRD measurements. The input is the XRD patterns, each resulting from a mixture of phases, and the output is the decomposed pure phase patterns and the reconstructed mixture. The encoder is composed of four 3-layer-fully-connected networks. The structured latent encoding is constrained to adhere to thermodynamic rules by the reasoning module. Prior knowledge also includes prototype stick patterns, which are used by the generative decoder, a Gaussian mixture model, to generate the corresponding possible phase patterns in the reconstructed XRD measurement. An overall objective combines responses from the generative decoder, for pattern reconstruction, and the reasoning module, for applying thermodynamic rules, which is optimized using constraint-aware stochastic gradient descent.

latter rule requires consideration of the composition graph of the input XRD patterns, which is also used to enforce a rule wherein each set of prototypes can only appear in compositions that are connected in the graph.

State-of-the-art approaches for phase mapping are unsupervised methods based on matrix factorization, and have incorporated thermodynamic rules to different extents, including analysis of the composition graph [79], integration of demixing with clustering [135], and recent work that interleaves matrix factorization with constraint optimization to enforce all the thermodynamic rules [43]. Nevertheless, these approaches only use known prototype patterns to post-process demixing results, resulting in a more ill-conditioned demixing. In contrast, DRNets provides the first framework that integrates enforcement

of thermodynamic rules with reasoning about the prototypes, solving previously unsolvable phase mapping problems. The DRNets for crystal-structure phase-mapping (Fig. 3.6) seamlessly integrate demixing and reconstruction of XRD patterns by coupling an encoder with four 3-layer-fully-connected neural networks, which produces a two-part structured latent space (Fig. 3.2b), to a generative Gaussian Mixture model that incorporates prior knowledge about prototype phases, and by generating XRD patterns based on mixtures of modified versions of prototypes. The modifications include peak intensity modulation and alloying-based peak shifting; the semantic representation of the prototype-modification parameters in the latent space enables DRNets to learn their optimal values under guidance from priors that are parameterized by prior knowledge of the maximum extent of prototype modification. The latent variables also enable expression of thermodynamic rules with entropy-based functions, which are imposed with a batching sampling strategy to tackle the combinatorics of all-sample thermodynamic constraints (Fig. 3.6). DRNets is optimized with the hybrid objective of reconstructing measured patterns and enforcing thermodynamic rules, as detailed in Methods.

Since ground truth is unavailable for previously-unsolvable experimental phase mapping datasets, we first compare DRNets to state of the art methods NMF-k[135] and IAFD[43] using a synthetic benchmark dataset (with ground truth), based on the Al-Li-Fe oxide system, [135, 83] which contains 6 phases in 15 unique combinations (from 159 prototypes) with substantial alloying. DRNets outperforms NMF-k and IAFD in a variety of metrics. We also considered a recently proposed supervised algorithm[87] in which a deep neural network, trained using XRD patterns simulated from prototypes of known phases, directly predicts the phases present in a given XRD pattern. While such an approach

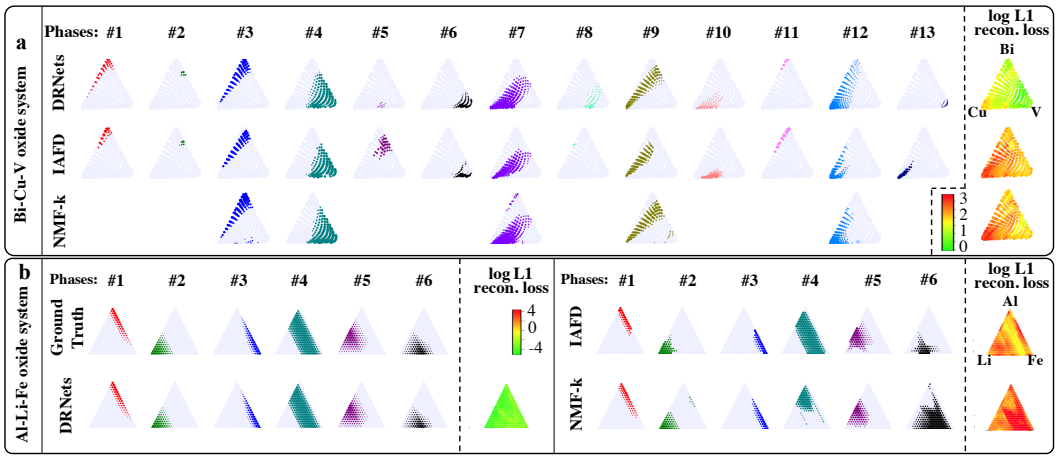


Figure 3.7: **Comparison of the activation maps produced by DRNets with other unsupervised approaches for the Bi-Cu-V oxide and Al-Li-Fe oxide systems.** (**a**) The activation map of the 307 composition points of the Bi-Cu-V oxide system for each of the 13 phases identified by DRNets is shown, with comparison to IAFD and NMF-k solutions (see Fig. 3.9), demonstrating their ability to capture some aspects of the phase activations while misrepresenting or omitting several phases that are key to generating a meaningful phase diagram. The reconstruction loss for each pattern is also shown demonstrating that only through correct identification of the phases can the XRD dataset be fully explained. In (**b**), we highlight the performance of the different methods on the synthetically generated Al-Li-Fe oxide system (231 composition points), which has ground truth: DRNets is the only system that nearly perfectly identifies the phases present in every XRD pattern. The different methods share a common color scale for reconstruction loss in each system, and the elemental labels for the composition triangle are only provided once per system. See further details in Fig. 3.10a.

can be effective for complementing human expertise for a single XRD pattern, it performed poorly on a complex system such as the benchmark Al-Li-Fe oxide system (phase identification accuracy around 1%), which exposes the limitations of a purely simulation-based supervised approach for handling the combinatorics of phase mapping (details in Supplementary Methods). In contrast, the DRNets approach is the only one that perfectly identifies the phases present in every XRD pattern and learns the phase-pure patterns (Fig. 3.8). The DRNets model outperforms other algorithms in a variety of other metrics (See Fig. 3.7b and Fig. 3.10a). In an ablation study we show that DRNets needs around 150 input XRD patterns to approach the ground truth solution in the Al-Li-Fe oxide system

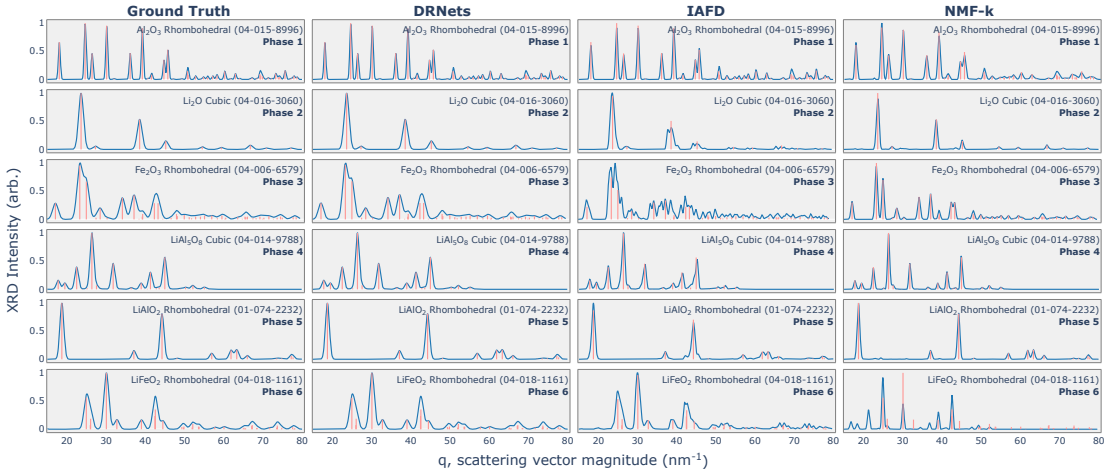
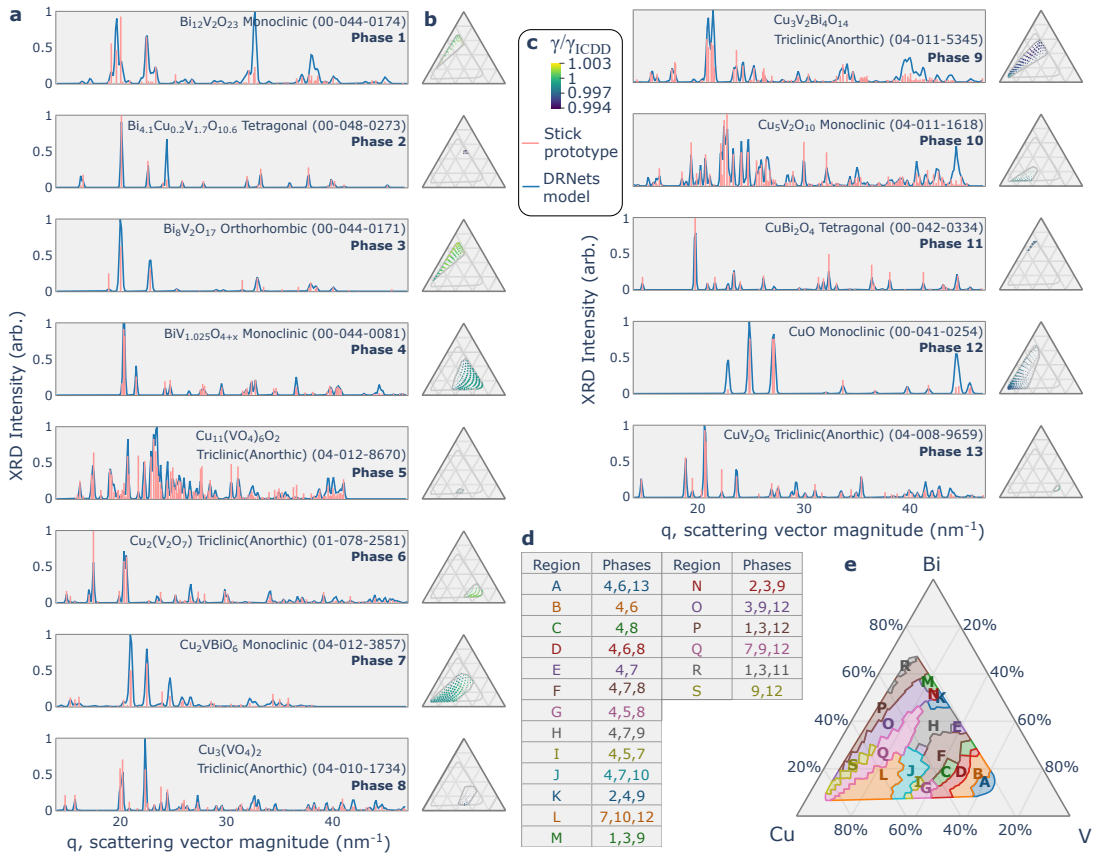


Figure 3.8: **Comparison of the phase patterns discovered by different methods vs. the ground truth phases for the Al-Li-Fe oxide system.** For each phase we plot the pattern of the recongized phase and the ICDD stick patterns. While the phases discovered by DRNets closely match the ground truth phases, some of the IAFD and NMF-k's phases do not match well the ground truth phases (e.g., phase 3 (IAFD) and phase 6 (NMF-k)) as also reflected in the phase fidelity loss (0.00002 (DRNets); 11.920 (IAFD); and 46.156 (NMF-k)); see also Fig. 3.10a).

(Fig. 3.10b). This study highlights the importance of the data-driven learning of the shared parameters of the demixing task across multiple XRD patterns, which enables each pattern to be demixed in a manner that is often not possible with single isolated XRD patterns. Nevertheless, DRNets' data requirements (hundreds of data points) are considerably smaller than those of standard deep learning approaches (hundreds of thousands of data points).

To represent unsolved experimental datasets, we use the Bi-Cu-V oxide system where manual analysis was found to be particularly ineffective to solve the system due to the complexity of the alloying in the set of 307 XRD patterns, as well as strong overlap of signals in the 100 prototypes (Fig. 3.1a-f and Fig. 3.9). DRNets identified 13 phases in 19 unique mixtures (Fig. 3.7a and Fig. 3.9), and the presence of each phase was verified by manual analysis using standard practices based on absence of missing peaks and inability to explain the signal with other



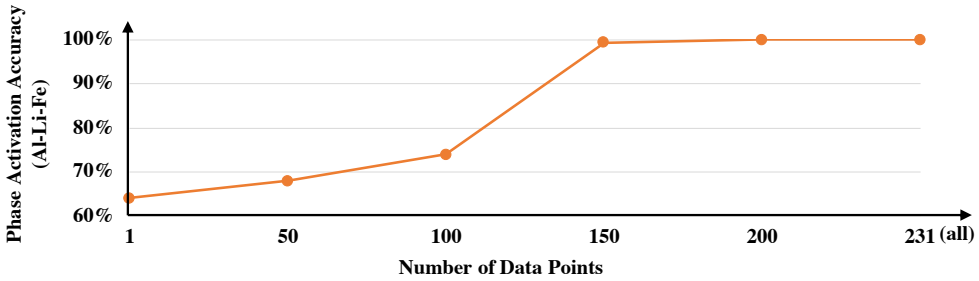
Region	Phases	Region	Phases
A	4,6,13	N	2,3,9
B	4,6	O	3,9,12
C	4,8	P	1,3,12
D	4,6,8	Q	7,9,12
E	4,7	R	1,3,11
F	4,7,8	S	9,12
G	4,5,8		
H	4,7,9		
I	4,5,7		
J	4,7,10		
K	2,4,9		
L	7,10,12		
M	1,3,9		

Figure 3.9: **DRNets' solution for the Bi-Cu-V oxide system.** **a.** The 13 demixed crystal phases for the 307 XRD measurements of the Bi-Cu-V oxide system (each plot includes the signal for the recognized phase and the corresponding ICDD stick pattern). **b.** DRNets' phase concentration maps for each of the phases, where point sizes are proportional to their estimated phase concentrations and heatmap denotes estimated shifting (alloying). **c.** Shows the universal legend for the 13 phases in **a** and **b**, where γ is the average lattice constant. **d.** Table of all phase mixtures in the DRNets solution. **e.** DRNets' crystal phase map for the Bi-Cu-V-O system with phase fields labeled according to **d.**

prototypes. We note that in practice verifying a solution is easier than producing it. For example, visual inspection of Fig. 3.9 reveals the excellent agreement between the stick prototype and the demixed DRNets model for each phase, and this analysis was extended to patterns from the experimental dataset that were chosen based on high activation of each phase and each phase mixture, a manual validation based on representative patterns from the solution. To assess the extent by which learning the pattern interrelationships is critical for solving the

	Methods	# of phases	Activation Accuracy	Reconstruction Loss (L1)	Reconstruction Loss (L2)	Gibbs	Gibbs Alloy	Phase Field Connectivity	Fidelity Loss
Al-Li-Fe	DRNets	6	100%	0.038	<0.001	100%	100%	100%	<0.001
	IAFD	6	80.6%	6.921	0.778	100%	100%	100%	11.920
	NMF-k (k=6)	6	63.1%	29.805	7.169	93.9%	87.0%	71.0%	46.156
Bi-Cu-V	DRNets	13	N/A	3.916	0.268	100%	100%	100%	0.482
	IAFD	13	N/A	11.254	1.781	100%	100%	100%	55.571
	NMF-k (k=5)	5	N/A	10.453	1.577	99.7%	99.7%	48%	48.674
	NMF-k (k=13)	13	N/A	11.258	1.399	52.4%	52.4%	86.9%	68.471

a



b

Figure 3.10: **a. Comparison of different performance metrics for different methods for the Al-Li-Fe oxide system and the Bi-Cu-V oxide system.** Gibbs, Gibbs alloy, and phase connectivity metrics denote the proportion of samples satisfying the Gibbs, Gibbs alloy, and phase connectivity rules; the phase fidelity loss (Fidelity Loss) denotes how well the discovered phase patterns match the ground truth (the lower the better (See Supplementary Methods)). For the Al-Li-Fe oxide system, the DRNets solution has 6 phases, which matches ground truth, and this known number of phases was applied to IAFD and NMF-k. For the Bi-Cu-V oxide system, both DRNets' and IAFD's solutions have 13 phases (the number of phases was specified for IAFD but not DRNets) while NMF-k has 5 phases. Nevertheless, we also run NMF-k with 13 phases following the verification of the presence of the 13 phases from the DRNets solution. Note that, there is no ground truth for the Bi-Cu-V oxide system, therefore the activation accuracy is not applicable (N/A). The results indicate that DRNets performs substantially better than IAFD and NMF-k for all the metrics on both systems (Additional details in Supplementary Methods). **b. The performance of DRNets on Crystal-Structure Phase Mapping (Al-Li-Fe oxide system) with different number of XRD data points.** Learning over multiple XRD patterns within a composition system plays an important role for DRNets to solve crystal-structure phase mapping problem. As shown in the plot, for Al-Li-Fe oxide system, DRNets can almost perfectly recover the phase activation of XRD patterns when it learns via demixing of a collection of at least 150 XRD patterns.

Bi-Cu-V oxide system, we consider a model analogous to DRNets for demixing a single XRD pattern in isolation. This model identifies the same phases as DRNets for only 27% of the patterns, highlighting that the nuanced phase behavior of this system can only be resolved through combinatorial experimentation combined with reasoning about the underlying thermodynamic constraints to learn the

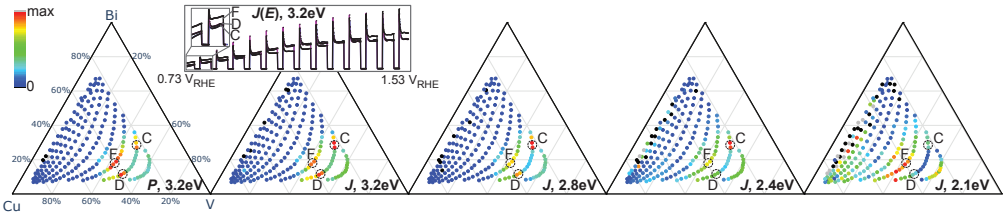


Figure 3.11: **Characterization of Bi-Cu-V oxide library for photoelectrocatalysis of the oxygen evolution reaction, a critical reaction for solar fuels technology.** After XRD and XRF measurements, a grid of compositions was characterized with chronoamperometry (CA) with 4 different light emitting diode (LED) illumination sources from which photocurrent (J) is calculated, as well as cyclic voltammetry with 3.2 eV illumination (CV) from the photoelectrochemical power generation (P) is calculated. The resulting 5 performance metrics are plotted with respect to composition, and select pairs of points from 3 different phase fields in Fig 3.9 are indicated with labels C, D and F. The common false color scale from 0 to a maximum value is used for each metric, with maximum values of 1.8 mW cm^{-2} for P and 13.3, 14.1, 0.5, 0.045 mA cm^{-2} for J with 3.2, 2.8, 2.4 and 2.1 eV illumination, respectively. The anodic sweep of the CV is shown for 3 select samples labeled by their phase region. All 3 of these regions contain $BiVO_4$, a well-known metal oxide photoanode, with much higher Cu concentration than typical Cu-free $BiVO_4$ photoelectrocatalysts. All 3 noted phase regions contain $BiVO_4$ and $Cu_3(VO_4)_2$ with D and F additionally containing Cu_2BiVO_6 and $Cu_2V_2O_7$, respectively. The different compositions and phase combinations lead to different performances, in particular the 3 phase region F exhibits higher photocurrent at low applied bias (see inset) and higher photocurrent with 2.1 eV illumination, which are 2 critical properties for $BiVO_4$ photoanodes that have been historically difficult to optimize. Despite common belief that phase mixtures are deleterious to photoactivity, these results demonstrate alloying and optimal phase mixtures as promising directions for photoanode discovery and optimization.

shared parameters across multiple XRD patterns. As shown in Fig. 3.7a, the DRNets activation maps (the amount of each demixed pattern in each input XRD pattern) for 5-8 of the phases are poorly reproduced by NMF-k and IAFD. The demixed patterns in these solutions, which are intended to be phase-pure patterns, contain mixtures with the less-commonly-occurring phases, making the minor phases undiscoverable by these methods and hampering inference of scientific knowledge from the phase mapping solution. The "fidelity loss" quantifies the deviation of the demixed patterns from its closest prototype (see Supplementary Methods), and the lack of phase purity in the demixed NMF-k and IAFD solutions contribute to their substantial fidelity loss compared to DR-

Nets (Fig. 3.10a). The low fidelity loss of the DRNets solution is commensurate with the manual verification of the presence of phases identified by DRNets, although this analysis does not preclude the false negative detection of a phase in any given XRD pattern. Such an imperfection in the solution would give rise to a reconstruction error, and Fig. 3.10a demonstrates that the reconstruction loss for DRNets is substantially lower than those of other methods, indicating that false negative phase detection is not a major issue in the DRNets solution. Substantial reconstruction loss can also occur if the experimental data contains a phase that is missing from the set of prototypes, which would prompt an investigation of phase discovery, as discussed further in the Supplementary Methods. The activation accuracy metric assesses the phase concentrations in each measured pattern but can be quantified only when ground truth is available, as in the synthetic dataset where DRNets substantially outperform other methods. Datasets with poor signal-to-noise ratio and/or XRD peak widths that do not enable unambiguous phase identification can result in different solutions that still satisfy thermodynamic rules and reconstruct the source data[43]. Evaluating the stability of DRNets' phase mapping solutions with active feedback from experiments is an interesting avenue that we are pursuing.

Informed by the DRNets solution, analysis of the photo-oxidation of water (a critically limiting component of solar fuels technology) revealed that a 3-phase mixture (alloy variants of $BiVO_4$, Cu_2BiVO_6 and $Cu_3(VO_4)_2$) outperforms the standard monoclinic $BiVO_4$ material, defying the conventional wisdom that phase mixtures are deleterious to photoactivity (Fig. 3.11). DRNets' performance for phase mapping is emblematic of how seamlessly combining deep learning with reasoning about prior scientific knowledge can automate the interpretation of scientific data and accelerate knowledge discovery.

3.5 Methods

Mathematical formulation of DRNets. In order to produce a DRNets encoding for a given task, we start by formulating the task as a data-driven constrained optimization problem, which is then transformed through a sequence of steps into a data-driven unconstrained optimization problem amenable to end-to-end optimization via state-of-the-art deep learning technology (Fig. 3.13a). More formally, the formulation of a DRNets task as a data-driven constrained optimization problems is as follows:

$$\min_{\theta} \ \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(G(\phi_\theta(\mathbf{x}_i)), \mathbf{t}_i) \tag{3.1}$$

$$\text{subject to: } \phi_\theta(\mathbf{x}_i) \in \Omega^{\text{local}} \text{ and } (\phi_\theta(\mathbf{x}_1), ..., \phi_\theta(\mathbf{x}_N)) \in \Omega^{\text{global}}$$

$$\text{where } \phi_\theta(\mathbf{x}_i) := (\mathbf{z}_{i,1}, ..., \mathbf{z}_{i,m}, \mathbf{e}_{i,1}, ..., \mathbf{e}_{i,m}) \tag{3.2}$$

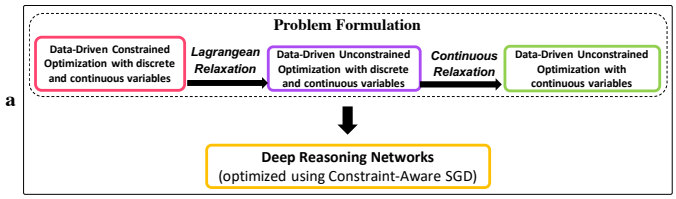
In this formulation, N is the number of input data points, m is the number of possible single patterns, $\mathbf{x}_i \in R^n$ is the i-th n-dimensional input data point, $\mathbf{t}_i$ is the corresponding targeted output, which is in general the input $\mathbf{x}_i$ in unsupervised cases, $\phi_\theta(\mathbf{x}_i) := (\mathbf{z}_{i,1}, ..., \mathbf{z}_{i,m}, \mathbf{e}_{i,1}, ..., \mathbf{e}_{i,m})$ is the latent space of DRNets for data point $\mathbf{x}_i$, a function of the encoder ϕ parameterized by θ, typically a neural network. $\mathbf{z}_{i,j}$ and $\mathbf{e}_{i,j}$ are the *shape* and *probability embeddings* of the possible pattern-j at data point $\mathbf{x}_i$ indicating its shape and probability. $G(\cdot)$ is the generative decoder which involves a fixed pre-trained or parametric generative model that generates single patterns from shape embeddings $\mathbf{z}_{i,j}$ and a process that mixes the generated single patterns factoring in their probabilities. $\mathcal{L}(\cdot, \cdot)$ is the loss function, which evaluates the loss between the output of the generative decoder and the target $\mathbf{t}_i$, Ω^{local} and Ω^{global} are the constrained spaces w.r.t. a single input data point and several input data points, respectively. Note that constraints can involve

Task	Prior Knowledge	Semantics of latent space	Encoder	Generative Model in Decoder	Loss Function
Multi-MNIST-Sudoku (n×n overlapping Sudokus)	Overlapping Sudokus Sudoku Rules Single digit training data	1 – Probability of possible digits for each of the two overlapping digits 2 – Shape embeddings of possible digits for each of the two overlapping digits	1 – ResNet-18 with $2{\times}n$ output units 2 – ResNet-18 with $2{\times}n{\times}100$ output units	Conditional GAN (trained on single digits)	Reconstruction loss: L1 norm Reasoning: entropy-based constraints enforcing Sudoku rules
Crystal-Structure Phase Mapping (m known phases)	Thermodynamic rules Prototype stick patterns (ideal experimental phases) for known crystal phases Phases captured by GMM	1 – Probability of known phases 2 – Variance/Shifting embeddings of known ICDD stick patterns, i.e., estimate of parameters (σ, shift, intensity variance) for for GMM	1 - 3-layer-MLP with m output units 2 - Three 3-layer-MLPs with $m, m, m{\times}K$ output units (K denotes the maximum number of sticks)	Gaussian Mixture Model + Prototype Stick Patterns	Reconstruction loss: JS-distance + L2 loss Reasoning Loss: entropy-based constraints for thermodynamic rules.

