

**Intelligence, physics and information – the tradeoff
between accuracy and simplicity in machine learning**

by

Tailin Wu

Bachelor of Science in Physics, Peking University 2012

Submitted to the Department of Physics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Physics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2020

© Massachusetts Institute of Technology 2020. All rights reserved.

Author
Department of Physics
November 27, 2019

Certified by
Isaac L. Chuang
Professor of Physics
Thesis Supervisor

Certified by
Max Tegmark
Professor of Physics
Thesis Supervisor

Accepted by
Nergis Mavalvala
Associate Department Head of Physics

Intelligence, physics and information – the tradeoff between accuracy and simplicity in machine learning

by

Tailin Wu

Submitted to the Department of Physics
on November 27, 2019, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Physics

Abstract

How can we make machines more intelligent, so that they can make sense of the world, and become better at learning? To approach this goal, I believe that viewing intelligence in terms of many integral aspects, and also in terms of a universal two-term tradeoff between task performance and complexity, provides two feasible perspectives, and physics and information should play central underlying roles. In this thesis, I address several key questions in some aspects of intelligence, and study the phase transitions in the two-term tradeoff, using strategies and tools from physics and information.

Firstly, how can we make the learning models more flexible and efficient, so that agents can learn quickly with fewer examples? Inspired by how physicists model the world, we approach this question by introducing a paradigm and an Artificial Intelligence Physicist (AI Physicist) agent for simultaneously learning many small specialized models (theories) and the domain they are accurate, which can then be simplified, unified and stored, facilitating few-shot learning in a continual way. We also introduce a Meta-Learning Autoencoder architecture, which utilizes learning good task representations to facilitate few-shot learning.

Secondly, for representation learning, when can we learn a good representation, and how does learning depend on the structure of the dataset? We approach this question by studying phase transitions in the two-term tradeoff: the hyperparameter¹ setting where key quantities, e.g. prediction accuracy change in a discontinuous way. We introduce a technique for predicting when the second-order phase transitions will occur, and in the information bottleneck objective, we show that the formulas we derive reveal deep connections between the data, the model, the learned representation, and the loss landscape of the objective function. For example, each phase transition corresponds to learning a new component of nonlinear maximum correlation between the input and the target, and in classification, they correspond to the learning of new classes.

¹The parameter whose value is set before the learning begins, e.g. relative strength between task performance and complexity in the learning objective.

Thirdly, how can agents discover causality from observations? We address part of this question by introducing an algorithm that combines prediction and minimizing information from the input, for exploratory causal discovery from observational time series.

Last but not least, how can we make classifiers more robust to noise? In the presence of label noise, we introduce Rank Pruning, a robust and general algorithm for classification with noisy labels, prove its consistency, and improve state-of-the-art of learning with noisy labels.

I believe that building on the work of my thesis we will be one step closer to enable more intelligent machines that can make sense of the world.

Thesis Supervisor: Isaac L. Chuang
Title: Professor of Physics

Thesis Supervisor: Max Tegmark
Title: Professor of Physics

Acknowledgments

Firstly, I would like to express my sincere gratitude to Professor Isaac Chuang, my thesis advisor. He encourages me to be a leader in the direction I work on, and his strict criteria help me form my own high standards to strive to realize my full potential. His keen insights in various aspects of physics and computer science have provided valuable guidance in my research. Moreover, he has provided valuable support and guidance in several key moments in my PhD career. Specifically, I would like to thank him for his support when I transformed my research direction from quantum computing experiment to machine learning in my mid-PhD; his introduction of me to a collaboration with Curtis Northcutt which helps me jump start my experience in machine learning; his introduction of me to a collaboration with Ian Fischer from Google, without which I would not have discovered the exciting intersection area between representation learning and phase transitions; and his insightful advice on graduating with momentum and job search. His support and guidance prove pivotal in hindsight, which I am deeply grateful.

Secondly, I would like to express my sincere thanks to Professor Max Tegmark, my thesis co-advisor. I'm grateful for having been working with Max, which greatly expanded my perspective and horizon in AI. As a physicist, Max's strive for simplicity and pursuit of "intelligible intelligence" resonate with me, and has inspired my initial idea of AI Physicist and discovery of the learnability phase transition in the Information Bottleneck. Max has sharp intuitions, and his first-principles thinking stands out, which has influenced me to also think from first principles. I enjoy our close collaboration, as well as discussing universe, intelligence, life.

Thirdly, I would like to thank Ian Fischer. We begin collaborating since the end of 2018, and Ian was also my host for my internship at Google in the summer of 2019. Ian introduced me to the powerful perspective of viewing many machine learning problems in terms of information, which constitutes a major perspective that underlies my thesis. He has guided me with his rich experience in the Information Bottleneck and representation learning, and I also learn a great deal from his scalable numerical experimental skills. I enjoyed our extensive discussions everyday in the January of 2019, when we analyzed the Information

Bottleneck together. I enjoyed a lot our close collaboration during my internship at Google, where we pushed the frontiers on various aspects of the Information Bottleneck.

I would like to thank my other thesis committee members: Professor Marin Soljačić and Professor Riccardo Comin, who have provided valuable guidance and feedbacks in my thesis preparation and advice on career. I would also like to thank my academic advisor Professor Wolfgang Ketterle, who has provided insightful advice throughout my PhD.

I would like to thank my other collaborators: Curtis Northcutt, who I had a great time working with, with our complementary skills we really form a good team; Thomas Breuel and Jan Kautzin, who guided me during my internship at NVIDIA; Michael Skuhersky, with whom I am having fun with the *C.elegans*.

I would also like to thank many other people who I have discussed research with: Professor Josh Tenenbaum, Alex Alemi, Kevin Murphy, who have provided valuable feedbacks on the research projects.