Figure 3.12: **Different components of DRNets for the different tasks.**

several (potentially all) data points: e.g., in Sudoku, all digits should form a valid Sudoku and in crystal-structure-phase-mapping, all data points in a composition graph should form a valid phase diagram. Thus, we specify local and global constraints in DRNets – local constraints only involve a single input data point whereas global constraints involve several input data points, and they are optimized using different strategies. For the Multi-MNIST-Sudoku DRNets, $\phi_\theta(\cdot)$ is composed of two ResNet-18, the generative model in $G(\cdot)$ is a pre-trained conditional GAN [104] using hand-written digits; for the Crystal-Structure-Phase-Mapping DRNets, $\phi_\theta(\cdot)$ is composed of four 3-layer-fully-connected networks and $G(\cdot)$ involves a Gaussian Mixture model. Below we provide details for each specific application.

Finally, we note that we refer to *data-driven* constrained/unconstrained optimization problems to highlight the fact that even though we assign an interpretation to the structured latent-space produced by the encoder, $\phi_\theta(\mathbf{x}_i)$, the latent-space semantics are ultimately determined by the data. (See also Fig. 3.12 for a summary of the different components of DRNets for the different tasks.)



b

Cardinality Constraint $e_{i,j} \in \{0,1\}$ j = 1 ...8 $s.t.\ \sum_{j=1}^{4} e_{i,j} = 1$ and $\sum_{j=5}^{8} e_{i,j} = 1$	**Cardinality Constraint Relaxation** $\min_{\theta} H(P_i) + H(Q_i)$ $= -\sum_{j=1}^{4} P_{i,j} \log P_{i,j} - \sum_{j=1}^{4} Q_{i,j} \log Q_{i,j}$
All-Different Constraint For all constrained set S $s.t. \sum_{i \in S} e_{i,j} = 1$ for j = 1 ...8	**All-Different Constraint Relaxation** For all constrained set S $\max_{\theta} H(\bar{P}_S) + H(\bar{Q}_S)$
k-Sparsity Constraint $e_{i,j} \in \{0,1\}$ j = 1 ...M $s.t. \sum_{j=1}^{M} e_{i,j} \leq k$	**k-Sparsity Constraint Relaxation** $\min_{\theta} max\{H(P_M), c\}$, where $c < \log k$

Figure 3.13: **The transformation flow for Deep Reasoning Networks and examples of continuous relaxations.** **a.** The process of formulating a problem into the DRNets framework. (i) We start by formulating tasks as data-driven constrained optimization problems, with discrete and continuous variables. For example, the data-driven constrained optimization task in MNIST-Sudoku is to demix images of two overlapping Sudokus such that the demixed Sudokus satisfy the Sudoku constraints and their reconstruction loss is minimized. Furthermore, in this formulation for MNIST-Sudoku, we assume that a the generative decoder (cGAN) reconstructs the demixed Sudoku digit images using a two-part latent space that encodes the digit probabilities and shapes and that the two-part latent space is produced by two convolutional neural networks (ResNet) and is subject to the Sudoku constraints. (ii) This data-driven constrained optimization problem is converted into a data-driven unconstrained optimization problem using Lagrangean relaxation, which essentially moves the constraints to the objective function, with associated penalty weights. (iii) We use entropy-based continuous relaxations to encode and replace discrete (non-differentiable) constraints with continuous functions, such as sparsity, cardinality, the all-different constraint, and logical constraints. The objective function combines two components: the reconstruction loss of the generative decoder (which for Sudoku corresponds to the reconstruction of the demixed overlapping digit images), and the reasoning loss (which for Sudoku corresponds to the penalty weighted entropy-based continuous function that capture the Sudoku rules). The result of these transformations is the DRNets data-driven unconstrained optimization formulation. (iv) DRNets optimizes the overall objective function using constraint-aware stochastic gradient descent (SGD). Note that we refer to these problems as *data-driven* problems since although we assign semantics to the structured latent-space (probabilities and shapes), their full meaning is ultimately determined by the data. **b.** Examples of the continuous relaxation of discrete constraints. $e_{i,j}$, P_i, Q_i, P_M, and H, represent indicator variables denoting if a given input image contains a given digit, the discrete distribution over digits 1 to 4, the discrete distribution over digits 5 to 8, the discrete distribution over values 1 to M, and the entropy function, respectively. See notation and further details in Supplementary Methods.

Transformation flow for DRNets. Solving the constrained optimization problem in equation (3.2) directly is challenging since the objective function in general

involves deep neural networks, which are highly non-linear and non-convex, and prior knowledge often involves combinatorial constraints, which cannot be directly encoded in a standard deep learning framework. Fig. 3.13a depicts how we the reduce the above constraint optimization problem into DRNets. In particular, we use Lagrangean relaxation to approximate the constrained optimization problem (equation 3.2) with an unconstrained optimization problem, moving the constraints to the objective function with associate penalty weights:

$$\min_{\theta} \ \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(G(\phi_\theta(\mathbf{x}_i)), \mathbf{t}_i) + \lambda^l \psi^l(\phi_\theta(\mathbf{x}_i)) + \sum_{j=1}^{N_g} \lambda_j^g \psi_j^g(\{\phi_\theta(\mathbf{x}_k) | k \in S_j\})$$
$$\text{where } \phi_\theta(\mathbf{x}_i) := (\mathbf{z}_{i,1}, ..., \mathbf{z}_{i,m}, \mathbf{e}_{i,1}, ..., \mathbf{e}_{i,m}) \tag{3.3}$$

Herein, N_g denotes the number of global constraints, S_j denotes the set of indices w.r.t. the data points involved in the j-th global constraint, and ψ^l, ψ_j^g denote the penalty functions for local constraints and global constraints, respectively, along with their corresponding penalty weights λ^l and λ_j^g. As outlined in Fig. 3.13a, we employ two mechanisms to tackle the above unconstrained optimization task: (1) *continuous relaxations* of constraints with discrete variables and (2) *constraint-aware stochastic gradient descent* to tackle global penalty functions, for the different types of combinatorial constraints.

Continuous Relaxations: Prior knowledge often involves combinatorial constraints with discrete variables that are difficult to optimize in an end-to-end manner using gradient-based methods. Therefore, we need to design proper continuous relaxations for discrete constraints to make the overall objective function differentiable. Many approaches have been used to handle continuous relaxations of discrete constraints [60, 156]. We apply a group of entropy-based continuous relaxations to encode general discrete constraints such as sparsity, cardinality, and All-Different constraints. We construct continuous relaxations

based on probabilistic modelling of discrete variables, where we model a probability distribution over all possible values for each discrete variable. For example, in Multi-MNIST-Sudoku, a way of encoding the possible two digits in the cell indicated by data point x_i (one from {1...4} and the other from {5...8}), is to use 8 binary variables $e_{i,j} \in \{0, 1\}$, while requiring $\sum_{j=1}^{4} e_{i,j} = 1$ and $\sum_{j=5}^{8} e_{i,j} = 1$. In DRNets, we model probability distribution P_i and Q_i over digits 1 to 4 and 5 to 8 respectively: $P_{i,j}, _{j=1...4}$ and $Q_{i,j}, _{j=1...4}$ denote the probability of digit j and the probability of digit $j + 4$, respectively. We approximate the cardinality constraint of $e_{i,j}$ by minimizing the entropy of P_i and Q_i, which encourages P_i and Q_i to collapse to one value. See details concerning relaxations for other constraints in Fig.3.13b and Supplementary Methods.

Constraint-Aware Stochastic Gradient Descent: We developed a variant of the standard SGD method that we refer to as constraint-aware SGD, which batches data points involved in the same global constraint together, conceptually similar to the optimization process in GraphRNN [161], to tackle the optimization of global penalty functions $\psi_j^g(\{\phi_\theta(\mathbf{x}_k)|k \in S_j\})$, which involve several (potentially all) data points.

We define a *constraint graph*, an undirected graph in which each data point forms a vertex and two data points are linked if they are in the same global constraint. Constraint-aware SGD batches data points from the randomly sampled (maximal) connected components in the *constraint graph*, and optimizes the objective function w.r.t. the subset of global constraints concerning those data points and the associated local constraints. For example, in Multi-MNIST-Sudoku, the data points (cells) in each overlapping Sudoku form a maximal connected component in the *constraint graph*, so we batch the data points from

Algorithm 1 Constraint-aware stochastic gradient descent optimization of deep reasoning networks.

Require: **(i)** Data points $\{x_i\}_{i=1}^N$. **(ii)** Constraint graph. **(iii)** Penalty functions $\psi^l(\cdot)$ and $\psi_j^g(\cdot)$ for the local and the global constraints. **(iv)** Pre-trained or parametric generative decoder $G(\cdot)$.

1: Initialize the penalty weights λ^l, λ_j^g and thresholds for all constraints.
2: **for** number of optimization iterations **do**
3: Batch data points $\{\mathbf{x}_1, ..., \mathbf{x}_m\}$ from the sampled (maximal) connected components.
4: Collect the global penalty functions $\{\psi_j^g(\cdot)\}_{j=1}^M$ concerning those data points.
5: Compute the latent space $\{\phi_\theta(\mathbf{x}_1), ..., \phi_\theta(\mathbf{x}_m)\}$ from the encoder.
6: Adjust the penalty weights λ_l, λ_j^g and thresholds accordingly.
7: minimize $\frac{1}{m}(\sum_{i=1}^m \mathcal{L}(G(\phi_\theta(\mathbf{x}_i)), \mathbf{x}_i) + \lambda_l \psi^l(\phi_\theta(\mathbf{x}_i))) + \sum_{j=1}^M \lambda_j^g \psi_j^g(\{\phi_\theta(\mathbf{x}_k) | k \in S_j\})$ using any standard gradient-based optimization method and update the parameters θ.
8: **end for**

several randomly sampled overlapping Sudokus and optimize the All-Different constraints (global) as well as the cardinality constraints (local) within them. However, in Crystal-Structure Phase Mapping, the maximal connected component becomes too large to batch together, due to the constraints (*phase field connectivity* and *Gibbs-alloying rule*) concerning all data points in the composition graph. Thus, we instead only batch a subset (still a connected component) of the maximal connected component – e.g., a path in the composition graph, and optimize the objective function that only concerns constraints within the subset (along the path). By iteratively solving sampled local structures of the "large" maximal component, we cost-efficiently approximate the entire global constraint. For efficiency, DRNets solves all instances together using constraint-aware SGD (see Algorithm 1)

Moreover, for optimizing the overall objective, constraint-aware SGD dynamically adjusts the thresholds and the weights of constraints according to their

satisfiability, which can involve non-differentiable functions. Specifically, we initialize the penalty weights and thresholds of constraints for penalty functions using hyper-parameters. During optimization DRNets keep track of the satisfiability of constraints (this step could involve non-differentiable functions) and adjust the penalty weights and thresholds based on their satisfiability. For example, in crystal-structure phase mapping, we use the k-sparsity constraint to implement the Gibbs rule (the number of phases than can coexist is at most 3 in a ternary chemical system) and the Gibbs-Alloy rule (the max phase count decreases by 1 if "alloying" occurs) in DRNets. The threshold c of k-sparsity is initialized as $\log 3$ (Gibbs rule), which is the entropy when the probability mass is evenly distributed among 3 phases. During the optimization of DRNets, if "alloying" occurs (by checking the phase shifting ratio between adjacent data points), DRNets decreases the max phase count from 3 to 2 and adjust the threshold c from $\log 3$ to $\log 2$ (Gibbs-Alloy rule) accordingly. Moreover, note that even if we optimize the penalty function ($H(P_M)$) to be equal or smaller than $\log k$ (k is 2 or 3), it could be the case that there are more than k phases with positive probability mass (> 0.01) and their probability mass may not be evenly distributed, which means the k-sparsity is still not satisfied. Thus, DRNets keeps tracking the satisfiability of k-sparsity constraint: if the entropy is already below the current threshold (e.g., $\log k$) and there are still more than k phases with probability mass more than ϵ (0.01), we decrease the threshold c to keep enforcing the model to minimize the entropy until reaching the k-sparsity.

Computational details: All the experiments are performed on one NVIDIA Tesla V100 GPU with 16GB memory. For the training process of our DRNets, we select a learning rate in $\{0.0001, 0.0005, 0.001\}$ with Adam optimizer [74], for all the experiments. For baseline models, we followed their original configurations

and further fine-tuned their hyper-parameters to saturate their performance on our tasks.

3.6 Discussion

A central tenet of the scientific process is the interpretation of data in the context of a rich body of scientific knowledge. However, the efficient integration of complex prior knowledge into machine learning approaches has been an open challenge in AI. DRNets provides a general modular framework for incorporating prior knowledge that can be customized to effectively tackle unsupervised pattern demixing tasks. Herein we provide an in-depth description of the application of DRNets to a scientific application, crystal-structure phase mapping, a fundamental long-standing challenge in materials science. We also illustrate the application of DRNets to a Sudoku demixing task, which is facilitated by the availability of benchmark data for algorithm comparisons and ablations studies.

In the DRNets framework, an *interpretable* structured latent space is crucial for the incorporation of background knowledge (see Fig. 3.2 and Fig. 3.12). *The semantics of the latent space emerges during the optimization process as the result of the interplay among the encoder, the generative decoder, the reasoning module, and (unlabeled) data.* We showed the effectiveness of our approach, such as for crystal-phase mapping, where DRNets significantly outperformed previous approaches and solved previously unsolved complex chemical systems, aiding the discovery of solar-fuels materials. Intuitively, the strong prior knowledge injected via the fixed generative decoder and reasoning constraints, combined with the data, constrain the optimization process to enforce the intended semantics of the latent space. As our Sudoku ablation studies show, if we either replace the fixed

cGAN with a (weaker) standard learnable decoder, without prior knowledge about single digits, or remove the reasoning modules, the optimization process can no longer find the right semantics for the latent space, and the digit and Sudoku accuracy deteriorate significantly. Data are critical, as they ultimately determine the semantics of the latent space, and DRNets clearly benefits from learning across multiple (unlabeled) instances. Nevertheless, somewhat surprisingly, the amount of unlabeled data required in DRNets for both the Sudoku and crystal-structure phase mapping applications is considerably more modest than in standard supervised deep learning settings.

The DRNets framework is general and modular and can be adapted to other de-mixing and unsupervised tasks for which domain rules and pattern prototypes or generative models are available. In addition, prior knowledge can also increase interpretability and boost performance of supervised models, another research directions for DRNets. In fact, some of the ideas in DRNets were inspired by our work in the context of supervised learning on multi-label classification and in particular deep-learning-based joint species distribution models (JSDMs)[18, 19] that integrate prior ecological knowledge and species observational training data from the eBird citizen science program. [139] The need to capture and interpret interactions between species and their local environments as well as interactions among different species, which are core questions in ecology and conservation, [142] was the initial motivation and inspiration for the semantically meaningful structured latent-space in the DRNets framework.[19] We anticipate that these concepts will extend to other tasks, for example, materials science tasks beyond phase mapping, such as materials property prediction.[76] The construction of a latent space with intended semantics can not only enable constraint reasoning, as demonstrated in the present work, but also facilitate

transfer learning wherein relationships among composition, structure, and some types of properties can be used to learn relationships for other types of properties. Such concepts extend the purview of prior knowledge from the rules and prototypes employed in the present work to rules, surrogate models, etc. from ancillary domains. More generally, research on incorporating neural network-based learning with symbolic knowledge representation and logical reasoning is an important next frontier in AI/ML research. [26] The DRNets framework represents a step in that direction, providing a modular end-to-end framework that can be customized for a range of tasks that require the combination of learning and reasoning, which are pervasive across a variety of application domains.

3.7 Supplementary Information

3.7.1 Ablation studies for Multi-MNIST-Sudoku

We performed ablation studies of DRNets on the Multi-MNIST-Sudoku task to demonstrate the importance of the reasoning module and the generative decoder. As shown in the Fig. 3.5, if we remove the reasoning modules, both the Sudoku accuracy and the digit accuracy would drop significantly, which shows that not only does the reasoning module help the Sudoku accuracy, it also improves the digit accuracy by eliminating impossible digits based on Sudoku rules. This effect is especially significant when it comes to the 9x9 case. In the 9x9 case, we used hand-written letters A-I for the second Sudoku in each overlapping Sudokus, which considerably increase the difficulty of recognizing the digit and the letter purely based on the reconstruction loss. Therefore, once we removed

the reasoning module, the Sudoku accuracy dropped from 99.2% to 0% and the digit accuracy also dropped from 99.9% to 63.8%. On the other hand, if we replace the generative decoder (cGAN) with a (weaker) standard learnable decoder, without prior knowledge about single digits, we can no longer connect the semantics of our latent space to the output of the decoder. Therefore, the optimization process can no longer find the right semantics for the latent space, even though the discovered nonsensical digits still follow the Sudoku rules (1.24% digit accuracy, 0% Sudoku accuracy).

To demonstrate the importance of the data-driven learning by optimizing the model over multiple instances using shared parameters, we performed ablation studies of the "down-scalability" of DRNets, i.e., when optimized using only a few data instances (Fig. 3.3). We evaluated the performance of DRNets with different dataset scales ranging from only one instance to 10,000 instances. The study shows the importance of learning the shared parameters across instances. Nevertheless, DRNets can reach 99% accuracy with only 100 (unlabeled) 9x9 Multi-MNIST-Sudoku instances, a considerable smaller amount of data compared to standard deep learning approaches. Since optimizing DRNets on a few instances may result in getting stuck at some local minimals such that the constraints are not all satisfied, we applied a restart mechanism [44] on Multi-MNIST-Sudoku to circumvent this issue. Specifically, since DRNets directly incorporate logical constraints, we can check whether those constraints are satisfied or not at the end of a run. If not, for instances with violated constraints, we re-run the algorithm again on them. In this ablation study, we applied at most 3 "restart" runs for each datasets to get the reported results on Fig. 3.3. Due to the restart mechanism, the performance of DRNets on 10,000 instances are slightly better than the one without restart (Fig. 3.5). Moreover, for the datasets

with small scales (¡1000), we performed multiple runs with randomly sampled instances to obtain an averaged performance.

3.7.2 Experimental details for Multi-MNIST-Sudoku

Data description: For Multi-MNIST-Sudoku, we generated $n^2 \times 10,000$ (n is 4 or 9) input data points for each training set, validation set and test set, where each data point corresponds to a 32x32 image of overlapping digits/letters from MNIST [86] and EMNIST [24] and every batch of n^2 data points forms a n-by-n overlapping Sudokus. For the 9x9 case, to distinguish the two overlapping Sudokus, we used letters A-I from EMNIST for the second Sudoku. For ease of presentation, we may still refer to letters as "digits" in the following content. We generated an extra *cGAN* dataset, which is composed of 25,000 *original MNIST/EMNIST images* for training the conditional GAN. Note that these four datasets are generated using disjoint sets of digit images.

DRNets for Multi-MNIST-Sudoku: For Multi-MNIST-Sudoku, the encoder is made of two ResNet-18 models [53] adapted from the PyTorch source code. The output layer for the first network has $2n$ (n is 4 or 9) dimensions, which models the two distributions P_i and Q_i for the two overlapping digits. Another network outputs $2n$ 100-dimensional ($200n$ dimensions in total) latent encoding $\mathbf{z}_{i,j}$ to encode the shape of the possible $2n$ digits conditioned on the input mixture, and is used by the generative decoder to generate the reconstructed digits. We use a conditional GAN [104], as the generative model in our generative decoder (for ease of presentation, we abuse a bit the notation of $G(\cdot)$ to also denote the generative model), which is adopted from the implementation of [88] and pre-

trained using digits in the *cGAN* dataset. Note that this is the only supervision we have in this task, which is even weaker than the general concept of the *weakly-supervised setting* [165]. Given the *shape* ($\mathbf{z}_{i,j}$) and *probability* embeddings (P_i and Q_i), DRNets estimate the two digits in the cell by computing the expected digits over P_i and Q_i, i.e., $\sum_{j=1}^{n} P_{i,j} G(\mathbf{z}_{i,j})$ and $\sum_{j=1}^{n} Q_{i,j} G(\mathbf{z}_{i,j+4})$, and remix them to reconstruct the original input mixture (Fig. 3.1g-i and Fig. 3.2a). We use entropy-based functions to impose the continuous relaxation of the cardinality and All-Different constraints to reason about the Sudoku structure, which results in a total of $2n^2 + 6n$ constraints for the nxn Sudoku ($n^2 \times 2$ cardinality constraints enforcing a single digit per cell; $n \times 3 \times 2$ all-different constraints enforcing no repetition of a digit in a row, column, and box (see Fig. 3.13b).

Continuous Relaxations for Multi-MNIST-Sudoku: In Multi-MNIST-Sudoku, there are two types of constraints in the Sudoku rules: (1) *cardinality constraints* and (2) *All-different constraints*.

Cardinality Constraints: One straightforward encoding of the discrete version of cardinality constraints, which are used to encode the possible two digits in the cell indicated by data point x_i (one from {1...4} and the other from {5...8}), is to use 8 binary variables $e_{i,j} \in \{0, 1\}$, while requiring $\sum_{j=1}^{4} e_{i,j} = 1$ and $\sum_{j=5}^{8} e_{i,j} = 1$. To relax the discrete variables, DRNets model probability distributions P_i and Q_i over digits 1 to 4 and 5 to 8 respectively: $P_{i,j}, _{j=1...4}$ and $Q_{i,j}, _{j=1...4}$ denote the probability of digit j and the probability of digit $j + 4$, respectively. Then, we can approximate the cardinality constraint of $e_{i,j}$ by minimizing the entropy of P_i and Q_i, which encourages P_i and Q_i to collapse to one value. One can see, when the entropy of distributions P_i and Q_i reaches 0, all the probability mass collapses to only one variable. Therefore, all $P_{i,j}$ and $Q_{i,j}$ are either 0 or 1, which is a valid

solution of the original discrete constraints.

All-different Constraints: Another combinatorial constraint in Multi-MNIST-Sudoku is the All-Different constraint, where all the cells in a *constrained set* S, i.e., each row, column, and any of four 2x2 boxes involving the corner cells, must be filled with non-repeating digits. For a probabilistic relaxation of the All-Different constraint, we analogously define the entropy of the averaged digit distribution for all cells in a constrained set S, i.e., $H(\bar{P}_S)$:

$$H(\bar{P}_S) = -\sum_{j=1}^{4} \bar{P}_{S,j} \log \bar{P}_{S,j} = -\sum_{j=1}^{4} \Big(\frac{1}{|S|}\sum_{i\in S} P_{i,j}\Big) \log \Big(\frac{1}{|S|}\sum_{i\in S} P_{i,j}\Big) \tag{3.4}$$

In this equation, a larger value implies that the digits in the cells of S distribute more uniformly. Thus, we can analogously approximate All-Different constraints by maximizing $H(\bar{P}_S)$ and $H(\bar{Q}_S)$ while minimizing all $H(P_i)$ and $H(Q_i)$, $i \in S$ (for the cardinality constraints of cell-i). When the entropy of the digit distribution in each cell is 0, we know that the digit distribution of each cell converges to one digit. Hence, if $H(\bar{P}_S)$ reaches its maximum, i.e., $\log|S|$, we have $\frac{1}{|S|}\sum_{i\in S} P_{i,j} = \frac{1}{|S|}$ for all digit j. Crossed with the fact that $P_{i,j}$ are either 0 or 1 when the cardinality constraints are satisfied, we know that only one $P_{i,j}$ is equal to 1 for all cell i in the set S and others are 0, which directly states the All-Different constraints.

Baselines for Multi-MNIST-Sudoku: We compared DRNets with the state-of-the-art for demixing digits, which are supervised methods: CapsuleNet [123] and ResNet [53]. Though ResNet and CapsuleNet have access to labeled data, they do not utilize Sudoku rules. Therefore, to saturate their performance, we further imposed Sudoku rules into those baselines via a post-process. Specifically, for 4x4 cases, we did a local search for the top-2 (top-3 would take too long to search) most likely choice of digits for each Sudoku of the two overlapping Sudokus and try to satisfy Sudoku rules with minimal modification compared

with the original prediction. Since there are only 288 different 4x4 Sudokus, we also performed an exhaustive search, which explicitly considers all the possible Sudoku configurations and selects the most likely one. However, such a strategy is out of the question for 9-by-9 instances, given that there are around 6.67×10^{21} possible 9x9 Sudokus. For 9x9 cases, we conduct a different local search algorithm since enumerating the top-2 digits for an 9x9 Sudoku is not feasible (2^{81}). We conducted a depth-first search for digits in each cell from most likely to less likely until it finds a valid Sudoku.

Because CapsuleNet [123] did not provide a source code, we adopted the implementation of Laodar [81], with minor modifications. For ResNet, we adopted the ResNet-18 architecture [53] and trained it in a multi-class classification setting, where we provide explicit supervision for the labels of two digits (one from $\{1..4\}$ and the other from $\{5..8\}$) in each cell. We did a grid search for the hyper-parameters of both baseline models to saturate their performance. Fig. 3.5 shows the comparison of the performance of different methods for Multi-MNIST-Sudoku showing how DRNets outperforms other approaches and how it even has the capbility of self-learning by reasoning about the Sudoku rules.

For the training of DRNets, we used $L1$ loss as the reconstruction loss between the reconstructed mixture and the original input. For the initial weights of constraints, we set 0.01 for the cardinality constraints, 1.0 for the All-Different constraints, and 0.001 for the $L1$ loss. Note that, DRNets are really "self-supervised" [69] by the Sudoku rules and the self-reconstruction, instead of the standard supervision by labeled data. Therefore, we can directly use DRNets to solve the test set without using a training set. We optimize DRNets on the test set for 100 epochs with a batch size of 100, and it took 50 minutes to finish and achieve the

reported performance for the 10,000 overlapping Sudokus.

3.7.3 Experimental details for Crystal Structure Phase Mapping

We illustrate the DRNets for crystal structure phase mapping for two chemical systems: (1) a ternary **Al-Li-Fe** oxide system, described in Le Bras et. al. [83], which is synthetically generated from a known phase diagram, which also provides the ground-truth solution, and (2) an experimental system from a continuous composition spread thin film from the ternary **Bi-Cu-V** oxide system.

For each system, the input data points are XRD patterns, each containing signals from 1 to 3 phases, as well as the composition of each system. Each XRD pattern is the (nonnegative) XRD scattering intensity as a function of the scattering vector magnitude, Q. A peak in the XRD pattern results from Bragg scattering and indicates the presence of a plane of atoms in the crystal structure with interplanar spacing of $d = 2\pi/Q$. Alloying refers to chemical substitution within a crystal structure where a change in composition causes the crystal structure to stretch or contract, so an expansion by 1% would cause the d values of the peaks to "shift" multiplicatively by 1%, corresponding to 1% decrease in the Q value of each peak. Alloying can occur with negligible expansion/contraction of the crystal structure, in which case it cannot be directly detected by measured peak positions. Complex alloying may occur where alterations to the aspect ratio of the crystal structure causes nonuniform peak shifting, which can be modelled by DRNets but was not in the present work. A phase diagram comprises a graph of the regions of composition space where each combination of phases is observed (the phase fields), as well as annotation of any alloying that occurs

within each region.[131] DRNets provide all phase fields within the hull of input XRD pattern compositions, as well as peak shifting for each phase that captures most instances of alloying.

Ternary compositions are plotted in a standard 2-D Euclidean triangle plot and Delaunay triangulation provides edges representing neighboring composition points, which is used to establish the mathematical expression of the thermodynamic rules. The mathematical descriptions of our implementations of these rules are described below. The rules result from considering free energy thermodynamics in the context of a composition space with pressure and temperature held constant, as they were in the synthesis of the materials. The names and brief description of the rules are as follows: "Gibbs" is based on the standards Gibbs' Phase Rule and limits the number of phases that can coexist; "Gibbs-Alloy" is an extension of the same thermodynamic rule where the thermodynamic degree of freedom assumed by alloying lowers the max phase count; "Phase-Field-Connectivity" is the composition graph implementation of the definition of a phase field in a phase diagram, i.e. that each phase field comprises a continuous region of composition space. These thermodynamic rules are central to phase diagram determination[131, 31] and have been implemented in different ways in our previous work,[43] and incorporated to some extent in Refs [135] and [79] (e.g. with sparsity constraints and clustering that lowers the number of activated phases).

For the Bi-Cu-V oxide system, the thin film materials were prepared by sputter co-deposition from Bi, Cu, and V sources. The positions of the sources form an equilateral triangle with a substrate above the center on the triangle collecting atoms at different rates at each position, as described previously [141]. At each

substrate location the unique mixture or composition of Bi, Cu and V atoms are mixed at the atomic level and crystallize into crystal domains generally between 5 and 100 nm upon subsequent annealing (550 °C in air in this case), forming a thin film approximately 200 nm in thickness. For a given 1 mm^2 area representing a given composition, the mixture of order 10^{10} crystal domains comprising the 1 to 3 different phases are characterized through synchrotron x-ray diffraction[49] to generate the XRD pattern for that composition.

There are 307 composition data points for Bi-Cu-V system and each XRD pattern contains D = 300 diffraction signals equally spaced from Q=5 to 45 nm^{-1}. There are 231 composition data points for the benchmark Al-Li-Fe system and each XRD pattern contains D = 650 diffraction signals equally spaced from Q=15 to 80 nm^{-1}. Each XRD pattern is normalized to a maximum intensity of 1. The set of prototype stick patterns for each system correspond to known crystal structures from the International Centre for Diffraction Data (ICDD) database.

DRNets for Phase-Mapping: Given the input D-dimensional XRD pattern, we use four 3-layer-fully-connected networks as our encoder to encode a two-part latent space, which captures the probabilities (denoted as $P_{i,j}$) and shapes of the possible phases (denoted as $\mathbf{z}_{i,j}$) and is constrained by the reasoning module to satisfy the thermodynamic rules (Fig. 3.2). To model more realistic conditions, the generative decoder of the DRNets uses Gaussian mixture models [89] to approximate the measured phase patterns where the relative peak locations and intensities are given by the prototype stick pattern, and the latent encoding $\mathbf{z}_{i,j}$ parameterizes the other information needed to simulate an XRD pattern with a series of Gaussian peaks: the peak width, multiplicative shift of peak locations, and possible amplitude variance. The set of phase activations sum

to 1 and are the relative intensities of the set of generated phase-pure patterns whose sum is the reconstruction of the input XRD pattern. To remove negligible activations that are mainly caused by experimental noise we applied a simple post-processing that cuts-off all the activations that are lower than 1.0%. The output of the first three networks in the encoder are M-dimensional vectors: $(P_{i,1}, ..., P_{i,M})$, $(\alpha_{i,1}, ..., \alpha_{i,M})$, $(\sigma_{i,1}, ..., \sigma_{i,M})$ (M is the number of possible phases, e.g., 159 for the Al-Li-Fe oxide system), which represent the probability $P_{i,j}$ of each phase-j at data point i, the multiplicative shifting ratio $\alpha_{i,j}$, and the standard deviation $\sigma_{i,j}$ of the Gaussians characterizing the peaks in phase-j, respectively. The output of the last network is a $M \times K$-dimensional vector, representing the possible amplitude variance of peaks in each phase. Here, K is the number of maximal peaks in a stick pattern ($K = 200$). For the first 3 networks there are 1024, 1024, and 512 hidden units, respectively per layer, and for the last network there are 512, 512, and 32 hidden units, respectively per layer. The last layer has fewer hidden units given its high-dimensional output space ($M \times K$). All networks use ReLU [107] as their activation function.

The thermodynamic rules can concern many to all points in the composition graph, creating a challenging set of combinatorial constraints to impose. In Multi-MNIST-Sudoku, where each overlapping Sudoku naturally forms the maximal connected components in the *constraint graph*, we can easily batch every n^2 data points together to reason about the All-Different constraints among them. However, in Crystal-Structure-Phase-Mapping, since the maximal connected component involves all data points in the composition graph, neither batching all data points into the memory nor reasoning about the whole graph is tractable. Therefore, to enforce the connectivity constraint we devised a strategy of sampling the large connected component through many local structures (still

connected components) and solve each of them iteratively. Specifically, for each oxide system, we sampled 100,000 paths in the composition graph via Breadth First Search to construct a path pool. Then, for every iteration, DRNets randomly sample a path from the pool and batch the data points along that path (see Fig. 3.6b). Finally, we only reason about the thermodynamic rules along the path. By iteratively solving sampled local structures (paths) of the ”large” maximal component, we can cost-efficiently approximate all global constraints.

Continuous Relaxations for Crystal-Structure Phase Mapping: The thermodynamic rules imposed in DRNets are based on standard properties of isothermal compositional diagrams[131, 31]. The only constraint applied to each XRD pattern independently is the maximum of 3 phases. This is a consequence of Gibbs' phase rule where the 2 composition degrees of freedom are the thermodynamic variables corresponding to a maximum phase count of 3 at any point within the ternary composition space. An extension of this rule is that alloying removes at least 1 thermodynamic degree of freedom and thus lowers the maximum phase count to 2. Alloying is detected through comparison of the multiplicative shifting parameters (in the DRNets latent space), between neighboring compositions, where a shifted pattern between 2 neighbors that share the same set of phases invokes the Gibbs-Alloy rule where both patterns may contain only 2 phases. The final rule relates to the compositional connectivity of each phase fields, i.e. Phase-Field-Connectivity. Thermodynamic compositional phase diagrams comprise the convex hull of the Gibbs free energy for all phases as a function of composition. The generally parabolic shape of the free energy for each phase implies that if a phase or set of phases appears on the convex hull at one composition, every other composition where it appears can be accessed by a contiguous composition path. Further the amount of the phase, i.e. its activation in the XRD

patterns, will vary smoothly in composition space. Importantly, the thin film synthesis of the Bi-Cu-V system can result in non-equilibrium phase behavior. In our experience, from inspection of thousands of XRD patterns of thin films, any deviation from equilibrium typically does not alter the rules as enforced, likely because the alterations correspond to a kinetically impeded phase that is removed from "consideration" in the free energy diagram, or alteration of the effective free energy for a phase, both of which can change which phases are experimentally observed but not the enforced rules. Indeed this is a key reason why phase mapping of experimental systems is required, as opposed to relying on computed free energy diagrams.

Implementation of thermodynamic rules for Crystal-Structure Phase Mapping: ***Gibbs:*** This rule states the maximum number of co-existing phases, which is imposed via the relaxation of the k-sparsity constraints. For the discrete version, we can model the existence of M possible phases of i-th data point using binary variables $e_{i,j}$ ($j = 1...M$), while requiring $\sum_{j=1}^{M} e_{i,j} \leq k$. We derive the relaxation of k-sparsity constraints in a similar way as the cardinality constraints except that we now want to force the distribution to concentrate on at most k entities (phases). By normalizing the values of discrete variables $e_{i,j}$ ($j = 1...M$) to a discrete distribution P_M, we can minimize the entropy of distribution P_M to at most $\log k$, which is the maximal entropy when the distribution concentrates on only k values. Though, $H(P_M) < \log k$ is not a sufficient condition for k-sparsity, we can initialize the threshold c of k-sparsity constraints to $\log k$ and dynamically adjust the value of c based on the satisfaction of the k-sparsity constraints. In practice, it works well with the supervision from other modules, such as the self-reconstruction.

Gibbs-Alloy: DRNets explicitly model the shifting ratio in the generative decoder and penalize the difference between adjacent data points along our sampled path. The reasoning module keeps track of the difference of shifting ratio between adjacent data points, and when it is larger than a threshold (0.001), we confirm the existence of "alloying" and reduce the maximum number of possible co-existing phases by one via adjusting the threshold c in the k-Sparsity Constraints.

Phase-Field-Connectivity: We impose this rule by penalizing the difference between the phase activation of adjacent data points P_u and P_v along the sampled path. In our implementation, we used L2-norm to penalize the difference.

Evaluation Criteria for Phase Mapping: Our evaluation criteria (see Fig. 3.10a) include reconstruction losses, phase fidelity loss and the satisfaction of thermodynamic rules. Note that, before evaluating the reconstruction losses, we fit the demixed patterns (for all methods) to the closest prototype using a model described previously[83] to exclude noise. We quantified the phase fidelity loss by measuring the Jensen-Shannon distance (JS distance) between the demixed XRD pattern and the closest pattern simulated from a prototype. The motivation for using the JS distance metric for fidelity of demixed patterns is that the set of peaks and their locations are the most important characteristics of a phase pattern. We normalize the area under each XRD pattern to be 1 and use the JS distance metric (with ϵ of 1e-9 to avoid division by zero) to quantify the difference between two patterns, which has a large loss when peaks appear in one pattern but not the other.

Manual Solution Evaluation for Phase Mapping: The detailed explanation of the main text statement "the presence of each phase was manually verified by matching each prototype peak to a signal in a corresponding XRD pattern and by

confirming that measured XRD peaks in the pattern are explained by the DRNets solution" is as follows: for powder XRD patterns, all peaks (with intensity above the detectability limit) in the prototype must be in the measured pattern, and all peaks in the XRD pattern must be explained by prototypes. Failure of the former indicates that one is considering a prototype of the wrong crystal structure, and failure of the latter could be due to the wrong prototype or an undetected phase, so this analysis can only be done once all phases are identified. This is precisely why complex phase mapping in high dimensional composition spaces is intractable for humans because the pure-phase XRD patterns are not known and the existence of peaks from multiple phases creates many different possible choice of prototype(s) that explain some to all of the measured peaks, especially under consideration of possible peak shifting of the prototypes. In many cases, a single XRD pattern can only be definitively solved by reasoning about XRD patterns of related compositions, so finding a logically consistent solution is very difficult.

Phase mapping problems are particularly challenging when the XRD data does not contain examples of pure phase patterns, as is often the case in combinatorial materials research. In the Bi-Cu-V DRNets solution, only 2 of the 13 phases appear with activations above 80%, demonstrating the ability of DRNets to identify prototypes that only appear in combination with other prototypes.

In phase mapping, we also note the possibility of the presence of a phase in the input XRD patterns that is entirely distinct (not just modified) from all prototypes. We don't discuss handling such cases since it is beyond the scope of this work, requiring density function theory analysis, although our results on reconstruction loss (DRNets produces much better data reconstruction than other methods)

demonstrate that the presence of a new phase would be readily identifiable by an uncharacteristically high reconstruction loss in a specific composition region.

Baselines for Crystal-Structure Phase Mapping: We compared DRNets with the state-of-the-art phase mapping algorithms, IAFD [43] and NMF-k [135], which are both non-negative matrix factorization (NMF) based unsupervised demixing models. NMF-k improves the pure NMF algorithm [91] by clustering common phase patterns from thousands of runs. However, NMF-k does not enforce thermodynamic rules and therefore the solutions produced are often not completely physically meaningful. IAFD uses external mixed-integer programming modules to enforce thermodynamic rules during the demixing. However, due to the gap between the external optimizer and NMF module, the solution of IAFD is still far from the ground truth.

Our evaluation criteria (see Fig. 3.10a) include reconstruction losses, phase fidelity loss and the satisfaction of thermodynamic rules. Note that, before evaluating the reconstruction losses, we fit the demixed phases (for all methods) to the closest ideal phases using the physical model [83] to exclude noise. Meanwhile, we quantified the phase fidelity loss by measuring the Jensen-Shannon distance (JS distance) between the demixed phases and the closest ideal phases. The reason of using the JS distance to measure the fidelity is that the location of peaks are the most important characteristics of a phase pattern. We normalize the area under each XRD pattern to be 1 and use the JS distance metric (with ϵ of 1e-9 to avoid division by zero) to quantify the difference between two patterns, which has a large loss when peaks appear in one pattern but not the other.

We also considered a recently proposed supervised algorithm [87] in which a deep neural network, trained using simulated data based on known crystal phase

patterns, directly predicts the phases present in a given XRD pattern. While such an approach can be effective for complementing human expertise for a single XRD pattern, it performed poorly on a complex system such as the benchmark Al-Li-Fe oxide system (phase identification accuracy around 1%), which highlights the limitations of a purely simulated-based supervised approach for handling the combinatorics of phase mapping. Moreover, we note that the method can only provide a classification for the existence or the quantized fractions of phases instead of actually estimating the weights (regression task) of the phase activations, as done by DRNets. So, we evaluated DRNets using the setting of their classification tasks on their Li-Sr-Al powder system dataset. We used a similar setting of hyper-parameters as for the Bi-Cu-V oxide system. We note that this powder system only contains one phase combination (Li_2O, SrO, and Al_2O_3): both their model and DRNets achieved 100% phase recognition accuracy. For the quantized phase fraction (0-33%, 33%-66%, and 66%-100%) classification tasks, unsupervised DRNets have an accuracy of 95.3%, outperforming their supervised model with accuracy of 86.0%. Beyond this quantized phase activation classification, DRNets can further provide estimates of the weights of the phase activations with an average error of 4.7%.

In the optimization process of DRNets, we used the JS distance with a weight of 20.0 plus the L2-distance with a weight of 0.05 as the reconstruction loss. Due to the different noise level in the Al-Li-Fe oxide system and the Bi-Cu-V oxide system, we use different weights for the penalty functions of different constraints. For Al-Li-Fe oxide system, the weights of k-sparsity constraints (Gibbs rule) is 1.0 and the weights of phase field connectivity constraints is 0.01. For Bi-Cu-V oxide system, the weights of k-sparsity constraints (Gibbs rule) is 30.0 and the weights of phase field connectivity constraints is 3.0. In

terms of the optimization process, DRNets took about 30 minutes to achieve the reported performance for both systems. In contrast, not only do IAFD and NMF-k have a much worse performance with respect to the solution quality, they also take considerable longer, several hours, to generate their solutions. In fact, for the Bi-Cu-V oxide system, both NMF-k's solution and IAFD's solution are not physically meaningful.

Down-Scaling Analysis for Crystal-Structure Phase Mapping: We also investigated the "down-scalability" of DRNets for crystal structure phase mapping tasks, which further confirmed the importance of learning DRNets across multiple samples. In this experiment, we used *phase activation accuracy* as the evalution metric, which evaluates the percentage of sample points that have the same set of phases as the ground truth. For Al-Li-Fe oxide system, we evaluated the performance of DRNets learned over different number of XRD patterns ranging from one XRD patterns to the entire set. For the case of using K XRD patterns, we performed $\lfloor 231/K \rfloor$ runs of DRNets on the XRD patterns randomly sampled from the composition space to obtain an averaged performance. As shown in Fig. 3.10, learning over multiple XRD patterns plays an important role for DRNets to solve the Al-Li-Fe oxide system and DRNets can almost perfectly recover the phase activation of XRD patterns when it is learned over more than 150 XRD patterns. Since we do not have the ground truth of the Bi-Cu-V oxide system, we only evaluate the consistency between the results of running DRNets on all 307 XRD samples and the results of running DRNets on each XRD sample one at a time (named DRNet-single). We made minor edition to the DRNet to adapt to the setting of running on a single XRD pattern. For DRNet-single, we ignored the Gibbs-alloying rule and the phase-field-connectivity rule since they do not make sense on a single sample point. As a result, DRNet-single can

only identify the same phases as DRNets for 27% of the patterns, highlighting that the nuanced phase behavior of this system can only be resolved through combinatorial experimentation matched with learning the shared parameters across multiple XRD patterns combined with reasoning about the underlying complex thermodynamic constraints.

Background Subtraction: As a pre-processing step for the X-Ray patterns, we employ our multi-component background learning model.[2]
The code is available at https://github.com/gomes-lab/BackgroundSubtraction.jl.

CHAPTER 4

OTHER RELATED WORKS

In this chapter, I introduce two of my related research works that are interweaved with the main story. In the first section, we study the covariate shift problem in the citizen science project, such as the eBird program [139], to reduce the data bias caused by the volunteers' personal preferences. With this method, we can mitigate the prediction bias caused by the non-uniform data distribution and learn a prediction model that is more aligned with the scientific objectives. In the second section, we study the task-based learning problem, where we integrate the task-based non-differentiable objective function, which often comes from hand-crafted rules or prior knowledge, into an end-to-end deep learning framework, to optimize a model directly towards the ultimate goal. Applied to two financial prediction problems, revenue surprise forecasting and credit risk modeling, our model boosted the profit by 20%, compared with industry benchmarks, in backtesting.

4.1 Bias Reduction for Citizen Science via End-to-End Shift Learning

4.1.1 Introduction

Citizen science projects [139, 82, 126] play a critical role in collecting rich datasets for scientific research, especially in computational sustainability [42], because they offer an effective, low-cost way to collect large datasets for non-commercial

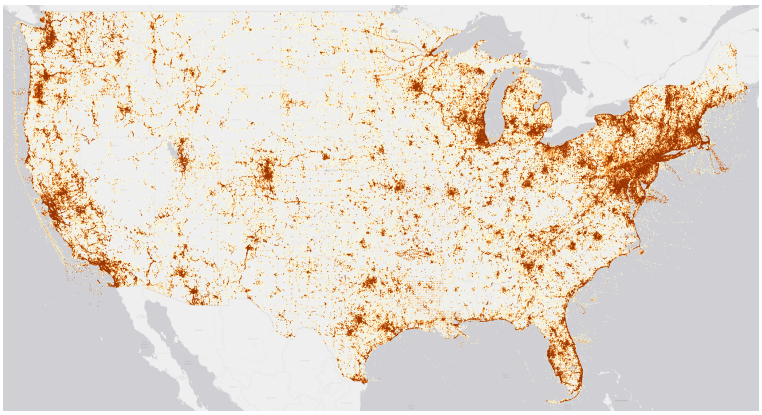


Figure 4.1: **Highly biased distribution of *eBird* observations in the continental U.S.** Submissions are concentrated in or near urban areas and along major roads.

research. The success of these projects depends heavily on the public's intrinsic motivations as well as the enjoyment of the participants, which engages them to volunteer their efforts [12]. Therefore, citizen science projects usually have few restrictions, providing as much freedom as possible to engage volunteers, so that they can decide where, when, and how to collect data, based on their interests. As a result, the data collected by volunteers are often biased, and align more with their personal preferences, instead of providing systematic observations across various experimental settings. For example, personal convenience has a significant impact on the data collection process, since the participants contribute their time and effort voluntarily. Consequently, most data are collected in or near urban areas and along major roads. On the other hand, most machine learning algorithms are constructed under the assumption that the training data are governed by the same data distribution as that on which the model will later be tested. As a result, the model trained with biased data could perform poorly when it is evaluated with unbiased test data designed for the scientific objectives.

In order to improve the scientific quality of the data, mechanisms to shift

the efforts of volunteers into the more unexplored areas have been proposed [158]. However, the scalability of those mechanisms is limited by the budget, and it takes a long time to realize the payback. Furthermore, the type of locality also restricts the distribution of collected data. For example, it is difficult to incentivize volunteers to go to remote places, such as deserts or primal forests, to collect data. Therefore, a tactical learning scheme is needed to bridge the gap between biased data and the desired scientific objectives.

In general, given only the labeled training data (collected by volunteers) and the unlabeled test data (designed for evaluating the scientific objectives), we set out to (i) learn the shift between the training data distribution P (associated with PDF $p(\mathbf{x}, y)$) and the test data distribution Q (associated with PDF $q(\mathbf{x}, y)$), and (ii) to compensate for that shift so that the model will perform well on the test data. To achieve these objectives, we needed to make assumptions on how to bring the training distribution into alignment with the test distribution. Two candidates are *covariate shift* [10], where $p(y|\mathbf{x}) = q(y|\mathbf{x})$, and *label shift* [90], where $p(\mathbf{x}|y) = q(\mathbf{x}|y)$. Motivated by the field observations in the *eBird* project, where the habitat preference $p(y|\mathbf{x})$ of a given species remains the same throughout a season, while the occurrence records $p(\mathbf{x}|y)$ vary significantly because of the volunteers' preferences, we focused on the *covariate shift* setting. Informally, covariate shift captures the change in the distribution of the feature (covariate) vector $\mathbf{x}$. Formally, under covariate shift, we can factor the distributions as follows:

$$
\begin{aligned}
p(\mathbf{x}, y) &= p(y|\mathbf{x})p(\mathbf{x}) \\
q(\mathbf{x}, y) &= q(y|\mathbf{x})q(\mathbf{x}) \\
p(y|\mathbf{x}) = q(y|\mathbf{x}) \Longrightarrow \frac{q(\mathbf{x}, y)}{p(\mathbf{x}, y)} &= \frac{q(\mathbf{x})}{p(\mathbf{x})}
\end{aligned} \tag{4.1}
$$

Thus we were able to learn the shift from P to Q and correct our model by quantifying the test-to-training **shift factor** $q(\mathbf{x})/p(\mathbf{x})$.

Our contribution is an end-to-end learning scheme, which we call the Shift Compensation Network (SCN), that estimates the shift factor while re-weighting the training data to correct the model. Specifically, SCN (i) estimates the shift factor by learning a discriminator that distinguishes between the samples drawn from the training distribution and those drawn from the test distribution, and (ii) it aligns the mean of the weighted feature space of the training data with the feature space of the test data, which guides the discriminator to improve the quality of the shift factor. Given the shift factor learned from the discriminator, SCN also compensates for the shift by re-weighting the training samples obtained in the training process in order to optimize classification loss under the test distribution. We worked with data from *eBird* [139], which is the world's largest biodiversity-related citizen science project. Applying SCN to the *eBird* observational data, we demonstrate that it significantly improves multi-species distribution modeling by detecting and correcting for the data bias, thereby providing a better approach for monitoring species distribution as well as for inferring migration changes and global climate fluctuation. We further demonstrate the advantage of combining the power of discriminative learning and feature space matching, by showing that SCN outperforms all competing models in our experiments.

4.1.2 Preliminaries

Notation

We use $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y \in \mathcal{Y} = \{0, 1, ..., l\}$ for the feature and label variables. For ease of notation, we use P and Q for the training data and test data distributions, respectively, defined on $\mathcal{X} \times \mathcal{Y}$. We use $p(\mathbf{x}, y)$ and $q(\mathbf{x}, y)$ for the probability density functions (PDFs) associated with P and Q, respectively, and $p(\mathbf{x})$ and $q(\mathbf{x})$ for the marginal PDFs of P and Q.

Problem Setting

We have labeled training data $D_P = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) ...,(\mathbf{x}_n, y_n)\}$ drawn iid from a training distribution P and unlabeled test data $D_Q = \{\mathbf{x}'_1; \mathbf{x}'_2; ...; \mathbf{x}'_n\}$ drawn iid from a test distribution Q, where P denotes the data collected by volunteers and Q denotes the data designed for evaluation of the scientific objectives. Our goal is to yield good predictions for samples drawn from Q. Furthermore, we make the following (realistic) assumptions:

- $p(y|\mathbf{x}) = q(y|\mathbf{x})$
- $p(\mathbf{x}) > 0$ for every $\mathbf{x} \in \mathcal{X}$ with $q(\mathbf{x}) > 0$.

The first assumption expresses the use of *covariate shift*, which is consistent with the field observations in the *eBird* project. The second assumption ensures that the support of P contains the support of Q; without this assumption, this task would not be feasible, as there would be a lack of information on some samples $\mathbf{x}$.

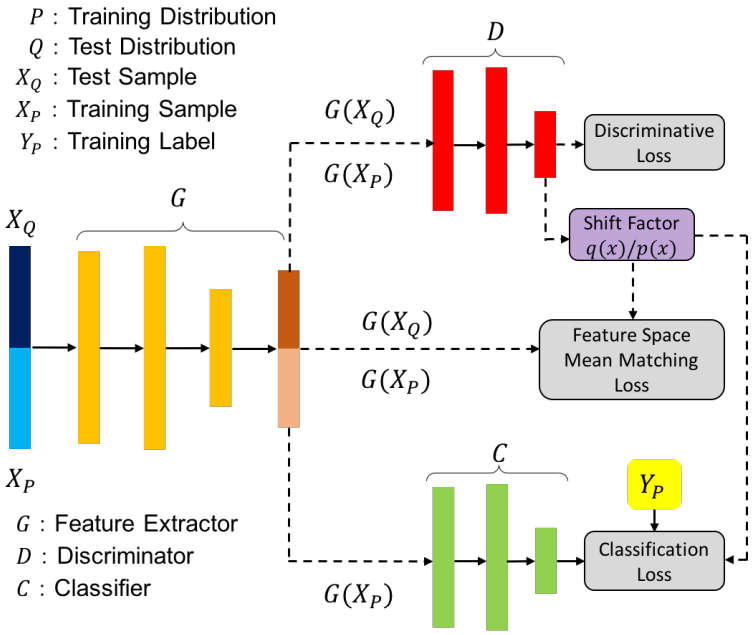


Figure 4.2: Overview of the Shift Compensation Network

As illustrated in [128], in the covariate shift setting the loss $\ell(f(\mathbf{x}), y)$ on the test distribution Q can be minimized by re-weighting the loss on the training distribution P with the shift factor $q(\mathbf{x})/p(\mathbf{x})$, that is,

$$\mathbb{E}_{(\mathbf{x},y)\sim Q}[\ell(f(\mathbf{x}), y)] = \mathbb{E}_{(\mathbf{x},y)\sim P}\left[\ell(f(\mathbf{x}), y)\frac{q(\mathbf{x})}{p(\mathbf{x})}\right] \tag{4.2}$$

Therefore, our goal is to estimate the shift factor $q(\mathbf{x})/p(\mathbf{x})$ and correct the model so that it performs well on Q.

4.1.3 End-to-End Shift Learning

Shift Compensation Network

Fig. 4.2 depicts the end-to-end learning framework implemented in the Shift Compensation Network (SCN). A feature extractor G is first applied to encode both the raw training features X_P and the raw test features X_Q into a high-level

feature space. Later, we introduce three different losses to estimate the shift factor $q(\mathbf{x})/p(\mathbf{x})$ and optimize the classification task.

We first introduce a discriminative network (with discriminator D), together with a discriminative loss, to distinguish between the samples coming from the training data and those coming from the test data. Specifically, the discriminator D is learned by maximizing the log-likelihood of distinguishing between samples from the training distribution and those from the test distribution, that is,

$$\begin{aligned}\mathcal{L}_D =& \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim P}[\log(D(G(\mathbf{x})))]+ \\ & \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim Q}[\log(1-D(G(\mathbf{x})))]\end{aligned} \tag{4.3}$$

Proposition 1 *For any fixed feature extractor G, the optimal discriminator D^* for maximizing $\mathcal{L}_D$ is*

$$D^*(G(\mathbf{x})) = \frac{p(\mathbf{x})}{p(\mathbf{x})+q(\mathbf{x})}.$$

Thus we can estimate the shift factor $\frac{q(\mathbf{x})}{p(\mathbf{x})}$ by $\frac{1-D(G(\mathbf{x}))}{D(G(\mathbf{x}))}$. Proof.

$$\begin{aligned}D^* &= \operatorname*{argmax}_D \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim P}[\log(D(G(\mathbf{x})))] + \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim Q}[\log(1-D(G(\mathbf{x})))] \\ &= \operatorname*{argmax}_D \int p(\mathbf{x})\log(D(G(\mathbf{x}))) + q(\mathbf{x})\log(1-D(G(\mathbf{x})))d\mathbf{x} \\ &\implies \textit{(maximizing the integrand)} \\ D^*(G(\mathbf{x})) &= \operatorname*{argmax}_{D(G(\mathbf{x}))} p(\mathbf{x})\log(D(G(\mathbf{x}))) + q(\mathbf{x})\log(1-D(G(\mathbf{x}))) \\ &\implies \textit{(the function } d \to p\log(d)+q\log(1-d) \textit{ achieves its} \\ &\qquad \textit{maximum in } (0,1) \textit{ at } \tfrac{p}{p+q}) \\ D^*(G(\mathbf{x})) &= \frac{p(\mathbf{x})}{p(\mathbf{x})+q(\mathbf{x})}\end{aligned}$$

Our use of a discriminative loss is inspired by the generative adversarial nets (GANs) [45], which have been applied to many areas. The fundamental idea of GANs is to train a generator and a discriminator in an adversarial way, where the generator is trying to generate data (e.g., an image) that are as similar to the source data as possible, to fool the discriminator, while the discriminator is trying to distinguish between the generated data and the source data. This idea has recently been used in domain adaptation [151, 57], where two generators are trained to align the source data and the target data into a common feature space so that the discriminator cannot distinguish them. In contrast to those applications, however, SCN does not have an adversarial training process, where the training and test samples share the same extractor G. In our setting, the training and test distributions share the same feature domain, and they differ only in the frequencies of the sample. Therefore, instead of trying to fool the discriminator, we want the discriminator to distinguish the training samples from the test samples to the greatest extent possible, so that we can infer the shift factor between the two distributions reversely as in Proposition 1.

Use of a feature space mean matching (FSMM) loss comes from the notion of kernel mean matching (KMM) [62, 50], in which the shift factor $w(\mathbf{x}) = \frac{q(\mathbf{x})}{p(\mathbf{x})}$ is estimated directly by matching the distributions P and Q in a reproducing kernel Hilbert space (RKHS) $\Phi_{\mathcal{H}} : \mathcal{X} \to \mathcal{F}$, that is,

$$\begin{aligned} &\underset{w}{\text{minimize}} \left\| \mathbb{E}_{\mathbf{x}\sim Q}[\Phi_{\mathcal{H}}(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})\Phi_{\mathcal{H}}(\mathbf{x})] \right\|_2 \\ &\text{subject to } w(\mathbf{x}) \geq 0 \text{ and } \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})] = 1 \end{aligned} \tag{4.4}$$

Though Gretton ([50]) proved that the optimization problem (4.4) is convex and has a unique global optimal solution $w(\mathbf{x}) = \frac{q(\mathbf{x})}{p(\mathbf{x})}$, the time complexity of KMM is cubic in the size of the training dataset, which is prohibitive when dealing with

very large datasets. We note that even though we do not use the universal kernel [136] in an RKHS, $w(\mathbf{x}) = q(\mathbf{x})/p(\mathbf{x})$ still implies that

$$\left\|\mathbb{E}_{\mathbf{x}\sim Q}[\Phi(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})\Phi(\mathbf{x})]\right\|_2 = 0$$

for any mapping $\Phi(\cdot)$. Therefore, our insight is to replace the $\Phi_{\mathcal{H}(\cdot)}$ with a deep feature extractor $G(\cdot)$ and derive the FSMM loss to further guide the discriminator and improve the quality of $w(\mathbf{x})$.

$$\begin{aligned}\mathcal{L}_{FSMM} &= \left\|\mathbb{E}_{\mathbf{x}\sim Q}[G(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})G(\mathbf{x})]\right\|_2 \\ &= \left\|\mathbb{E}_{\mathbf{x}\sim Q}[G(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}\left[\frac{1 - D(G(\mathbf{x}))}{D(G(\mathbf{x}))}G(\mathbf{x})\right]\right\|_2 \end{aligned} \tag{4.5}$$

One advantage of combining the power of $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$ is to prevent overfitting. Specifically, if we were learning the shift factor $w(\mathbf{x})$ by using only $\mathcal{L}_{FSMM}$, we could end up getting some ill-posed weights $w(\mathbf{x})$ which potentially would not even be relevant to $q(\mathbf{x})/p(\mathbf{x})$. This is because the dimensionality of $G(\mathbf{x})$ is usually smaller than the number of training data. Therefore, there could be infinitely many solutions of $w(\mathbf{x})$ that achieve zero loss if we consider minimizing $\mathcal{L}_{FSMM}$ as solving a linear equation. However, with the help of the discriminative loss, which constrains the solution space of equation (4.5), we are able to get good weights which minimize $\mathcal{L}_{FSMM}$ while preventing overfitting. On the other hand, the feature space mean matching loss also plays the role of a regularizer for the discriminative loss, to prevent the discriminator from distinguishing the two distributions by simply memorizing all the samples. Interestingly, in our experiments, we found that the feature space mean matching loss works well empirically even without the discriminative loss. A detailed comparison will be shown in the Experiments section.

Using the shift factor learned from $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$, we derive the weighted

Algorithm 2 Two-step learning in iteration t for SCN

Require: X_P and X_Q are raw features sampled iid from the training distribution P and the test distribution Q; Y_P is the label corresponding to X_P; M_P^{t-1} and M_Q^{t-1} are the moving averages from the previous iteration.

1: **Step 1**

$m_Q^t = \sum_{\mathbf{x}_i \in X_Q} G(\mathbf{x}_i)/|X_Q|$

$m_P^t = \sum_{\mathbf{x}_j \in X_P} \frac{1-D(G(\mathbf{x}_j))}{D(G(\mathbf{x}_j))} G(\mathbf{x}_j)/|X_P|$

$M_Q^t \leftarrow \alpha M_Q^{t-1} + (1-\alpha) m_Q^t$

$M_P^t \leftarrow \alpha M_P^{t-1} + (1-\alpha) m_P^t$

$\widehat{M_Q^t} \leftarrow M_Q^t/(1-\alpha^t)$

$\widehat{M_P^t} \leftarrow M_P^t/(1-\alpha^t)$

$\mathcal{L}_{FSMM} \leftarrow \left\| \widehat{M_Q^t} - \widehat{M_P^t} \right\|_2$

$\mathcal{L}_D \leftarrow \frac{\sum_{\mathbf{x}_j \in X_P} \log(D(G(\mathbf{x}_j)))}{2|X_P|} + \frac{\sum_{\mathbf{x}_i \in X_Q} \log(1-D(G(\mathbf{x}_i)))}{2|X_Q|}$

Update the discriminator D and the feature extractor G by ascending along the gradients:

$$\nabla_{\theta_D}(\lambda_1 \mathcal{L}_D - \lambda_2 \mathcal{L}_{FSMM}) \text{ and } \nabla_{\theta_G}(\lambda_1 \mathcal{L}_D - \lambda_2 \mathcal{L}_{FSMM})$$

2: **Step 2**

For $\mathbf{x}_j \in X_P$, $w(\mathbf{x}_j) \leftarrow \frac{1-D(G(\mathbf{x}_j))}{D(G(\mathbf{x}_j))}$

$\mathcal{L}_C = \frac{\sum_{\mathbf{x}_j \in X_P} w(\mathbf{x}_j) \ell(C(G(\mathbf{x})), y)}{|X_P|}$

Update the classifier C and the feature extractor G by ascending along the gradients: $\nabla_{\theta_C} \mathcal{L}_C$ and $\nabla_{\theta_G} \mathcal{L}_C$ Here we ignore the gradients coming from the weights $w(\mathbf{x}_j)$, that is, we consider the $w(\mathbf{x}_j)$ as constants.

classification loss, that is,

$$\mathcal{L}_C = \mathbb{E}_{\mathbf{x} \sim P}[w(\mathbf{x}) \ell(C(G(\mathbf{x})), y)], \tag{4.6}$$

where $\ell(\cdot, \cdot)$ is typically the cross-entropy loss. The classification loss $\mathcal{L}_C$ not only is used for the classification task but also ensures that the feature space given by the feature extractor G represents the important characteristics of the raw data.

End-to-End Learning for SCN

One straightforward way to train SCN is to use mini-batch stochastic gradient descent (SGD) for optimization. However, the feature space mean matching loss $\mathcal{L}_{FSMM}$ could have a large variance with small batch sizes. For example, if the two sampled batches X_P and X_Q have very few similar features $G(\mathbf{x}_i)$, the $\mathcal{L}_{FSMM}$ could be very large even with the optimal shift factor. Therefore, instead of estimating $\mathcal{L}_{FSMM}$ based on each mini batch, we maintain the moving averages of both the weighted training features M_P and the test features M_Q. Algorithm 2 shows the pseudocode of our two-step learning scheme for SCN. In the first step, we update the moving averages of both the training data and the test data using the features extracted by G with hyperparameter α. Then we use the losses $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$ to update the parameters in the feature extractor G and the discriminator D with hyperparameters λ_1 and λ_2, respectively, which adjusts the importance of $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$. (We set $\lambda_1 = 1$ and $\lambda_2 = 0.1$ in our experiments.) In the second step, we update the classifier C and the feature extractor G using the estimated shift factor $w(\mathbf{x})$ from the first step. We initialize the moving averages M_P and M_Q to 0, so that operations (5) and (6) in Algorithm (2) are applied to compensate for the bias caused by initialization to 0, that is,

$$\begin{aligned}
\mathbb{E}[M_Q^t] &= \mathbb{E}[\alpha M_Q^{t-1} + (1-\alpha) m_Q^t] \\
&= \sum_{i=1}^{t} (1-\alpha)\alpha^{i-1} \mathbb{E}[m_Q^i] \\
&\approx \widetilde{\mathbb{E}}[m_Q^i](1-\alpha^t)
\end{aligned} \tag{4.7}$$

Further, since the mini batches are drawn independently, we show that

$$\begin{aligned}
\mathrm{Var}[M_Q^t] &= \mathrm{Var}[\alpha M_Q^{t-1} + (1-\alpha) m_Q^t] \\
&= \sum_{i=1}^{t} (1-\alpha)^2 \alpha^{2i-2} \mathrm{Var}[m_Q^i] \\
&\approx \widetilde{\mathrm{Var}}[m_Q^i] \frac{1-\alpha}{1+\alpha}(1-\alpha^{2t})
\end{aligned} \tag{4.8}$$

That is, by using moving-average estimation, the variance can be reduced by approximately $\frac{1-\alpha}{1+\alpha}$. Consequently, we can apply an α close to 1 to significantly

reduce the variance of $\mathcal{L}_{FSMM}$. However, an α close to 1 implies a strong momentum which is too high for the early-stage training. Empirically, we chose $\alpha = 0.9$ in our experiments.

In the second step of the training process, the shift factor $w(\mathbf{x})$ is used to compensate for the bias between training and test. Note that we must consider the shift factor as a constant instead of as a function of the discriminator D. Otherwise, minimizing the classification loss $\mathcal{L}_C$ would end up trivially causing all the $w(\mathbf{x})$ to be reduced to zero.

4.1.4 Related Work

Different approaches for reducing the data bias problem have been proposed. In mechanism design, [158] proposed a two-stage game for providing incentives to shift the efforts of volunteers to more unexplored tasks in order to improve the scientific quality of the data. In ecology, [112] improved the modeling of presence-only data by aligning the biases of both training data and background samples. [37] explored the complementary strengths of doing a joint analysis of data coming from different sources to reduce the bias. In domain adaptation, various methods [68, 129, 151, 57] have been proposed to reduce the bias between the source domain and the target domain by mapping them to a common feature space while reserving the critical characteristics.

Our work is most closely related to the approaches of [164, 62, 138, 50] developed under the names of *covariate shift* and *sample selection bias*, where the shift factor is learned in order to align the training distribution with the test distribution. The earliest work in this domain came from the statistics and econometrics

communities, where they addressed the use of non-random samples to estimate behavior. [54] addressed sample selection bias, and [99] investigated estimating parameters under *choice-based bias*, cases that are analogous to a shift in the data distribution. Later, [128] proposed correcting models via weighting of samples in empirical risk minimization (ERM) by the shift factor $q(\mathbf{x})/p(\mathbf{x})$.

One straightforward approach to learning the weights is to directly estimate the distributions $p(\mathbf{x})$ and $q(\mathbf{x})$ from the training and test data respectively, using kernel density estimation [128, 137]. However, learning the data distribution $p(\mathbf{x})$ is intrinsically model based and performs poorly with high-dimensional data. [62] and [50] proposed kernel mean matching (KMM), which estimates the shift factor $w(\mathbf{x}) = q(\mathbf{x})/p(\mathbf{x})$ directly via matching the first moment of the covariate distributions of the training and test data in a reproducing kernel Hilbert space (RKHS) using quadratic programming. KLIEP [138] estimates $w(\mathbf{x})$ by minimizing the Kullback-Leibler (KL) divergence between the test distribution and the weighted training distribution. Later, [150] derived an extension of KLIEP for applications with a large test set and revealed a close relationship of that approach to kernel mean matching. Also, [119] and [93] introduced propensity scoring to design unbiased experiments, which they applied in settings related to sample selection bias.

While the problem of covariate shift has received much attention in the past, it has been used mainly in settings where the size of the dataset is relatively small and the dimensionality of the data is relatively low. Therefore, it has not been adequately addressed in settings with massive high-dimensional data, such as hundreds of thousands of high-resolution images. Among the studies in this area, [10] is the one most closely related to ours. They tackled this task by modeling

the sample selection process using Bayesian inference, where the shift factor is learned by modeling the probability that a sample is selected into training data. Though we both use a discriminative model to detect the shift, SCN provides an end-to-end deep learning scheme, where the shift factor and the classification model are learned simultaneously, providing a smoother compensation process, which has considerable advantages for work with massive high-dimensional data and deep learning models. In addition, SCN introduces the feature space mean matching loss, which further improves the quality of the shift factor and leads to a better predictive performance. For the sake of fairness, we adapted the work of [10] to the deep learning context in our experiments.

4.1.5 Experiments

Datasets and Implementation Details

We worked with a crowd-sourced bird observation dataset from the successful citizen science project *eBird* [139], which is the world's largest biodiversity-related citizen science project, with more than 100 million bird sightings contributed each year by eBirders around the world. Even though *eBird* amasses large amounts of citizen science data, the locations of the collected observation records are highly concentrated in urban areas and along major roads, as shown in Fig. 4.1. This hinders our understanding of species distribution as well as inference of migration changes and global climate fluctuation. Therefore, we evaluated our SCN [1] approach by measuring how we could improve multi-species distribution modeling given biased observational data.

[1]Code to reproduce the experiments can be found at https://bitbucket.org/DiChen9412/aaai2019-scn/.

One record in the *eBird* dataset is referred to as a checklist, in which the bird observer records all the species he/she detects as well as the time and the geographical location of the observation site. Crossed with the National Land Cover Dataset for the U.S. (NLCD) [58], we obtained a 16-dimensional feature vector for each observation site, which describes the landscape composition with respect to 16 different land types such as water and forest. We also collected satellite images for each observation site by matching the geographical location of a site to Google Earth, where several preprocesses have been conducted, including cloud removal. Each satellite image covers an area of 17.8 km^2 near the observation site and has 256×256 pixels. The dataset for our experiments was formed by using all the observation checklists from Bird Conservation Regions (BCRs) 13 and 14 in May from 2002 to 2014, which contains 100,978 observations [25]. May is a migration period for BCR 13 and 14; therefore a lot of non-native birds pass over this region, which gives us excellent opportunities to observe their choice of habitats during the migration. We chose the 50 most frequently observed birds as the target species, which cover over 97.4% of the records in our dataset. Because our goal was to learn and predict multi-species distributions across landscapes, we formed the unbiased test set and the unbiased validation set by overlaying a grid on the map and choosing observation records spatially uniformly. We used the rest of the observations to form the spatially biased training set. Table 4.1 presents details of the dataset configuration.

In the experiments, we applied two types of neural networks for the feature extractor G: multi-layer fully connected networks (MLPs) and convolutional neural networks (CNNs). For the NLCD features, we used a three-layer fully connected neural network with hidden units of size 512, 1024 and 512, and with ReLU [107] as the activation function. For the Google Earth images, we used

Feature Type	NLCD	Google Earth Image
Dimensionality	16	$256 \times 256 \times 3$
#Training Set	79060	79060
#Validation Set	10959	10959
#Test Set	10959	10959
#Labels	50	50

Table 4.1: Statistics of the *eBird* dataset

DenseNet [61] with minor adjustments to fit the image size. The discriminator D and Classifier C in SCN were all formed by three-layer fully connected networks with hidden units of size 512, 256, and #*outcome*, and with ReLU as the activation function for the first two layers; there was no activation function for the third layer. For all models in our experiments, the training process was done for 200 epochs, using a batch size of 128, cross-entropy loss, and an Adam optimizer [74] with a learning rate of 0.0001, and utilized batch normalization [64], a 0.8 dropout rate [133], and early stopping to accelerate the training process and prevent overfitting.

Analysis of Performance of the SCN

We compared the performance of SCN with baseline models from two different groups. The first group included *models that ignore the covariate shift* (which we refer to as vanilla models), that is, models are trained directly by using batches sampled uniformly from the training set without correcting for the bias. The second group included *different competitive models for solving the covariate shift problem*: (1) kernel density estimation (KDE) methods [128, 137], (2) the Kullback-Leibler Importance Estimation Procedure (KLIEP) [138], and (3) discriminative factor weighting (DFW) [10]. The DFW method was implemented initially by using a Bayesian model, which we adapted to the deep learning model in order to

NLCD Feature			
Test Metrics (%)	**AUC**	**AP**	**F1 score**
vanilla model	77.86	63.31	54.90
SCN	**80.34**	**66.17**	**57.06**
KLIEP	78.87	64.33	55.63
KDE	78.96	64.42	55.27
DFW	79.38	64.98	55.79
Google Earth Image			
vanilla model	80.93	67.33	59.97
SCN	**83.80**	**70.39**	**62.37**
KLIEP	81.17	67.86	60.23
KDE	80.95	67.42	60.01
DFW	81.99	68.44	60.77

Table 4.2: Comparison of predictive performance of different methods under three different metrics. (The larger, the better.)

use it with the *eBird* dataset. We did not compare SCN with the kernel mean matching (KMM) methods [62, 50], because KMM, like many kernel methods, requires the construction and inversion of an $n \times n$ Gram matrix, which has a complexity of $O(n^3)$. This hinders its application to real-life applications, where the value of n will often be in the hundreds of thousands. In our experiments, we found that the largest n for which we could feasibly run the KMM code is roughly 10,000 (even with SGD), which is only 10% of our dataset. To make a fair comparison, we did a grid search for the hyperparameters of all the baseline models to saturate their performance. Moreover, the structure of the networks for the feature extractor and the classifier used in all the baseline models, were the same as those in our SCN (i.e., G and C), while the shift factors for those models were learned using their methods.

Table 4.2 shows the average performance of SCN and other baseline models with respect to three different metrics: (1) **AUC**, area under the ROC curve; (2) **AP**, area under the precision–recall curve; (3) **F1 score**, the harmonic mean of precision and recall. Because our task is a multi-label classification problem, these three metrics were averaged over all 50 species in the datasets. In our ex-

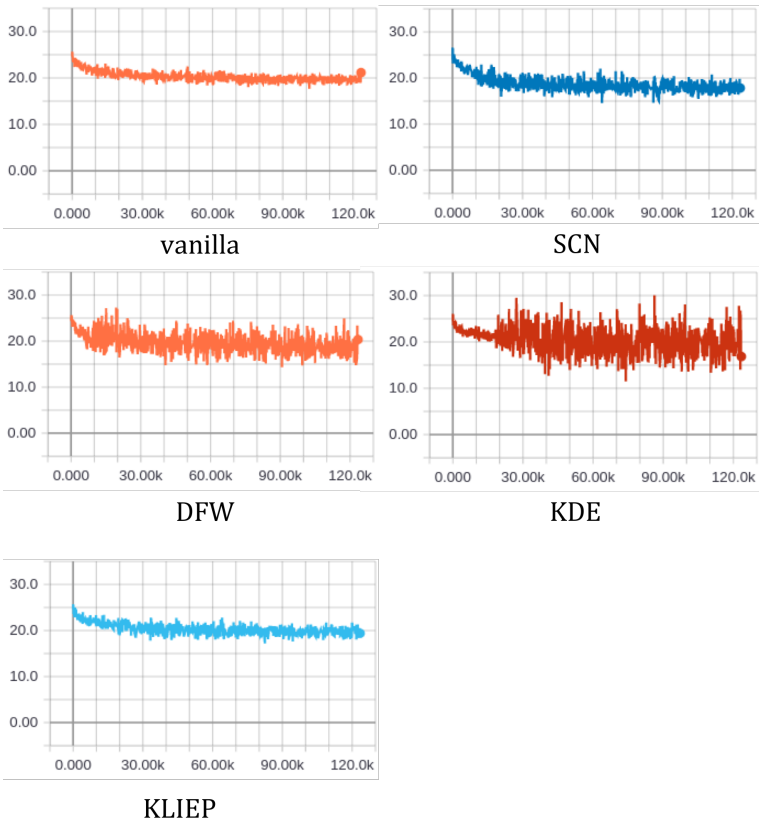


Figure 4.3: The learning curves of all models. The vertical axis shows the cross-entropy loss, and the horizontal axis shows the number of iterations.

periments, the standard error of all the models was less than 0.2% under all three metrics. There are two key results in Table 4.2: **(1) All bias-correction models outperformed the vanilla models under all metrics, which shows a significant advantage of correcting for the covariate shift. (2) SCN outperformed all the other bias-correcting models, especially on high-dimensional Google Earth images.**

The kernel density estimation (KDE) models had the worst performance, especially on Google Earth images. This is not only because of the difficulty of modeling high-dimensional data distributions, but also because of the sensitivity of the KDE approach. When $p(\mathbf{x}) \ll q(\mathbf{x})$, a tiny perturbation of $p(\mathbf{x})$ could result in a huge fluctuation in the shift factor $q(\mathbf{x})/p(\mathbf{x})$. KLIEP performed slightly better

than KDE, by learning the shift factor $w(\mathbf{x}) = \frac{q(\mathbf{x})}{p(\mathbf{x})}$ directly, where it minimized the KL divergence between the weighted training distribution and the test distribution. However, it showed only a slight improvement over the vanilla models on Google Earth images. DFW performed better than the other two baseline models, which is not surprising, given that DFW learns the shift factor by using a discriminator similar to the one in SCN. SCN outperformed DFW not only because it uses an additional loss, the feature space mean matching loss, but also because of its end-to-end training process. DFW first learns the shift factor by optimizing the discriminator, and then it trains the classification model using samples weighted by the shift factor. However, SCN learns the classifier C and the discriminator D simultaneously, where the weighted training distribution approaches the test distribution smoothly through the training process, which performed better empirically than directly adding the optimized weights to the training samples. [153] also discovered a similar phenomenon in cost-sensitive learning, where pre-training of the neural network with unweighted samples significantly improved the model performance. One possible explanation of this phenomenon is that the training of deep neural networks depends highly on mini-batch SGD, so that the fluctuation of gradients caused by the weights may hinder the stability of the training process, especially during the early stage of the training. Fig. 4.3 shows the learning curves of all five models, where we used the same batch size and an Adam optimizer with the same learning rate. As seen there, SCN had a milder oscillation curve than DFW, which is consistent with the conjecture we stated earlier. In our experiments, we pre-trained the baseline models with unweighted samples for 20 epochs in order to achieve a stable initialization. Otherwise, some of them would end up falling into a bad local minimum, where they would perform even worse than the vanilla models.

NLCD Feature			
Test Metrics (%)	**AUC**	**AP**	**F1-score**
SCN	**80.34**	**66.17**	**57.06**
SCN_D	79.53	65.11	56.11
SCN_FSMM	79.58	65.17	56.26
SCN$^-$	80.09	65.97	56.83
Google Earth Image			
SCN	**83.80**	**70.39**	**62.37**
SCN_D	82.35	68.96	61.23
SCN_FSMM	82.49	69.05	61.51
SCN$^-$	83.44	69.72	62.01

Table 4.3: Comparison of predictive performance of the different variants of SCN

To further explore the functionality of the discriminative loss and the feature mean matching loss in SCN, we implemented several variants of the original SCN model:

- SCN: The original Shift Compensation Network
- SCN_D: The Shift Compensation Network without the feature space mean matching loss ($\lambda_2 = 0$)
- SCN_FSMM: The Shift Compensation Network without the discriminative loss ($\lambda_1 = 0$)
- SCN$^-$: The Shift Compensation Network without using moving-average estimation for the feature space mean matching loss ($\alpha = 0$)

Table 4.3 compares the performance of the different variants of SCN, where we observe the following: (1) Both the discriminative loss and the feature space mean matching loss play an important role in learning the shift factor. (2) The moving-average estimation for the feature space mean matching loss shows an advantage over the batch-wise estimation (compare SCN to SCN$^-$). (3) Crossed with Table 4.2, SCN performs better than DFW, even with only the discriminative loss, which shows the benefit of fitting the shift factor gradually through the

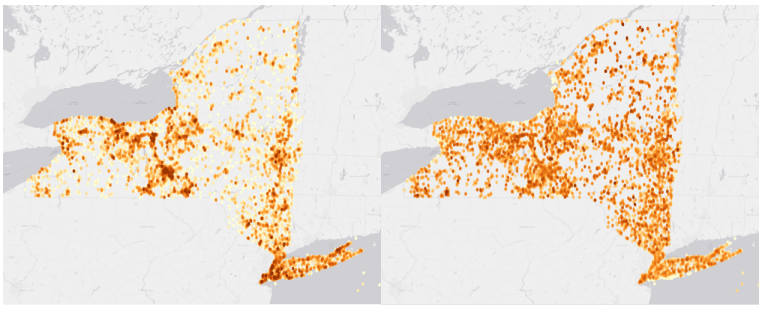


Figure 4.4: Heatmap of the observation records for the month of May in New York State, where the left panel shows the distribution of the original samples and the right one shows the distribution weighted with the shift factor

training process. (4) Surprisingly, even if we use only the feature space mean matching loss, which would potentially lead to ill-posed weights, SCN_FSMM still shows much better performance than the other baselines.

Shift Factor Analysis

We visualized the heatmap of the observation records for the month of May in New York State (Fig. 4.4), where the left panel shows the distribution of the original samples and the right one shows the distribution weighted with the shift factor. The colors from white to brown indicates the sample popularity from low to high using a logarithmic scale from 1 to 256. As seen there, the original samples are concentrated in the southeastern portion and Long Island, while the weighted one is more balanced over the whole state after applying the shift factor. This illustrates that SCN learns the shift correctly and provides a more balanced sample distribution by compensating for the shift.

We investigated the shift factors learned from the different models (Table 4.4) by analyzing the ratio of the feature space mean matching loss to the dimension-

Averaged Feature Space Mean Matching Loss	
vanilla model	0.8006
SCN	0.0182
KLIEP	0.0015
KDE	0.0028
DFW	0.0109

Table 4.4: Feature space discrepancy between the weighted training data and the test data

ality of the feature space using equation (4.9).

$$\begin{aligned} &\left\|\mathbb{E}_{\mathbf{x}\sim Q}[\Phi(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})\Phi(\mathbf{x})]\right\|_2 / \dim(\Phi(\mathbf{x})) \\ \approx &\left\|\frac{1}{|X_Q|}\sum_i \Phi(\mathbf{x}_i) - \frac{1}{|X_P|}\sum_i w(\mathbf{x}_i)\Phi(\mathbf{x}_i)\right\|_2 / \dim(\Phi(\mathbf{x})) \end{aligned} \tag{4.9}$$

Here, we chose the output of the feature extractor in each model (such as $G(\mathbf{x})$ in SCN) as the feature space $\Phi(\mathbf{x})$. Compared to the vanilla models, all the shift correction models significantly reduced the discrepancy between the weighted training distribution and the test distribution. However, crossed with Table 4.2, it is interesting to see the counter-intuitive result that the models with the smaller feature space discrepancies (KDE & KLIEP) did not necessarily perform better.

4.1.6 Discussion

In this section, we proposed the Shift Compensation Network (SCN) along with an end-to-end learning scheme for solving the covariate shift problem in citizen science. We incorporated the discriminative loss and the feature space mean matching loss to learn the shift factor. Tested on a real-world biodiversity-related citizen science project, *eBird*, we show that SCN significantly improves multi-species distribution modeling by learning and correcting for the data bias, and that it consistently performs better than previous models. We also discovered the importance of fitting the shift factor gradually through the training process,

which raises an interesting question for future research: How do the weights affect the performance of models learned by stochastic gradient descent?

4.2 Task-Based Learning via Task-Oriented Prediction Network with Applications in Finance

4.2.1 Introduction

Prediction models have been widely used to facilitate decision making across domains, for example, retail demand prediction for inventory control [118], user behavior prediction in regard to display ads [159], and financial market movement prediction for portfolio management [115], to name a few. These models are often trained using standard machine learning loss functions, such as mean square error (MSE), mean absolute error (MAE), and cross-entropy loss (CE). However, these criteria, which are commonly used to train prediction models, are often different from the task-based criteria used to evaluate model performance [8, 28]. For instance, a standalone image classification model is often trained by optimizing cross-entropy loss. However, when it is used to guide autonomous driving, we may care more about misclassifying a traffic sign than about misclassifying a garbage can. In revenue surprise forecasting, financial institutes often train a regression model to predict the revenue surprise for each public company by minimizing mean square error. However, they evaluate the model performance based on the *Directional Accuracy* (percentage of predictions that are directionally more accurate) and the *Magnitude Accuracy* (percentage of predictions that are 50% or more accurate) with respect to industry

benchmarks (e.g., the consensus of Wall Street analysts)[2], which provide more value for downstream portfolio management. In loan default risk modeling, banks often train a classification model to predict the default probability of each loan application and to optimize the probability threshold for accepting/rejecting loans with low/high risk. Eventually, they evaluate the model performance by aggregating the total profit made on those loans.

Despite the popularity of standard machine learning losses, models trained with such standard losses are not necessarily aligned with the task-based evaluation criteria, and as a result may perform poorly with respect to the ultimate task-based objective. One straightforward solution to this problem is to directly use the task-based evaluation criteria as the loss function. However, task-based evaluation criteria are often unfriendly to an end-to-end gradient-based training process, because such performance objectives are not necessarily differentiable and may even require additional decision-making optimization processing. Existing works [34, 8, 28, 154, 111, 155] in this area focus mainly on deriving heuristic surrogate loss functions that differentiate from downstream evaluation criteria to the upstream prediction model via certain relaxations or KKT conditions. However, those derivations are mainly handcrafted and task specific. As a result, a considerable amount of effort to find proper surrogate losses for new tasks is required, especially when the evaluation criteria are complicated or involve non-convex optimization. Moreover, handcrafted surrogate losses are not optimized, hence they can hardly become the optimal choice. Therefore, a general end-to-end learning scheme which can automatically integrate the task-based evaluation criteria is still needed.

We propose the Task-Oriented Prediction Network (TOPNet), a generic end-

[2]https://www.investopedia.com/terms/c/consensusestimate.asp

to-end learning scheme that automatically integrates task-based evaluation criteria into the learning process via a learnable differentiable surrogate loss function, which approximates the true task-based loss and directly guides the prediction model to the task-based goal. Specifically, **(i)** TOPNet learns a differentiable surrogate loss function parameterized by a task-oriented loss estimator network that approximates the true task-based loss given the prediction, the ground-truth label, and the necessary contextual information. **(ii)** TOPNet optimizes a predictor using the learned surrogate loss function and uses it to approximately optimize its performance w.r.t. the true task-based loss. **(iii)** We demonstrate the performance of TOPNet on two real-world financial prediction tasks, a revenue surprise forecasting task and a credit risk modeling task, where the former is a regression task and the latter is a classification task. Applying TOPNet to these two tasks, we show that TOPNet significantly boosts the ultimate task-based goal by integrating the task-based evaluation criteria, outperforming both traditional modeling with standard losses and modeling with heuristic differentiable (relaxed) surrogate losses.

4.2.2 Related Work

Integrating task-based evaluation criteria into the learning process was studied under different names, such as *task-based learning* and *decision-focused learning*. The earliest work [8], which is closely related to ours, optimizes the neural network based on returns obtained via a hedging strategy, to predict financial prices. Later, [73] proposed Directed Regression, which minimizes a convex combination of least square loss and a task-based loss, to achieve a better regression performance w.r.t. the decision objective. [34] derived a convex surrogate loss function

called SPO+ loss via duality theory, to leverage the upstream prediction model and the downstream optimization task for linear programming. [28] proposed task-based model learning for stochastic programming, where they differentiate through the KKT condition of the convex objective, to provide gradients for the upstream prediction model to capture the downstream optimization objective. Recent works [111, 154, 155] applied a similar idea to security games, combinatorial optimization problems and graph optimization problems, to integrate the downstream objectives into the upstream modeling.

Those previous works [8, 34, 28, 111, 154, 155] mainly focus on deriving a differentiable surrogate loss function for the downstream evaluation criteria to provide gradients to the upstream prediction model. Even though those works have developed many surrogate losses for different evaluation criteria, their approaches either require the objective to be convex or use handcrafted relaxation to approximate the ultimate objective. In contrast, Task-Oriented Prediction Network (TOPNet) does not require handcrafted differentiation of the downstream evaluation criteria. Instead, TOPNet learns a differentiable surrogate loss via a task-oriented loss estimator network, which automatically approximates the true task-based loss and directly guides the upstream predictor towards the downstream task-based goal. In the context of task-based learning, TOPNet is the first work that automatically integrates the true task-based evaluation criteria into an end-to-end learning process via a learnable surrogate loss function.

4.2.3 Problem Formulation

We first formally define the task-based prediction problem that we address in this paper. We use $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y \in \mathcal{Y}$ for the feature and label variables. Given dataset $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) ...,(\mathbf{x}_n, y_n)\}$, which is sampled from an unknown data distribution P with density function $p(x, y)$, our prediction task can be formulated as learning a conditional distribution $q_\theta(\hat{y}|\mathbf{x})$ that minimizes the expected task-based loss (task-based criteria) $\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$, i.e.,

$$\min_\theta \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)], \tag{4.10}$$

where c denotes some necessary contextual information related to task-based criteria, $p(\mathbf{x})$ denotes the marginal distribution of $\mathbf{x}$, and θ denotes the parameters of our prediction model. As implied in formulation (4.10), we mainly consider the tasks whose task-based losses can be computed point-wisely.

A key challenge of task-based learning comes from the fact that the true task-based loss function $\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ is often non-differentiable and may even involve additional decision-making optimization processing, which cannot be used directly in popular gradient-based learning methods. For instance, in revenue surprise forecasting, the task-based criteria evaluate a prediction $\hat{y}$ based on both the true revenue surprise y and the prediction of the consensus of the Wall Street analysts c (in that case, both $q_\theta(\hat{y}|\mathbf{x})$ and $p(y|\mathbf{x})$ are Dirac delta distribution). Specifically, the criteria compute whether the prediction is more directional accurate and whether the prediction is significantly (50%) more accurate compared with the Wall Street consensus, which both involve non-differentiable functions (see detailed formula in our experiments). Likewise, in credit risk modeling, the task-based criteria involve optimizing a probability decision threshold p_D to maximize the profit after approving all loan applications with a predicted default

probability p_i lower than p_D.

A straightforward solution to this challenge is to use a surrogate loss function $\ell^S(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ to replace the true task-based loss and guide the learning process. Existing works mainly focus on using standard machine learning loss functions, such as mean square error (MSE), mean absolute error (MAE) and cross-entropy loss (CE), or other task-specific differentiable loss functions [8, 34, 28, 111, 154, 155] as the surrogate loss, that is,

$$\min_\theta \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)]. \tag{4.11}$$

However, both standard machine learning losses and task-specific differentiable losses are selected manually. Thus, finding a proper surrogate loss function requires a considerable amount of effort, especially when the evaluation criteria are complicated or involve non-convex optimization. Therefore, such approaches require considerable customization and do not provide a general methodology to task-based learning.

4.2.4 Task-Based Learning via A Learnable Differentiable Surrogate Loss

Instead of manually designing a handcrafted differentiable loss, we propose to learn a differentiable surrogate loss function $\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ via a neural network parameterized by ω, to approximate the true task-based loss and guide the prediction model. Specifically, we formulate the task-based learning problem as a bilevel optimization, i.e.,

$$\min_\theta \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S_{\omega^*}(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)] \tag{4.12}$$

subject to:

$$\omega^* = \underset{\omega}{\operatorname{argmin}}\, \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[D(\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)||\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c))] \tag{4.13}$$

, where $D(\cdot||\cdot)$ is a discrepancy function.

In this paper, we assume that both $\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ and $\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ are real-valued loss functions. Thus, we mainly consider using absolute error loss or square error loss as the discrepancy function, i.e., $D(x||y) = |x - y|$ or $D(x||y) = (x - y)^2$.

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)] \\
\leq &\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)]] + \qquad (4.14)\\
&\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[|\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c) - \ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)|] \\
\leq &\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)]] + \qquad (4.15)\\
&\mathbb{E}^{1/2}_{\mathbf{x}\sim p(\mathbf{x})}[(\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c) - \ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c))^2] \\
&\text{(Jensen's Inequality)}
\end{aligned}$$

As shown in the inequality (4.14) and (4.15), if we use absolute/square error loss as the discrepancy function and minimize the discrepancy term (4.13) to a small value ϵ/ϵ^2, then we have

$$\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)] \leq \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)] + \epsilon\,.$$

Therefore, since the expected true task-based loss is upper bounded by the expected surrogate loss plus the discrepancy, we can approximately (with an ϵ-tolerance) learn the prediction model $q_\theta(\hat{y}|\mathbf{x})$ w.r.t. the task-based loss via solving the above bilevel optimization problem.

One straightforward idea to tackle the above bilevel optimization problem is to use Lagrangian relaxation (LR), i.e.,

$$\min_{\theta,\omega}\ \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)] +$$

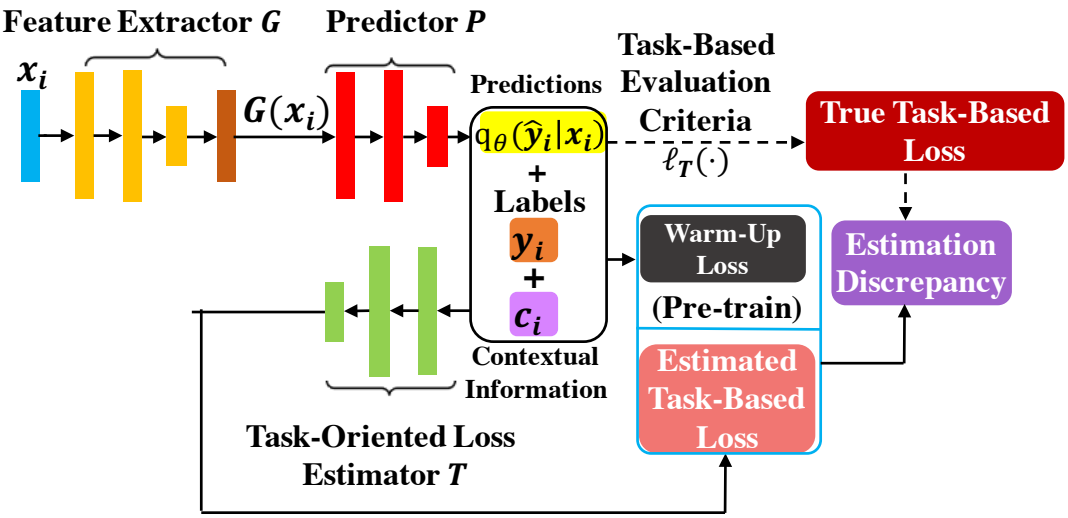


Figure 4.5: Overview of the Task-Oriented Prediction Network.

$$\lambda \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[D(\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)||\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c))]$$

, where λ is a non-negative weight (we set $\lambda = 1$). (4.16)

However, given the fact that $\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ is non-differentiable, we cannot directly use gradient-based method to minimize LR (4.16) w.r.t. both θ and ω. Fortunately, though the second term in the LR (4.16) is non-differentiable w.r.t. θ, it is differentiable w.r.t. ω given the fact that $\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ does not involve ω and $\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$ is differentiable. Therefore, instead of minimizing LR (4.16) directly using all parameters, we propose to separate the optimization regarding θ and ω, and only minimize the first term in LR (4.16) w.r.t. θ, i.e.,

$$\min_\theta \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)] \tag{4.17}$$

$$\min_\omega \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)]+ \tag{4.18}$$

$$\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[D(\ell^S_\omega(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)||\ell^T(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c))]$$

Intuitively, we are alternating between (i) optimizing the prediction model $q_\theta(\hat{y}|\mathbf{x})$ w.r.t. the current learned surrogate loss and (ii) minimizing the gap between the learned surrogate loss and the true task-based loss obtained from the current

Algorithm 3 End-to-End learning process for TOPNet

Require: $\mathbf{x}_i$, y_i and c_i are raw input features, ground-truth label and corresponding contextual information sampled iid from the training set D_{train}. $\ell^T(\cdot,\cdot,\cdot)$ is the true task-based loss function. $\ell^W(\cdot,\cdot,\cdot)$ is the warm-up loss function. $D(\cdot\|\cdot)$ is the loss discrepancy function. T, P and G denote the task-based loss estimator, the predictor and the feature extractor respectively. N_{train} is the number of training iterations. N_{pre} is the number of iterations for "warm-up" pretraining. For ease of presentation, here we assume the batch size is 1.

for $t \leftarrow 1$ to N_{train} **do**
  Sample a data point $(\mathbf{x}_i, y_i)$ from D_{train}.
  Make prediction $q_\theta(\hat{y}_i|\mathbf{x}_i) = P(G(\mathbf{x}_i))$.
  Invoke the true task-based criteria to compute the true task-based loss $\ell^T(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)$.
  Approximate the true task-based loss using the learnable surrogate loss $\ell^S_{\omega_T}(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i) = T(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)$.
  Update the task-oriented estimator T via $\min_{\omega_T} D(\ell^S_{\omega_T}(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)\|\ell^T(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i))$.
  if $t \leq N_{\text{pre}}$ **then**
    Update the prediction model (P and G) using the warm-up loss: $\min_{\theta_G,\theta_P} \ell^W(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)$.
  else
    Update the prediction model (P and G) using the learned surrogate loss: $\min_{\theta_G,\theta_P} \ell^S_{\omega_T}(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)$.
  end if
end for

prediction model. One can see, the learning of the prediction model and the surrogate loss depends on each other. Thus, a bad surrogate loss would mislead the prediction model and vice versa. For example, if the true task-based loss is a bounded loss function, then with a bad prediction model the learned surrogate loss is likely to get stuck on some insensitive area, where the loss is saturated due to the huge difference between $q_\theta(\hat{y}|\mathbf{x})$ and $p(y|\mathbf{x})$. Therefore, instead of starting learning the prediction model with a randomly initialized surrogate loss function, we propose to "warm-up" the prediction model $q_\theta(\hat{y}|\mathbf{x})$ with a designed warm-up loss function $\ell^W(q_\theta(\hat{y}|\mathbf{x}), p(y|\mathbf{x}), c)$. Thus, we can warm up the prediction model to

be close to the ground truth so that the learning of the surrogate loss would focus more on the sensitive area and better boost the task-based performance. In our experiments, we investigated different warm-up losses ranging from standard machine learning losses to heuristic surrogate losses. We empirically show that the model would achieve a better performance with the "warm-up" step.

4.2.5 End-to-End Implementation via Task-Oriented Prediction Network

We instantiate the task-based learning process described above via the Task-Oriented Prediction Network (TOPNet). As depicted in Fig.4.5, a feature extractor G is first applied to extract meaningful features from the raw input data $\mathbf{x}_i$. Then, a predictor network P takes the extracted feature $G(\mathbf{x}_i)$ to predict the conditional distribution $P(G(\mathbf{x}_i)) = q_\theta(\hat{y}_i|\mathbf{x}_i)$ (θ denotes the parameters in P and G). Note that, in practice, we do not have access to the true distribution $p(y, \mathbf{x})$. Therefore we use the empirical distribution, i.e., a uniform distribution $p(y_i, \mathbf{x}_i)$ over samples in the dataset, to replace $p(y, \mathbf{x})$. Given the fact that the conditional distribution $p(y_i|\mathbf{x}_i)$ is indeed a Dirac Delta distribution over the value y_i, for ease of presentation, we use the point-wise ground truth label y_i to replace the role of $p(y_i|\mathbf{x}_i)$ in the following content. With our prediction $q_\theta(\hat{y}_i|\mathbf{x}_i)$, the ground truth label y_i and necessary contextual information c_i concerning the task, we can invoke the true task-based evaluation criteria, which potentially involve a decision-making optimization process, to generate the true task-based loss $\ell^T(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)$. Meanwhile, a task-oriented loss estimator network T takes the predictions $q_\theta(\hat{y}_i|\mathbf{x}_i)$, the labels y_i, and the contextual information c_i, to ap-

proximate the true task-based loss via minimizing the discrepancy between the learned surrogate loss $\ell^S_{\omega_T}(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)$ (ω_T denotes the parameters in T) and the true task-based loss. Finally, we can update the prediction model using the gradients obtained from the learned surrogate loss function. As we discussed in the previous section, to facilitate the learning of both $q_\theta(\hat{y}|\mathbf{x})$ and $\ell^S_{\omega_T}(q_\theta(\hat{y}_i|\mathbf{x}_i), y_i, c_i)$, we propose to warm-up the prediction model using a warm-up loss function $\ell^W(q_\theta(\hat{y}|\mathbf{x}), y_i, c_i)$, which could be either a standard machine learning loss or a designed heuristic loss, for the first N_{pre} iterations. In our experiments, we use the square error as the loss discrepancy function $D(\cdot\|\cdot)$ due to its better empirical performance compared with the absolute error. We empirically set the hyper-parameter $N_{\text{pre}} = |D_{train}|$ to just warm up the prediction model for one training epoch. We summarize the implementation of the alternative minimizing process in Algorithm 3.

4.2.6 Experimental Results

TOPNet is a generic learning scheme that can be used in a variety of applications with task-based criteria. In this section, we validate its performance via datasets from two real-world applications in finance. Due to business confidentiality, we are not allowed to share the datasets. The experiments are mainly designed to compare the benefit of using TOPNet learning scheme over standard machine learning schemes or handcrafted heuristic surrogate loss functions.

General Experimental Setup: For all models in our experiments, the training process was done for 50 epochs, using a batch size of 1024, an Adam optimizer [74] with a learning rate of 3e-5, and early stopping to accelerate the training

process and prevent overfitting.

Revenue Surprise Forecasting

Revenue growth is the key indicator of the valuation and profitability of a company and it is widely used for investment decisions [65], such as stock selection and portfolio management. Due to the long tail distribution of revenue growth, the investment communities usually predict revenue surprise which is given by revenue growth minus *consensus*. Here, *consensus* is the average of the Wall street estimates of revenue growth published by stock analysts. Despite the fact that revenues are published quarterly, daily forecasts of revenue surprise enable investors to adjust their portfolio in a granular way for return and risk analysis. To predict quarterly revenue surprise at the daily level before their announcement, we collect information including quarterly revenue, consensus, stock price and various of financial indicators of 1090 US public companies ranging from Jan 1st, 2004 to June 30th, 2019. Each data point is associated with a 10x12-dimensional feature vector describing up-to-date sequential historical information of the corresponding company. The label of each data point is a real number describing the revenue surprise of the corresponding company on that specific date. We split the whole dataset chronologically into training set (01-01-2004 to 06-30-2015, 3,267,584 data points), validation set (07-01-2015 to 06-30-2017, 465,383 data points) and test set (07-01-2017 to 06-30-2019, 421,225 data points) to validate the performance of models. Note that some companies only have a few data points due to their short history. Thus, we filtered companies to make sure that all remaining companies have enough (1,000) historical data points in the training set and end up using 902 companies in our experiments.

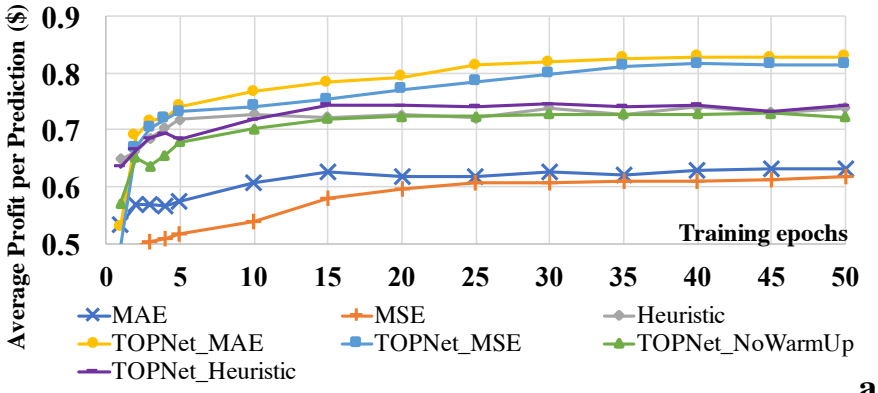
Even though we have about 4 million data points, on average each company only has about 3,600 training examples. Therefore, instead of learning a model for each company, we aim to use all data points to learn a company-agnostic prediction model. Though it is possible to build a multi-task learning framework for this specific task, it is out of the scope of this paper.

Task-based Criteria In this regression problem, the task-based criterion is the total reward calculated based on the Directional Accuracy (DirAcc) and the Magnitude Accuracy (MagAcc) with respect to the industry benchmark, *consensus*. To be specific,

$$\text{DirAcc}_i = \begin{cases} \alpha & \text{if } \text{sign}(\tilde{\hat{y}}_i) = \text{sign}(\tilde{y}_i) \\ -\beta & \text{if } \text{sign}(\tilde{\hat{y}}_i) \neq \text{sign}(\tilde{y}_i) \end{cases} \quad \text{MagAcc}_i = \begin{cases} \gamma & \text{if } |y_i - \hat{y}_i| < 0.5|y_i| \\ 0 & \text{otherwise} \end{cases}$$

where $\tilde{\hat{y}}_i = \hat{y}_i - \text{median}(\hat{y})$, $\tilde{y}_i = y_i - \text{median}(y)$, $\hat{y}_i$ (y_i) denotes predicted (true) revenue surprise of a public company at a specific date, $\text{sign}(\cdot)$ denotes the sign function, and $\text{median}(\cdot)$ represents the median of the predicted (true) revenue surprise of data points of all the companies within the same quarter as the i-th data point. Here, we use DirAcc_i and MagAcc_i to denote the Directional Hit/Miss and Magnitude Hit/Miss of data point i, and α, β and γ are 3 parameters denoting the reward/penalty of Directional Hit, Directional Miss, and Magnitude Hit. In our experiments, we set $\alpha = \$5.00$, $\beta = \$6.11$ and $\gamma = \$2.22$ based on business judgement.

Intuitively, the DirAcc measures the percentage of predictions among all the companies that are more *directional* accurate than the industry benchmark, which is critical for long/short investment decisions. The DirAcc uses the median as the anchor to adjust both our prediction and the label in order to cancel the seasonal trend within a quarter. The MagAcc evaluates the percentage of predictions



a

Models	Average Profit per Prediction ($)
MAE	0.638±0.028
MSE	0.611±0.039
Heuristic	0.732±0.037
TOPNet_MAE	**0.828±0.011**
TOPNet_MSE	0.815±0.015
TOPNet_NoWarmUp	0.725±0.045
TOPNet_Heuristic	0.745±0.021

b

Figure 4.6: The task-based performance of all models in the revenue surprise forecasting task. **a**. Evaluation on the validation set along the training process. **b**. Evaluation (mean and stderr) on the test set for 15 runs of all models. The TOPNet warmed up with MAE (TOPNet_MAE) achieved the best performance.

that are significantly (50%) more accurate than the industry benchmark, which is the essential input for optimizing the weight of stocks in a portfolio. Given DirAcc_i and MagAcc_i, the task-based goal is to maximize the average profit the model earned from n predictions, i.e., $\frac{1}{n}\sum_{i=1}^{n} \text{DirAcc}_i + \text{MagAcc}_i$. Since algorithm 3 minimizes the loss function, we use the negative of equation (4.2.6) as the task-based loss in TOPNets.

Benchmark Methods

(i) Models that are trained with standard machine learning loss function: In this regression task, we selected mean square error (MSE) loss and mean absolute error (MAE) loss as candidates of standard machine learning loss functions.

(ii) Models that are trained with heuristic surrogate loss functions: Given the task-based criteria, we observe that a proper heuristic surrogate loss function could be designed by approximating DirAcc_i and MagAcc_i using $\tanh(\cdot)$, i.e.,

$$\begin{aligned}
\text{DirAcc}_i &\approx \alpha(1 + \text{sign}(\tilde{\hat{y}}_i \cdot \tilde{y}_i))/2 + \beta(1 - \text{sign}(\tilde{\hat{y}}_i \cdot \tilde{y}_i))/2 \\
&\approx \alpha(1 + \tanh(k \cdot \tilde{\hat{y}}_i \cdot \tilde{y}_i))/2 + \beta(1 - \tanh(k \cdot \tilde{\hat{y}}_i \cdot \tilde{y}_i))/2 \\
\text{MagAcc}_i &\approx \gamma(1 + \text{sign}(0.5|y_i| - |y_i - \hat{y}_i|)/2) \\
&\approx \gamma(1 + \tanh(k \cdot (0.5|y_i| - |y_i - \hat{y}_i|))/2)
\end{aligned}$$

Here, k is a scale factor and we neglect some boundary situations such as $\text{sign}(\tilde{\hat{y}}_i) = \text{sign}(\tilde{y}_i) = 0$ and $|y_i - \hat{y}_i| = 0.5|y_i|$. The key idea of this approximation is to approximate $\text{sign}(x)$ with $\tanh(kx)$ since $\lim_{k \to +\infty} \tanh(kx) = \text{sign}(x)$. To saturate the performance of this surrogate loss function, we exhaustively explored the best scale factor k and found that it achieves the best performance with $k = 100$.

Experimental Setup We use the Long Short-Term Memory (LSTM) networks [56] as the feature extractors and 3-layer fully-connected neural networks as the predictors for all models in our experiments. For a fair comparison, we explored the configuration of networks for all models to saturate their performance. For LSTMs and 3-layer fully-connected networks, the number of hidden units are chosen from [64, 128, 256, 512, 1024]. In TOPNets, the task-oriented loss estimator T is a 3-layer fully-connected neural network with hidden units 1024, 512, 256.

Performance Analysis We did 15 runs for all models with different random seed to compute the mean and the standard error of their performance. Since we proposed to "warm up" the predictor, we investigated the performance of TOPNets with different warm-up losses (denoted as TOPNet_MAE, TOPNet_MSE, TOPNet_Heuristic, and TOPNet_NoWarmUp). As shown in Fig.4.6, TOPNets significantly outperformed the standard machine learning models trained with either MSE or MAE, boosting the average profit by about 30%. TOPNets also outperformed the model trained using the handcrafted heuristic surrogate loss function, showing the advantage of using an optimized learnable surrogate loss. Moreover, as we expected, warming up the predictor does significantly (14%) boost the performance compared with the TOPNet without a warm-up step (TOPNet_NoWarmUp). Interestingly, we observe that though the model trained with the heuristic loss alone achieved a better performance than the models trained with MSE or MAE, the heuristic loss actually made it harder to further improve the predictor with the learned surrogate loss. The same phenomenon can also be found in the next task.

Credit Risk Modeling

Credit is a fundamental tool for financial transactions and many forms of economic activity. The main elements of credit risk modeling include the estimation of the probability of default and the loss given default [29]. In this study, our data includes 1.3 million personal loan applications and their payment history. Each loan is associated with an 88-dimensional feature vector and a binary label denoting whether the loan application is defaulted or not. The feature vector includes information such as the loan status (e.g., current, fully paid, default

or charged off), the anonymized applicant's information (e.g., asset, debt, and FICO scores) and the loan characteristics (e.g., amount, interest rate, various cost factors of default), etc. We split the whole dataset randomly into a training set (80%), a validation set (10%), and a test set (10%) to evaluate model performance.

Task-based Criteria The credit risk data provides information to compute the profit/loss of approving a loan application, i.e.,

$$\begin{aligned}\text{Profit/Loss} = &(\text{Received Principle} + \text{Received Interest} - \text{Funded Amount}) \\ &+ (\text{Recovery Amount} - \text{Recovery cost})\end{aligned}$$

Note also that, the recovery happens only if the loan has defaulted and that if we reject a loan application, we simply earn \$0 from it. Recall in credit risk modeling, the task-based criteria involve the prediction of the default probability p_i of the i-th loan application as well as the probability decision threshold p_D to maximize the profit after approving all loan applications with a default probability lower than p_D, i.e.,

$$\frac{1}{n}\sum_{i=1}^{n} \text{Profit/Loss}_i \cdot \text{I}\{p_i < p_D\} + 0 \cdot \text{I}\{p_i \geq p_D\} \tag{4.19}$$

Here, we use $\text{I}\{\cdot\}$ to denote the indicator function.

Benchmark Methods

(i) Models that are trained with standard machine learning loss function: In this classification task, we selected cross-entropy loss as the standard machine learning loss.

(ii) Models that are trained with heuristic surrogate loss functions: Given the profit/loss of approving a loan application and the predicted probability of

default p_i, a natural surrogate loss function is,

$$(1 - p_i) \cdot \text{profit/loss} + p_i \cdot 0,$$

which measures the expected profit/loss given p_i.

Experimental Setup We use 3-layer fully-connected neural networks with hidden units 1024, 512, 256 for the feature extractors G of all models, and the predictors P are linear layers. In TOPNets, the task-oriented loss estimator T is a 3-layer fully-connected neural network with hidden units 1024, 512, 256.

In this task, the evaluation criteria would optimize the decision probability threshold p_D to maximize the average profit via a validation set. Specifically, it would sort the data points based on the predicted default probability p_i and optimize the threshold p_D based on the cumulative sum of the profit/loss of approving load applications with $p_i < p_D$. Note that, TOPNet requires point-wise task-based loss as the feedback from the task-based criteria in the training phase. However, computing the task-based loss involves making decisions (approve/reject), which requires the decision probability threshold p_D that is supposed to be optimized on the validation set. Noting that, the decision probability threshold p_D is a relative value that depends on the predicted default probability p_i. Therefore, maintaining the order of predicted probabilities while shrinking or increasing them together does not affect the ultimate profit but leads to a different optimal threshold. Conversely, given a fixed decision threshold p_D (e.g., 0.5), we can learn a predictor that predicts the default probability with respect to the threshold. Thus, in the learning process of TOPNet, we used a fixed decision threshold (0.5) to make decisions and provide task-based losses in Algorithm 3. During the test, we still apply the same threshold optimization process on the

Models	Average Profit per Loan ($)
Cross-Entropy	618.4 ± 0.3
Heuristic	770.4 ± 0.2
TOPNet_NoWarmUp	770.6 ± 0.2
TOPNet_CE	**784.1 ± 0.2**
TOPNet_Heuristic	777.0 ± 0.3

Table 4.5: Task-based loss results (mean and stderr) of all models in the credit risk modeling task. The TOPNet warmed-up with cross-entropy loss (TOPNet_CE) achieved the best performance.

predictions made by TOPNets as other models.

Performance Analysis We did 15 runs for all models with different random seed to compute the mean and the standard error of their performance. We evaluate the performance of TOPNets that use cross-entropy loss or heuristic loss as the warm-up loss function (denoted as TOPNet_CE and TOPNet_Heuristic). We also evaluate the performance of the TOPNet without a warm-up step. As shown in Table.4.5, TOPNets significantly outperformed the standard machine learning models learned with cross-entropy, boosting the average profit by $165.7. Taking advantage of the optimized learnable surrogate loss function, the TOPNet warmed-up with cross-entropy loss further boosts the profit by $13.5 per loan compared with the model trained using the heuristic loss function. Similar to the phenomenon in the previous task, the TOPNet warmed-up with the heuristic loss function performed slightly worse than the TOPNet warmed-up with cross-entropy loss.

4.2.7 Discussion

In this section, we proposed the Task-Oriented Prediction Network (TOPNet), a generic learning scheme that automatically integrates the true task-based evaluation criteria into an end-to-end learning process via a learnable surrogate loss function. We tested TOPNet on two real-world financial prediction tasks and demonstrate that it can significantly boost the ultimate task-based goal, outperforming both traditional modeling with standard losses and modeling with heurstic differentiable (relaxed) surrogate losses. Future directions include exploring how to integrate task-based criteria that involve a strong connection among multiple data points.

CHAPTER 5

CONCLUSION

In this thesis, I introduced novel frameworks for combining prior knowledge-based reasoning with the pattern recognition capabilities of deep learning via an interpretable latent space. The idea of the interpretable latent space was first introduced via a novel framework based on a multivariate probit model to tackle multi-label classification — in particular, deep-learning-based joint species distribution models (JSDMs). As a first example, we showed how we advanced the interpretability of deep learning via entity embeddings in the latent space in order to capture and interpret interactions among entities as well as the interactions between entities and the corresponding contextual information. We applied this method to the eBird citizen science project to uncover species interactions and habitat associations for the entire North American avifauna. Furthermore, we proposed the Deep Multivariate Probit Model (DMVP), which improved on the initial model by further incorporating prior knowledge of the low-rank covariance structure of entity interaction and accelerated the learning by an order of magnitude. DMVP provides a solid foundation for the learning of multi-entity interactions, and was later generalized to multi-target regression tasks for species abundance estimation [75] and multi-property prediction tasks [4, 76].

Next, we extended the idea of combining deep learning with prior knowledge reasoning to unsupervised tasks with rich prior knowledge, where we proposed a novel model called Deep Reasoning Networks (DRNets), which integrates pattern recognition with prior knowledge reasoning to perform unsupervised demixing tasks. As demonstrated on two challenging tasks, Multi-MNIST-Sudoku

and crystal-structure phase mapping, with the "self-supervision" from the prior knowledge reasoning, DRNets outperforms supervised state-of-the-art methods in an unsupervised manner and surpass previous approaches as well as the capability of experts in crystal-structure phase mapping, unraveling the Bi-Cu-V oxide phase diagram and enabling the discovery of solar-fuels materials. This is a collaboration with the Joint Center for Artificial Photosynthesis (JCAP) at Caltech. Our work was featured as the cover article in the journal *Nature Machine Intelligence*.

Interweaved with the main story, we also discussed our studies on reducing the data bias in citizen science projects and the task-based learning problems. The bias reduction work mitigates the covariate shift problem in citizen science projects to provide predictions that are more aligned with the scientific objectives. The task-based learning method incorporates task-based non-differentiable objectives, which often come from rule-based prior knowledge, into end-to-end learning, to learn a model that directly towards the ultimate goal.

In the future, we will further extend the concept of combining deep learning with prior knowledge and the concept of using an interpretable latent space to other tasks. For example, in our on-going project, we incorporate the chemical rules of how how molecular structure could possibly fragment into the graph neural network based deep learning framework to predict the experimental mass spectra from known chemical structures. This work has significantly outperformed existing rule-based methods and pure deep learning methods with respect to both the spectrum accuracy and the structure-level interpretability. More generally, research on incorporating neural-network-based learning with symbolic knowledge representation and logical reasoning is an important next

frontier in AI/ML research [26], and we expect that a variety of research projects will head in this direction in the future.

APPENDIX A

APPENDIX FOR CHAPTER 2

A.1 Appendix

Theorem 1 *Let $\mu \in R^l$ and $\Sigma \in R^{l\times l}$ be the rescaled mean and the rescaled residual covariance matrix of the random variable $w^{(k)}$ in the equation (2.19) of the main text, then we have*

$$\Pr\left[\left|\frac{1}{M}\sum_{k=1}^{M}\prod_{j=1}^{l}\Phi(w_{i,j}^{(k)}) - \Pr(y_i|x_i)\right| \geq \epsilon \Pr(y_i|x_i)\right]$$

$$\leq \frac{\Phi\left(0; \begin{bmatrix}-\mu\\ -\mu\end{bmatrix}, \begin{bmatrix}\Sigma+I & \Sigma\\ \Sigma & \Sigma+I\end{bmatrix}\right) - \Phi^2(0; -\mu, \Sigma+I)}{M\Phi^2(0; -\mu, \Sigma+I)\epsilon^2} \tag{A.1}$$

$$\leq \frac{\left(\frac{\Phi(0;-\mu,2\Sigma+I)}{\Phi(0;-\mu,\Sigma+I)}\right)^2 |2\Sigma+I|^{1/2} - 1}{M\epsilon^2} \tag{A.2}$$

$$\leq \frac{\prod_{i=1}^{l} g(\mu_i)^2 |2\Sigma+I|^{1/2} - 1}{M\epsilon^2} \tag{A.3}$$

where $g(\mu_i) = \max_x \frac{\Phi(\sqrt{2}x+\mu_i)}{\Phi(x+\mu_i)}$. The function $g(\mu_i)$ does not have a closed form but it is a monotonous decreasing function, which converges to 1 as μ_i increases.

Proof. For the ease of expression, we omit the subscripts related to i-th data point in our proof. Without loss of generality, we can also assume the diagonal matrix V is an indentity matrix. Defining $\Pr(y|w) = \prod_{j=1}^{n}\Phi(w_j)$, $\Pr(y|x) = E_{w\sim N(\mu,\Sigma)}[\Pr(y|w)]$. We prove this convergence bound by analysing the first and second moment of random variable $\Pr(y|w)$.

$$E_w[\Pr(y|w)] = \int_w \prod_{j=1}^{n}\Phi(w_j) Pr_w(w)\mathrm{d}w$$

$$
\begin{aligned}
&= \int_w Pr_z(z \preceq w|w) Pr_w(w) \mathrm{d}w \\
&= Pr_{z,w}(z \preceq w) \\
&= Pr_{z,w}(z - w \preceq 0) \qquad \text{(A.4)}
\end{aligned}
$$

Here z $N(0, I)$ and $a \preceq b$ means $\forall a_i \leq b_i$

Since z is subject to multivariate gaussian distribution, $z - w$ is still a multivariate gaussian random variable, which is subject to $N(-\mu, \Sigma + I)$. Thus, $\Pr(y|x) = E_w[\Pr(y|w)] = \Phi(0; -\mu, \Sigma + I)$. ($\Phi(\cdot)$ denotes the cumulative function of multivariate gaussian distribution.)

Similarly, we can derive that

$$
\begin{aligned}
E[\Pr(y|w)^2] &= \Pr(z_1 \preceq w \wedge z_2 \preceq w) \\
&= \Pr\left(\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \preceq \begin{bmatrix} r \\ r \end{bmatrix}\right) \\
&= \Phi\left(0; \begin{bmatrix} -\mu \\ -\mu \end{bmatrix}, \begin{bmatrix} \Sigma + I & \Sigma \\ \Sigma & \Sigma + I \end{bmatrix}\right)
\end{aligned}
$$

Let $B = \begin{bmatrix} \Sigma + I & \Sigma \\ \Sigma & \Sigma + I \end{bmatrix}$, we have $|B| = \left| det\left(\begin{bmatrix} 2\Sigma + I & \Sigma \\ 0 & I \end{bmatrix}\right) \right| = |2\Sigma + I|$. Since Σ is a positive definite matrix, we can decompose $\Sigma = UDU^T$, where U is an orthogonal matrix and D is a diagonal matrix. Similarly, we can decompose

$$
B^{-1} = \begin{bmatrix} U & 0 \\ 0 & U \end{bmatrix} \begin{bmatrix} (2D + I)^{-1}(D + I) & -(2D + I)^{-1}D \\ -(2D + I)^{-1}D & (2D + I)^{-1}(D + I) \end{bmatrix} \begin{bmatrix} U^T & 0 \\ 0 & U^T \end{bmatrix}
$$

Let $x_1, x_2 \in R^l$, $y_1 = U^T(x_1 + \mu)$, $y_2 = U^T(x_1 + \mu)$ and $D = diag(d_1, ..., d_l)$, then we have,

$$
E[\Pr(y|r)^2] = \Phi\left(0; \begin{bmatrix} -\mu \\ -\mu \end{bmatrix}, \begin{bmatrix} \Sigma + I & \Sigma \\ \Sigma & \Sigma + I \end{bmatrix}\right)
$$

$$
\begin{aligned}
&= \frac{1}{(2\pi)^l |B|^{1/2}} \int_{(-\infty,0]^l} e^{-\frac{1}{2}(\sum_{i=1}^{l}(y_{1,i}^2+y_{2,i}^2)\frac{d_i+1}{2d_i+1} - 2\sum_{i=1}^{l} y_{1,i}y_{2,i}\frac{d_i}{2d_i+1})} \mathrm{d}x_1 \mathrm{d}x_2 \\
&\leq \frac{1}{(2\pi)^l |B|^{1/2}} \int_{(-\infty,0]^l} e^{-\frac{1}{2}(\sum_{i=1}^{l}(y_{1,i}^2+y_{2,i}^2)\frac{1}{2d_i+1})} \mathrm{d}x_1 \mathrm{d}x_2 \\
&= |2\Sigma + I|^{1/2} \Phi\left(0; \begin{bmatrix} -\mu \\ -\mu \end{bmatrix}, \begin{bmatrix} 2\Sigma + I & 0 \\ 0 & 2\Sigma + I \end{bmatrix}\right)
\end{aligned}
$$

Thus,

$$
E[\Pr(y|r)^2]^{1/2} \leq |2\Sigma + I|^{1/4} \Phi(0; -\mu, 2\Sigma + I)
$$

Using the inverse transformation in equation (A.4), we have

$$
\begin{aligned}
&\Phi(0; -\mu, 2\Sigma + I) \\
&= \frac{1}{(2\pi)^{l/2}|2\Sigma|^{1/2}} \int \prod \Phi(x) e^{\frac{1}{4}(x-\mu)^T \Sigma^{-1}(x-\mu)} \mathrm{d}x \\
&= \frac{1}{(2\pi)^{l/2}|\Sigma|^{1/2}} \int \prod \Phi(\sqrt{2}y + \mu_i) e^{\frac{1}{2}y^T \Sigma^{-1} y} \mathrm{d}y
\end{aligned}
\tag{A.5}
$$

Let $g(\mu_i) = \max_x \frac{\Phi(\sqrt{2}x+\mu_i)}{\Phi(x+\mu_i)}$, then we have

$$
\begin{aligned}
&\Phi(0; -\mu, 2\Sigma + I) \\
&= \frac{1}{(2\pi)^{l/2}|\Sigma|^{1/2}} \int \prod \Phi(\sqrt{2}y + \mu) e^{\frac{1}{2}y^T \Sigma^{-1} y} \mathrm{d}y \\
&\leq \frac{\prod_{i=1}^{l} g(\mu_i)}{(2\pi)^{l/2}|\Sigma|^{1/2}} \int \prod \Phi(y + \mu) e^{\frac{1}{2}y^T \Sigma^{-1} y} \mathrm{d}y \\
&= \prod_{i=1}^{l} g(\mu_i) \Phi(\mu|\Sigma + I) \\
&= \prod_{i=1}^{l} g(\mu_i) \Pr(y|x)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
E[\Pr(y|w)^2]^{1/2} &\leq |2\Sigma + I|^{1/4} \Phi(0; -\mu, 2\Sigma + I) \\
&\leq |2\Sigma + I|^{1/4} \prod_{i=1}^{l} g(\mu_i) \Phi(0; -\mu, \Sigma + I)
\end{aligned}
$$

Using the Chebyshev's inequality, we have

$$
\begin{aligned}
&\Pr[|\frac{1}{M}\sum_{k=1}^{M}\prod_{j=1}^{l}\Phi(w_{i,j}^{(k)}) - \Pr(y_i|x_i)| \geq \epsilon \Pr(y_i|x_i)] \\
&= \Pr[|\frac{1}{M}\sum_{k=1}^{M}\Pr(y_i|w_i^{(k)}) - \Pr(y_i|x_i)| \geq \epsilon \Pr(y_i|x_i)] \\
&= \Pr[|\frac{1}{M}\sum_{k=1}^{M}\Pr(y_i|w_i^{(k)}) - \Pr(y_i|x_i)|^2 \geq \epsilon^2 \Pr(y_i|x_i)^2] \\
&\leq \frac{E[(\frac{1}{M}\sum_{k=1}^{M}\Pr(y_i|w_i^{(k)}) - \Pr(y_i|x_i))^2]}{\epsilon^2 \Pr(y_i|x_i)^2} \\
&= \frac{\prod_{i=1}^{l} g^2(\mu_i)|2\Sigma + I|^{1/2} - 1}{M\epsilon^2} \qquad \blacksquare
\end{aligned}
$$

The function $g(\mu_i)$ does not have a closed form but it is a monotonous decreasing function, which converges to 1 as μ_i increases. The figure (A.1) is the

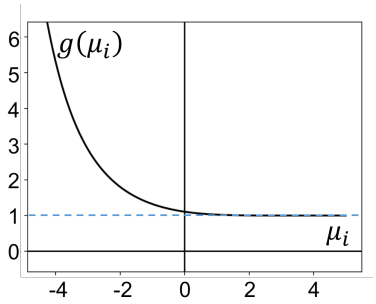


Figure A.1: The visualization of function $g(\mu_i)$.

visualization of function $g(\mu_i)$. As you see, the function $g(\mu_i)$ is very close to 1 when μ_i is positive. The following lemma provides a more analytical upper bound for function $g(\mu_i)$.

Lemma 1 *For any y,* $\Phi(\sqrt{2}y + \mu) \leq g(\mu)\Phi(y + \mu)$*, where*

$$
g(\mu) \leq \begin{cases} \sqrt{2}e^{\frac{3-2\sqrt{2}}{2}\mu^2} & \text{if} \quad \mu < 0 \\ 1.182 & \text{if} \quad \mu \geq 0 \end{cases}
$$

Proof. $\frac{\Phi(\sqrt{2}y+\mu)}{\Phi(y+\mu)}$ achieves the maximum when its derivative is equal to zero, i.e.,

$$\left(\frac{\Phi(\sqrt{2}y+\mu)}{\Phi(y+\mu)}\right)' = 0 \Longrightarrow$$
$$\frac{\frac{1}{\sqrt{2\pi}}(\sqrt{2}e^{-\frac{1}{2}(\sqrt{2}y+\mu)^2}\Phi(y+\mu) - e^{-\frac{1}{2}(y+\mu)^2}\Phi(\sqrt{2}y+\mu)}{\Phi^2(y+\mu)} = 0$$
$$\Longrightarrow \frac{\Phi(\sqrt{2}y+\mu)}{\Phi(y+\mu)} = \sqrt{2}e^{-\frac{1}{2}(y^2+2(\sqrt{2}-1)\mu y)}$$

Since $\Phi(x)$ is a monotonic increasing function, $max_y \sqrt{2}e^{-\frac{1}{2}(y^2+2(\sqrt{2}-1)\mu y)} = \sqrt{2}e^{\frac{3-2\sqrt{2}}{2}\mu^2}$ when $\mu < 0$. Similarly, when $\mu \geq 0$, we know $y^* = argmax_y \frac{\Phi(\sqrt{2}y+\mu)}{\Phi(y+\mu)} \geq 0$. Thus, $\Phi(y^*+\mu) \geq \frac{1}{2}$. By analysing the maximal value of $\Phi(\sqrt{2}y+\mu) - \Phi(y+\mu)$ as well as the fact that $\Phi(\sqrt{2}y+\mu) - \Phi(y+\mu) \leq (\sqrt{2}-1)y * \frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(y+\mu)^2}$, we could know that $\Phi(\sqrt{2}y+\mu) - \Phi(y+\mu) \leq 0.091$. That is,

$$g(\mu) \leq \begin{cases} \sqrt{2}e^{\frac{3-2\sqrt{2}}{2}\mu^2} & \text{if} \quad \mu < 0 \\ 1.182 & \text{if} \quad \mu \geq 0 \end{cases}$$

.

Theorem 2 *Let $\mu \in R^l$ and $\Sigma \in R^{l\times l}$ be the rescaled mean and rescaled residual covariance matrix of the random variable $w^{(k)}$ in equation (2.19) of the main text, we have*

$$\Pr\left[\left|\frac{\partial \frac{1}{M}\sum_{k=1}^{M}\prod_{j=1}^{l}\Phi(w_{i,j}^k)}{\partial\mu_i} - \frac{\partial \Pr(y_i|x_i)}{\partial\mu_i}\right| \geq \epsilon\frac{\partial \Pr(y_i|x_i)}{\partial\mu_i}\right]$$
$$\leq \frac{e^{\frac{\mu_i^2}{2(\Sigma_{i,i}+1)}}(\Sigma_{i,i}+1)\lambda_{max}\prod_{j\neq i}^{l} g(\mu_j')^2|2\Sigma+I|^{1/2}-1}{M\epsilon^2} \tag{A.6}$$

Here λ_{max} denotes the largest eigenvalue of Σ and $\mu' = \mu - \frac{\mu_i}{\nu+1}\Sigma^{1/2}b_i$. ($b_i$ denotes the i-th row of $\Sigma_{1/2}$.)

Proof. For the ease of symbolism, we omit all the subscript i related to the index of i-th data point. For any $1 \leq i \leq l$,

$$\frac{\partial \Pr(y|x)}{\partial\mu_i} = E_{w\sim N(\mu,\Sigma)}\left[\frac{\partial\prod_{j=1}^{l}\Phi(w_j)}{\partial\mu_i}\right]$$

$$= \int \prod_{j \neq i}^{l} \Phi(w_j) * \phi(w_i)\phi(w|\mu, \Sigma)\mathrm{d}w$$

$$= \int \prod_{j \neq i}^{l} \Phi(\Sigma_j^{1/2}x + \mu_j) * \phi(\Sigma_i^{1/2}x + \mu_i)\phi(x|0, I)\mathrm{d}x$$

Let $B = \Sigma^{1/2}$ and let b_j denote the j-th row of B.

$$= \int \prod_{j \neq i}^{l} \Phi(b_j^T x + \mu_j) * \phi(b_i^T x + \mu_i)\phi(x|0, I)\mathrm{d}x$$

let $v = b_i^T b_i = \Sigma_{i,i}$ and $C = I - \frac{b_i b_i^T}{v+1}$ ($C^{-1} = I + b_i b_i^T$).

$$= \phi(\frac{\mu_i}{v+1}) * |C|^{1/2} \int \prod_{j \neq i}^{l} \Phi(b_j^T x + \mu_j) * \phi(x| - \frac{\mu_i}{v+1}b_i, C)\mathrm{d}x$$

$$= \phi(\frac{\mu_i}{v+1}) * |C|^{1/2} * \Pr(\forall j \neq i, z_j \leq b_j^T x + \mu_j)$$

(where $x \sim N(-\frac{\mu_i}{v+1}b_i, C)$ and $z \sim N(0, I)$.)

$$= \phi(\frac{\mu_i}{v+1}) * |C|^{1/2} * \Pr(z \leq w)$$

(where $w \sim N(\mu_{-i} - \frac{\mu_i}{v+1}B_{-i}b_i, B_{-i}CB_{-i}^T)$,

$\mu_{-i} \in R^{l-1}$ denotes the vector derived from μ by

eliminating the i-th entry. $B_{-i} \in R^{l-1 \times l}$ denotes the

matrix derived from B by eliminating the i-th row.)

Thus, using the transformation above, we can transform the derivative in terms of μ_i into the form similar to theorem (1). Because $B_{-i}CB_{-i}^T = B_{-i}B_{-i}^T - \frac{(B_{-i}b_i)(B_{-i}b_i)^T}{v+1}$, where $B_{-i}B_{-i}^T$ is a principal submatrix of Σ, whose eigenvalues are interlaced with the eigenvalues of Σ, and $\frac{(B_{-i}b_i)(B_{-i}b_i)^T}{v+1}$ is a rank-1 matrix, we have $|2B_{-i}CB_{-i}^T + I| \leq |2\Sigma + I| * \lambda_{max}$.

In terms of the second moment of the derivative of μ_i, we have,

$$E_{w \sim N(\mu,\Sigma)}\left[\left(\frac{\partial \prod_{j=1}^{l} \Phi(w_j)}{\partial \mu_i}\right)^2\right]$$

$$
\begin{aligned}
&= \int \prod_{j\neq i}^{l} \Phi^2(\Sigma_j^{1/2}x + \mu_j) * \phi^2(\Sigma_i^{1/2}x + \mu_i)\phi(x|0, I)\mathrm{d}x \\
&\leq \int \prod_{j\neq i}^{l} \Phi^2(\Sigma_j^{1/2}x + \mu_j) * \phi(\Sigma_i^{1/2}x + \mu_i)\phi(x|0, I)\mathrm{d}x \\
&= \phi(\frac{\mu_i}{v+1}) * |C|^{1/2} \int \prod_{j\neq i}^{l} \Phi^2(b_j^T x + \mu_j) * \phi(x| - \frac{\mu_i}{v+1}b_i, C)\mathrm{d}x \\
&= \phi(\frac{\mu_i}{v+1}) * |C|^{1/2} * \Pr(z^1 \preceq w \wedge z^2 \preceq w)
\end{aligned}
$$

Here we use the same notation as the proof above.

Using the similar trick as theorem (1), we have

$$
\begin{aligned}
&\Pr\left[\left|\frac{\partial \frac{1}{M}\sum_{k=1}^{M}\prod_{j=1}^{l}\Phi(w_{i,j}^k)}{\partial\mu_i} - \frac{\partial \Pr(y_i|x_i)}{\partial\mu_i}\right| \geq \epsilon\frac{\partial \Pr(y_i|x_i)}{\partial\mu_i}\right] \\
&\leq \frac{e^{\frac{\mu_i^2}{2(v+1)}}|C^{-1}|\lambda_{max}\prod_{j\neq i}^{l} g(\mu_j')^2|2\Sigma + I|^{1/2} - 1}{M\epsilon^2} \\
&\leq \frac{e^{\frac{\mu_i^2}{2(\Sigma_{i,i}+1)}}(\Sigma_{i,i}+1)\lambda_{max}\prod_{j\neq i}^{l} g(\mu_j')^2|2\Sigma + I|^{1/2} - 1}{M\epsilon^2}
\end{aligned}
$$

Here $\mu' = \mu - \frac{\mu_i}{v+1}\Sigma^{1/2}b_i$.

In this way, we bound the convergence of the derivatives in terms of μ, so that the derivatives in term of the parameters in feature network can be derived by chain rule. However, because the derivatives of $\Sigma^{1/2}$ could be negative or zero, we can not apply the Chebyshev's inequality to have a similar multiplicative error bound. Nevertheless, because all the data points share a global residual covariance matrix, empirical experiments show that $\Sigma^{1/2}$ converges well on all the datasets.

Here we show that the variance of our sampling process is strictly lower than the rejection sampling.

Theorem 3 *Here we follow the notation of equation(2.19) in the main paper. Let* θ_1 *be*

the reject sampling estimator of $\Phi(0; -\mu, \Sigma)$*, where* $E[\theta_1] = E_{r\sim N(0,\Sigma)}[I\{r \leqslant \mu\}]$*. Let* θ_2 *be the estimator of DMVP's sampling process, where* $E[\theta_2] = E_{w\sim N(0,\Sigma_r)}[\Pr(z \leqslant (w+\mu)|w)]$ *and* $z \sim N(0, V)$*. We have* $Var[\theta_2] < Var[\theta_1]$*.*

Proof.

$$
\begin{aligned}
Var[\theta_2] &= E[(\theta_2 - E[\theta_2])^2] \\
&= E_{w\sim N(0,\Sigma_r)}[(\Pr(z \leqslant (w+\mu)|w) - E[\theta_2])^2] \\
&= E_{w\sim N(0,\Sigma_r)}[(E_{z\sim N(0,V)}[I\{z \leqslant (w+\mu)\} - E[\theta_2]|w])^2] \\
&< E_{w\sim N(0,\Sigma_r)}[E_{z\sim N(0,V)}[(I\{z \leqslant (w+\mu)\} - E[\theta_2])^2|w]] \\
&= E_{r\sim N(0,\Sigma)}[(I\{r \leqslant \mu\} - E[\theta_1])^2] \\
&\qquad \text{(Here } r = z - w \text{ and } E[\theta_1] = E[\theta_2]\text{)} \\
&= E[(\theta_1 - E[\theta_1])^2] = Var[\theta_1]
\end{aligned}
$$

The inequality follows the fact that $E[x^2] > E[x]^2$ *given* $Var[x] \neq 0$*.*

BIBLIOGRAPHY

[1] Artificial intelligence. Science, 349, 2015.

[2] Sebastian E Ament, Helge S Stein, Dan Guevarra, Lan Zhou, Joel A Haber, David A Boyd, Mitsutaro Umehara, John M Gregoire, and Carla P Gomes. Multi-component background learning automates signal detection for spectroscopic data. *npj Computational Materials*, 5(1):1–7, 2019.

[3] JR Ashford and RR Sowden. Multi-variate probit analysis. *Biometrics*, pages 535–546, 1970.

[4] Junwen Bai, Shufeng Kong, and Carla Gomes. Disentangled variational autoencoder based multi-label classification with covariance-aware multivariate probit model. *IJCAI*, 2020.

[5] Philip Ball. Learning from the big picture. *Nature materials*, 17(12):1062–1062, 2018.

[6] David Belanger and Andrew McCallum. Structured prediction energy networks. In *International Conference on Machine Learning*, pages 983–992, 2016.

[7] David Belanger, Bishan Yang, and Andrew McCallum. End-to-end learning for structured prediction energy networks. In *International Conference on Machine Learning*, pages 429–439. PMLR, 2017.

[8] Yoshua Bengio. Using a financial training criterion rather than a prediction criterion. *International Journal of Neural Systems*, 8(04):433–443, 1997.

[9] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *journal of machine learning research*, 3(Feb), 2003.

[10] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10(Sep):2137–2155, 2009.

[11] David Blackwell. Conditional expectation and unbiased sequential estimation. *The Annals of Mathematical Statistics*, pages 105–110, 1947.

[12] Rick Bonney, Caren B Cooper, Janis Dickinson, Steve Kelling, Tina Phillips, Kenneth V Rosenberg, and Jennifer Shirk. Citizen science: a developing tool for expanding science knowledge and scientific literacy. *BioScience*, 59(11):977–984, 2009.

[13] Léon Bottou. Stochastic learning. In *Advanced lectures on machine learning*, pages 146–168. Springer, 2004.

[14] Jonathan Kenneth Bunn, Shizhong Han, Yan Zhang, Yan Tong, Jianjun Hu, and Jason R. Hattrick-Simpers. Generalized machine learning technique for automatic phase attribution in time variant high-throughput experimental studies. *Journal of Materials Research*, 30(7):879–889, April 2015.

[15] Di Chen, Yiwei Bai, Sebastian Ament, Wenting Zhao, Dan Guevarra, Lan Zhou, Bart Selman, R Bruce van Dover, John M Gregoire, and Carla P Gomes. Automating crystal-structure phase mapping by combining deep learning with constraint reasoning. *Nature Machine Intelligence*, 3(9):812–822, 2021.

[16] Di Chen, Yiwei Bai, Wenting Zhao, Sebastian Ament, John Gregoire, and Carla Gomes. Deep reasoning networks for unsupervised pattern demixing with constraint reasoning. In *International Conference on Machine Learning*, pages 1500–1509. PMLR, 2020.

[17] Di Chen, Yiwei Bai, Wenting Zhao, Sebastian Ament, John Gregoire, and Carla Gomes. Deep reasoning networks for unsupervised pattern de-mixingwith constraint reasoning. In *Proceedings of the 37th international conference on machine learning (ICML-2020)*, 2020.

[18] Di Chen, Yexiang Xue, Shuo Chen, Daniel Fink, and Carla Gomes. Deep multi-species embedding. In *IJCAI*, 2017.

[19] Di Chen, Yexiang Xue, and Carla Gomes. End-to-end learning for the deep multivariate probit model. In *International Conference on Machine Learning*, pages 932–941. PMLR, 2018.

[20] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012.

[21] Siddhartha Chib and Edward Greenberg. Analysis of multivariate probit models. *Biometrika*, 85(2):347–361, 1998.

[22] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yan-Tao Zheng. Nus-wide: A real-world web image database from national university of singapore. In *Proc. of ACM Conf. on Image and Video Retrieval (CIVR'09)*, Santorini, Greece., July 8-10, 2009.

[23] William J Cody. Rational chebyshev approximations for the error function. *Mathematics of Computation*, 23(107):631–637, 1969.

[24] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[25] US NABCI Committee et al. North american bird conservation initiative: Bird conservation region descriptions, a supplement to the north american bird conservation initiative bird conservation regions map. 2000.

[26] Artur d'Avila Garcez and Luis C. Lamb. Neurosymbolic AI: The 3rd wave, 2020.

[27] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-scale object classification using label relation graphs. In *ECCV*, 2014.

[28] Priya Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 5484–5494, 2017.

[29] M. Doumpos, C. Lemonakis, D. Niklis, and C. Zopounidis. Introduction to credit risk modeling and assessment. In *In: Analytical Techniques in the Assessment of Credit Risk. EURO Advanced Tutorials on Operational Research*, pages 1–21. Springer, Cham, 2019.

[30] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 2011.

[31] Doreen Edwards. Chapter Ten - Phase Diagram Determination of Ceramic Systems. In J. C. Zhao, editor, *Methods for Phase Diagram Determination*, pages 341–360. Elsevier Science Ltd, Oxford, January 2007.

[32] André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*, pages 681–687, 2002.

[33] Jane Elith and John R Leathwick. Species distribution models: ecological explanation and prediction across space and time. *Annual Review of Ecology, Evolution, and Systematics*, 40(1):677, 2009.

[34] Adam N Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management Science*, 2021.

[35] V. Elser, I. Rankenburg, and P. Thibault. Searching with iterated maps. *Proceedings of the National Academy of Sciences*, 104(2):418–423, January 2007. Publisher: National Academy of Sciences Section: Physical Sciences.

[36] Daniel Fink, Theodoros Damoulas, and Jaimin Dave. Adaptive spatio-temporal exploratory models: Hemisphere-wide species distributions from massively crowdsourced ebird data. In *AAAI*, 2013.

[37] William Fithian, Jane Elith, Trevor Hastie, and David A Keith. Bias correction in species distribution models: pooling survey and collection data for multiple species. *Methods in Ecology and Evolution*, 2015.

[38] William Fithian and Trevor Hastie. Finite-sample equivalence in statistical models for presence-only data. *The annals of applied statistics*, 7(4):1917, 2013.

[39] Alan Genz. Numerical computation of multivariate normal probabilities. *Journal of computational and graphical statistics*, 1992.

[40] Yolanda Gil, Mark Greaves, James Hendler, and Haym Hirsh. Amplify scientific discovery with artificial intelligence. *Science*, 346(6206):171–172, 2014.

[41] Carla Gomes, Thomas Dietterich, Christopher Barrett, Jon Conrad, Bistra Dilkina, Stefano Ermon, Fei Fang, Andrew Farnsworth, Alan Fern, Xiaoli

Fern, et al. Computational sustainability: Computing for a better world and a sustainable future. *Communications of the ACM*, 62(9):56–65, 2019.

[42] Carla P Gomes. Computational sustainability: Computational methods for a sustainable environment, economy, and society. *The Bridge*, 39(4):5–13, 2009.

[43] Carla P Gomes, Junwen Bai, Yexiang Xue, Johan Björck, Brendan Rappazzo, Sebastian Ament, Richard Bernstein, Shufeng Kong, Santosh K Suram, R Bruce van Dover, et al. Crystal: a multi-agent ai system for automated mapping of materials' crystal structures. *MRS Communications*, pages 1–9, 2019.

[44] Carla P Gomes, Bart Selman, Henry Kautz, et al. Boosting combinatorial search through randomization. *AAAI/IAAI*, 98:431–437, 1998.

[45] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[46] Simon Gravel and Veit Elser. Divide and concur: A general approach to constraint satisfaction. *Physical Review E*, 78(3):036706, September 2008.

[47] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[48] Martin L Green, CL Choi, JR Hattrick-Simpers, AM Joshi, I Takeuchi, SC Barron, E Campo, T Chiang, S Empedocles, JM Gregoire, et al. Fulfill-

ing the promise of the materials genome initiative with high-throughput experimental methodologies. *Applied Physics Reviews*, 4(1):011105, 2017.

[49] JM Gregoire, DG Van Campen, CE Miller, RJR Jones, SK Suram, and A Mehta. High-throughput synchrotron x-ray diffraction for combinatorial phase mapping. *Journal of synchrotron radiation*, 21(6):1262–1268, 2014.

[50] Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten Borgwardt, and Bernhard Schölkopf. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

[51] Gurutzeta Guillera-Arroita, José J Lahoz-Monfort, Jane Elith, Ascelin Gordon, Heini Kujala, Pia E Lentini, Michael A McCarthy, Reid Tingley, and Brendan A Wintle. Is my species distribution model fit for purpose? matching data and models to applications. *Global Ecology and Biogeography*, 2015.

[52] David J Harris. Generating realistic assemblages with a joint species distribution model. *Methods in Ecology and Evolution*, 6(4):465–473, 2015.

[53] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[54] James J Heckman. Sample selection bias as a specification error (with an application to the estimation of labor supply functions), 1977.

[55] Sergi Herrando, Verena Keller, Petr Voříšek, Marina Kipson, Martí Franch, Marc Anton, Magda Pla, Dani Villero, Henk Sierdsema, Christian Kampichler, et al. High resolution maps for the second european breeding bird atlas: A first provision of standardised data and pilot modelled maps. *Vogelwelt*, 2017.

[56] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[57] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. PMLR, 2018.

[58] Collin Homer, Jon Dewitz, Limin Yang, Suming Jin, Patrick Danielson, George Xian, John Coulston, Nathaniel Herold, James Wickham, and Kevin Megown. Completion of the 2011 national land cover database for the conterminous united states–representing a decade of land cover change information. *Photogrammetric Engineering & Remote Sensing*, 2015.

[59] Jon S Horne, Edward O Garton, Stephen M Krone, and Jesse S Lewis. Analyzing animal movements using brownian bridges. *Ecology*, 88(9):2354–2363, 2007.

[60] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420, Berlin, Germany, August 2016. Association for Computational Linguistics.

[61] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, volume 1, page 3, 2017.

[62] Jiayuan Huang, Arthur Gretton, Karsten M Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by un-

labeled data. In *Advances in neural information processing systems*, pages 601–608, 2007.

[63] Rebecca A. Hutchinson, Li-Ping Liu, and Thomas G. Dietterich. Incorporating boosted regression trees into ecological latent variable models. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[64] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[65] Narasimhan Jegadeesh and Joshua Livnat. Revenue surprises and stock returns. *Journal of Accounting and Economics*, 41, 2006.

[66] Ivan Jeliazkov and Esther Hee Lee. Mcmc perspectives on simulated likelihood estimation. In *Maximum simulated likelihood methods and applications*, pages 3–39. Emerald Group Publishing Limited, 2010.

[67] Shuiwang Ji and Jieping Ye. Linear dimensionality reduction for multi-label classification. In *IJCAI*, 2009.

[68] Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 264–271, 2007.

[69] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[70] Julia Jones, Jeffrey Miller, and Matt White. Multi-label classification for species distribution modeling. 2011.

[71] Yu Jun, Weng-Keen Wong, Tom Dietterich, Julia Jones, Matthew Betts, Sarah Frey, Susan Shirley, Jeffrey Miller, and Matt White. Multi-label classification for species distribution. In *Proceedings of the ICML 2011 Workshop on Machine Learning for Global Challenges*, 2011.

[72] Daniel Kahneman. *Thinking, fast and slow*. Macmillan, 2011.

[73] Yi-hao Kao, Benjamin V Roy, and Xiang Yan. Directed regression. In *Advances in Neural Information Processing Systems*, pages 889–897, 2009.

[74] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015.

[75] Shufeng Kong, Junwen Bai, Jae Hee Lee, Di Chen, Andrew Allyn, Michelle Stuart, Malin Pinsky, Katherine Mills, and Carla P Gomes. Deep hurdle networks for zero-inflated multi-target regression: Application to multiple species abundance estimation. *IJCAI*, 2020.

[76] Shufeng Kong, Dan Guevarra, Carla P. Gomes, and John M. Gregoire. Materials representation and transfer learning for multi-property prediction. Applied Physics Reviews, https://doi.org/10.1063/5.0047066, 2021.

[77] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[78] A. Gilad Kusne, Heshan Yu, Changming Wu, Huairuo Zhang, Jason Hattrick-Simpers, Brian DeCost, Suchismita Sarker, Corey Oses, Cormac Toher, Stefano Curtarolo, Albert V. Davydov, Ritesh Agarwal, Leonid A. Bendersky, Mo Li, Apurva Mehta, and Ichiro Takeuchi. On-the-fly closed-

loop materials discovery via Bayesian active learning. *Nature Communications*, 11(1):5966, November 2020.

[79] Aaron G Kusne, D Keller, A Anderson, A Zaban, and I Takeuchi. High-throughput determination of structural phase diagram and constituent phases using grendel. *Nanotechnology*, 26(44):444002, 2015.

[80] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, 2001.

[81] Laodar. A tensorflow implementation for capsnet, 2017.

[82] Maxim Larrivée, Kathleen L Prudic, Kent McFarland, and J Kerr. ebutterfly: a citizen-based butterfly database in the biological sciences, 2014.

[83] Ronan Le Bras, Richard Bernstein, John M Gregoire, Santosh K Suram, Carla P Gomes, Bart Selman, and R Bruce Van Dover. Challenges in materials discovery–synthetic generator and real datasets. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[84] Ronan LeBras, Theodoros Damoulas, John M Gregoire, Ashish Sabharwal, Carla P Gomes, and R Bruce Van Dover. Constraint reasoning and kernel clustering for pattern decomposition with scaling. In *International Conference on Principles and Practice of Constraint Programming*, pages 508–522. Springer, 2011.

[85] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[86] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[87] Jin-Woong Lee, Woon Bae Park, Jin Hee Lee, Satendra Pal Singh, and Kee-Sun Sohn. A deep-learning technique for phase identification in multiphase inorganic compounds using synthetic xrd powder patterns. *Nature Communications*, 11(1):1–11, 2020.

[88] Erik Linder-Noren. Pytorch-gan, 2019.

[89] Bruce G Lindsay. Mixture models: theory, geometry and applications. In *NSF-CBMS regional conference series in probability and statistics*, pages i–163. JSTOR, 1995.

[90] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, pages 3122–3130. PMLR, 2018.

[91] CJ Long, D Bunker, X Li, VL Karen, and I Takeuchi. Rapid identification of structural phases in combinatorial thin-film libraries using x-ray diffraction and non-negative matrix factorization. *Review of Scientific Instruments*, 80(10):103902, 2009.

[92] Alfred Ludwig. Discovery of new materials using combinatorial synthesis and high-throughput characterization of thin-film materials libraries combined with computational methods. *npj Computational Materials*, 5(1):70, 2019.

[93] Jared K Lunceford and Marie Davidian. Stratification and weighting via the

propensity score in estimation of causal treatment effects: a comparative study. *Statistics in medicine*, 23(19):2937–2960, 2004.

[94] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[95] Darryl I MacKenzie, Larissa L Bailey, James Nichols, et al. Investigating species co-occurrence patterns when species are detected imperfectly. *Journal of Animal Ecology*, pages 546–555, 2004.

[96] D.I. MacKenzie. *Occupancy Estimation And Modeling: Inferring Patterns And Dynamics of Species Occurrence*. Elsevier Science & Tech, 2006.

[97] Andy R Magid. *Lectures on differential Galois theory*. American Mathematical Soc., 1994.

[98] Stephan Mandt, Florian Wenzel, Shinichi Nakajima, John Cunningham, Christoph Lippert, and Marius Kloft. Sparse probit linear mixed model. *Machine Learning*, 106(9-10):1621–1642, 2017.

[99] Charles F Manski and Steven R Lerman. The estimation of choice probabilities from choice based samples. *Econometrica: Journal of the Econometric Society*, pages 1977–1988, 1977.

[100] Peter McCullagh. Generalized linear models. *European Journal of Operational Research*, 1984.

[101] Daniel McFadden and Kenneth Train. Mixed mnl models for discrete response. *Journal of applied Econometrics*, pages 447–470, 2000.

[102] Brad H. McRae, Brett G. Dickson, Timothy H. Keitt, and Viral B. Shah.

Using circuit theory to model connectivity in ecology, evolution, and conservation. *Ecology*, 2008.

[103] T Mikolov and J Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 2013.

[104] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[105] M. Arthur Munson, Kevin Webb, Daniel Sheldon, Daniel Fink, Wesley M Hochachka, Marshall Iliff, Mirek Riedewald, Dara Sorokina, Brian Sullivant, Christopher Wood, and Steve Kelling. The eBird Reference Dataset, Version 4.0, 2012.

[106] M Arthur Munson, Kevin Webb, Daniel Sheldon, Daniel Fink, Wesley M Hochachka, Marshall Iliff, Mirek Riedewald, Daria Sorokina, Brian Sullivan, Christopher Wood, et al. The ebird reference dataset, version 4.0. 2012.

[107] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[108] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. Large-scale multi-label text classification—revisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 437–452. Springer, 2014.

[109] Felipe Oviedo, Zekun Ren, Shijing Sun, Charles Settens, Zhe Liu, Noor Titan Putri Hartono, Savitha Ramasamy, Brian L DeCost, Siyu IP Tian,

Giuseppe Romano, et al. Fast and interpretable classification of small x-ray diffraction datasets using data augmentation and deep neural networks. *npj Computational Materials*, 5(1):1–9, 2019.

[110] W. B. Park, J. Chung, J. Jung, K. Sohn, S. P. Singh, M. Pyo, N. Shin, and K.-S. Sohn. Classification of crystal structure using a convolutional neural network. *IUCrJ*, 4(4):486–494, July 2017.

[111] Andrew Perrault, Bryan Wilder, Eric Ewing, Aditya Mate, Bistra Dilkina, and Milind Tambe. Decision-focused learning of adversary behavior in security games. *IJCAI 2019 Workshop*, 2019.

[112] Steven J Phillips, Miroslav Dudík, Jane Elith, Catherine H Graham, Anthony Lehmann, John Leathwick, and Simon Ferrier. Sample selection bias and presence-only distribution models: implications for background and pseudo-absence data. *Ecological applications*, 19(1):181–197, 2009.

[113] Steven J. Phillips, Miroslav Dudík, and Robert E. Schapire. A maximum entropy approach to species distribution modeling. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 83–, New York, NY, USA, 2004. ACM.

[114] Laura J Pollock, Reid Tingley, William K Morris, Nick Golding, Robert B O'Hara, Kirsten M Parris, Peter A Vesk, and Michael A McCarthy. Understanding co-occurrence by modelling species simultaneously with a joint species distribution model (jsdm). *Methods in Ecology and Evolution*, 5(5):397–406, 2014.

[115] Marcos Lopez de Prado. *Advances in Financial Machine Learning*. Wiley, 2018.

[116] Jesse Read, Luca Martino, Pablo M Olmos, and David Luengo. Scalable multi-output label prediction: From classifier chains to classifier trellises. *Pattern Recognition*, 2015.

[117] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820. ACM, 2010.

[118] Matthew Riemer, Aditya Vempaty, Flavio P. Calmon, Fenno F. Heath, III, Richard Hull, and Elham Khabiri. Correcting forecasts with multifactor neural attention. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 3010–3019. JMLR.org, 2016.

[119] Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

[120] Francesca Rossi, Peter Van Beek, and Toby Walsh. *Handbook of constraint programming*. Elsevier, 2006.

[121] David Rossouw, Pierre Burdet, Francisco de la Pena, Caterina Ducati, Benjamin R Knappett, Andrew EH Wheatley, and Paul A Midgley. Multicomponent signal unmixing from nanoheterostructures:overcoming the traditional challenges of nanoscale x-ray analysis via machine learning. *Nano letters*, 15(4):2716–2720, 2015.

[122] Maja Rudolph, Francisco Ruiz, Stephan Mandt, and David Blei. Expo-

nential family embeddings. In *Advances in Neural Information Processing Systems*, 2016.

[123] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017.

[124] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, 2018.

[125] Sunita Sarawagi and Rahul Gupta. Accurate max-margin training for structured output spaces. In *Proceedings of the 25th international conference on Machine learning*, pages 888–895. ACM, 2008.

[126] Jan Seibert, Barbara Strobl, Simon Etter, Marc Vis, Tracy Ewen, and HJ van Meerveld. Engaging the public in hydrological observations-first experiences from the crowdwater project. In *EGU General Assembly Conference Abstracts*, volume 19, page 11592, 2017.

[127] Daniel R Sheldon and Thomas G Dietterich. Collective graphical models. In *Advances in Neural Information Processing Systems*, pages 1161–1169, 2011.

[128] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

[129] Hiroyuki Shinnou, Minoru Sasaki, and Kanako Komiya. Learning under covariate shift for domain adaptation for word sense disambiguation. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation: Posters*, pages 215–223, 2015.

[130] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

[131] John F. Smith. Chapter One - Introduction to Phase Diagrams. In J. C. Zhao, editor, *Methods for Phase Diagram Determination*, pages 1–21. Elsevier Science Ltd, Oxford, January 2007.

[132] Xin-Yuan Song and Sik-Yum Lee. A multivariate probit latent variable model for analyzing dichotomous responses. *Statistica Sinica*, pages 645–664, 2005.

[133] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

[134] Jelena Stajic, Richard Stone, Gilbert Chin, and Brad Wible. Rise of the Machines. *Science*, 349(6245):248–249, July 2015. Publisher: American Association for the Advancement of Science Section: Introduction to special issue.

[135] Valentin Stanev, Velimir V Vesselinov, A Gilad Kusne, Graham Antoszewski, Ichiro Takeuchi, and Boian S Alexandrov. Unsupervised phase mapping of x-ray diffraction data by nonnegative matrix factorization integrated with custom clustering. *npj Computational Materials*, 4(1):43, 2018.

[136] Ingo Steinwart. Support vector machines are universally consistent. *Journal of Complexity*, 18(3):768–791, 2002.

[137] Masashi Sugiyama and Klaus-Robert Müller. Input-dependent estimation of generalization error under covariate shift. *Statistics & Decisions*, 2005.

[138] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul V Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems*, pages 1433–1440, 2008.

[139] Brian L Sullivan, Jocelyn L Aycrigg, Jessie H Barry, Rick E Bonney, Nicholas Bruns, Caren B Cooper, Theo Damoulas, André A Dhondt, Tom Dietterich, Andrew Farnsworth, et al. The ebird enterprise: an integrated approach to development and application of citizen science. *Biological Conservation*, 169:31–40, 2014.

[140] Shijing Sun, Noor T. P. Hartono, Zekun D. Ren, Felipe Oviedo, Antonio M. Buscemi, Mariya Layurova, De Xin Chen, Tofunmi Ogunfunmi, Janak Thapa, Savitha Ramasamy, Charles Settens, Brian L. DeCost, Aaron G. Kusne, Zhe Liu, Siyu I. P. Tian, Ian Marius Peters, Juan-Pablo Correa-Baena, and Tonio Buonassisi. Accelerated Development of Perovskite-Inspired Materials via High-Throughput Synthesis and Machine-Learning Diagnosis. *Joule*, 3(6):1437–1451, June 2019.

[141] Santosh K Suram, Lan Zhou, Natalie Becerra-Stasiewicz, Kevin Kan, Ryan JR Jones, Brian M Kendrick, and John M Gregoire. Combinatorial thin film composition mapping using three dimensional deposition profiles. *Review of Scientific Instruments*, 86(3):033904, 2015.

[142] William J Sutherland, Steven Broad, Stuart HM Butchart, Stewart J Clarke, Alexandra M Collins, Lynn V Dicks, Helen Doran, Nafeesa Esmail, Erica

Fleishman, Nicola Frost, et al. A horizon scan of emerging issues for global conservation in 2019. *Trends in ecology & evolution*, 2018.

[143] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[144] Aline Tabet. *Bayesian inference in the multivariate probit model*. PhD thesis, University of British Columbia, 2007.

[145] Daniel P Tabor, Loïc M Roch, Semion K Saikin, Christoph Kreisbeck, Dennis Sheberla, Joseph H Montoya, Shyam Dwaraknath, Muratahan Aykol, Carlos Ortiz, Hermann Tribukait, et al. Accelerating the discovery of materials for clean energy in the era of smart automation. *Nature Reviews Materials*, 3(5):5–20, 2018.

[146] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[147] Luming Tang, Yexiang Xue, Di Chen, and Carla Gomes. Multi-entity dependence learning with rich context via conditional variational auto-encoder. In *AAAI*, 2017.

[148] James T Thorson, James N Ianelli, Elise A Larsen, Leslie Ries, Mark D Scheuerell, Cody Szuwalski, and Elise F Zipkin. Joint dynamic species distribution models: a tool for community ordination and spatio-temporal monitoring. *Global Ecology and Biogeography*, 2016.

[149] Grigorios Tsoumakas and Ioannis Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *European Conference on Machine Learning*, pages 406–417. Springer, 2007.

[150] Yuta Tsuboi, Hisashi Kashima, Shohei Hido, Steffen Bickel, and Masashi Sugiyama. Direct density ratio estimation for large-scale covariate shift adaptation. *Journal of Information Processing*, 17:138–155, 2009.

[151] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 4, 2017.

[152] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2285–2294, 2016.

[153] Lequn Wang, Qiantong Xu, Christopher De Sa, and Thorsten Joachims. Cost-sensitive learning via deep policy erm. 2018.

[154] Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1658–1665, 2019.

[155] Bryan Wilder, Eric Ewing, Bistra Dilkina, and Milind Tambe. End to end learning and optimization on graphs. *Advances in Neural Information Processing Systems*, 32:4672–4683, 2019.

[156] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Broeck. A

semantic loss function for deep learning with symbolic knowledge. In *International conference on machine learning*, pages 5502–5511. PMLR, 2018.

[157] Xin-Shun Xu, Yuan Jiang, Liang Peng, Xiangyang Xue, and Zhi-Hua Zhou. Ensemble approach based on conditional random field for multi-label image and video annotation. In *Proceedings of the 19th ACM International Conference on Multimedia*, 2011.

[158] Yexiang Xue, Ian Davies, Daniel Fink, Christopher Wood, and Carla P Gomes. Avicaching: A two stage game for bias reduction in citizen science. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 776–785. International Foundation for Autonomous Agents and Multiagent Systems, 2016.

[159] Hongxia Yang, Yada Zhu, and Jingrui He. Local algorithm for user action prediction towards display ads. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 2091–2099, New York, NY, USA, 2017. ACM.

[160] Takayuki Yato and Takahiro Seta. Complexity and completeness of finding another solution and its application to puzzles. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 86(5):1052–1060, 2003.

[161] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International conference on machine learning*, pages 5708–5717. PMLR, 2018.

[162] Gary Young, Emiliano A Valdez, and Robert Kohn. Multivariate probit

models for conditional claim-types. *Insurance: Mathematics and Economics*, 44(2):214–228, 2009.

[163] Kai Yu, Shipeng Yu, and Volker Tresp. Multi-label informed latent semantic indexing. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2005.

[164] Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, page 114. ACM, 2004.

[165] Ning Zhang, Junchi Yan, and Yuchen Zhou. Weakly supervised audio source separation via spectrum energy preserved wasserstein learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4574–4580. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[166] Yin Zhang and Zhi-Hua Zhou. Multilabel dimensionality reduction via dependence maximization. *ACM Transactions on Knowledge Discovery from Data*, 2010.