Last but not least, I would like to thank my parents, who have given me the chance to experience this wonderful Universe, and have given me unconditional support all the time. I love you!

Contents

1	Introduction	29
1.1	What is intelligence?	29
1.2	First perspective: aspects of intelligence	34
1.3	Second perspective: universal two-term trade-off	37
1.4	Roadmap	39
1.4.1	Scope and structure of the thesis	39
1.4.2	My contributions	42
2	AI Physicist for few-shot, lifelong learning of physics	45
2.1	Introduction	46
2.1.1	Motivation	46
2.1.2	Goals & relation to prior work	47
2.2	Methods	49
2.2.1	Definition of Theory	49
2.2.2	Divide-and-Conquer	50
2.2.3	AI Physicist Architecture Overview	50
2.2.4	Occam's Razor	53
2.2.5	Unification	55

2.2.6	Lifelong Learning	57
2.3	Results of Numerical Experiments	58
2.3.1	Physics Environments	58
2.3.2	Numerical Results	58
2.4	Conclusions	63
2.4.1	Key findings	64
2.4.2	What has been learned?	64
2.4.3	Outlook	66
3	Learnability phase transition: onset of learning	67
3.1	Introduction	68
3.2	Related Work	70
3.3	IB-Learnability	72
3.4	Sufficient conditions for IB-Learnability	74
3.5	Discussion	78
3.5.1	The conspicuous subset determines β_0	78
3.5.2	Multiple phase transitions.	78
3.5.3	Similarity to information measures.	79
3.5.4	Estimating model capacity.	79
3.5.5	Learnability and the Information Plane.	80
3.5.6	IB-Learnability, hypercontractivity, and maximum correlation.	81
3.6	Estimating the IB-Learnability Condition	81
3.6.1	Estimation Algorithm	81
3.6.2	Special Cases for Estimating β_0	82
3.7	Experiments	84

3.7.1	Synthetic Dataset Experiments	84
3.7.2	MNIST Experiments	88
3.7.3	MNIST Experiments using Equation 3.2	89
3.7.4	CIFAR10 Forgetting Experiments	90
3.8	Conclusion	90
4	Intermediate phase transitions	95
4.1	Introduction	96
4.2	Related Work	98
4.3	Formula for IB phase transitions	99
4.3.1	Definitions	99
4.3.2	Condition for IB phase transitions	101
4.4	Understanding the formula for IB phase transitions	103
4.4.1	Jensen’s Inequality	103
4.4.2	Representational Maximum Correlation	104
4.5	Algorithm for phase transitions discovery in classification	107
4.6	Empirical study	108
4.6.1	Categorical dataset	108
4.6.2	MNIST dataset	108
4.6.3	CIFAR10 dataset	110
4.7	Conclusion	110
5	Pareto-optimal data compression for binary classification tasks	113
5.1	Introduction	114
5.1.1	Objectives & relation to prior work	115

5.2	Method	119
5.2.1	Lossless distillation for classification tasks	120
5.2.2	Pareto-optimal compression for binary classification tasks	122
5.2.3	Mapping the frontier	129
5.3	Results	130
5.3.1	Analytic warmup example	130
5.3.2	The Pareto frontier	131
5.3.3	MNIST, Fashion-MNIST and CIFAR-10	134
5.3.4	Interpretation of our results	137
5.3.5	Real-world issues	140
5.3.6	Performance compared with Blahut-Arimoto method	142
5.4	Conclusions	143
5.4.1	Relation to Information Bottleneck	143
5.4.2	Relation to phase transitions in DIB learning	144
5.4.3	Outlook	145
6	Learning causal relations from observations	149
6.1	Introduction and Related Work	149
6.2	Method	152
6.2.1	Problem setup	152
6.2.2	Our method	153
6.3	Experiments	157
6.3.1	Synthetic experiment with log-normal causal strengths	157
6.3.2	Experiments with video games	159
6.3.3	Experiment with heart-rate vs. breath-rate and rat brain EEG datasets	161
6.4	Discussion and conclusion	162

7 Meta-learning autoencoders for few-shot prediction	163
7.1 Introduction	164
7.2 Methods	166
7.2.1 Meta-learning problem setup	166
7.2.2 Meta-learning autoencoder architecture	166
7.2.3 MeLA's meta-training and evaluation	168
7.2.4 Influence identification	170
7.2.5 Interactive learning	171
7.3 Related work	172
7.4 Experiments	173
7.4.1 Simple regression problem	174
7.4.2 Ball bouncing with state representation	175
7.4.3 Video prediction	176
7.5 Conclusions	177
8 Rank Pruning for robust learning with noisy labels	179
8.1 Introduction	180
8.1.1 Related Work	181
8.1.2 Contributions	184
8.2 Framing the $\tilde{P}\tilde{N}$ Learning Problem	185
8.3 Rank Pruning	187
8.3.1 Deriving Noise Rate Estimators $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$	187
8.3.2 Noise Estimation: Unassuming Case	189
8.3.3 The Rank Pruning Algorithm	191
8.3.4 Rank Pruning: A simple summary	192

8.3.5	Expected Risk Evaluation	193
8.4	Experimental Results	195
8.4.1	Synthetic Dataset	196
8.4.2	MNIST and CIFAR Datasets	197
8.5	Discussion	199
9	Conclusion and Prospects	203
9.1	Conclusions	203
9.2	Prospects	204
A	Appendix	207
A.1	Appendix for Chapter 2	208
A.1.1	AI Physicist Algorithm	208
A.1.2	The Differentiable Divide-and-Conquer (DDAC) Algorithm	208
A.1.3	Occam's Razor with MDL Algorithm	210
A.1.4	Unification Algorithm	213
A.1.5	Adding and Proposing Theories	214
A.1.6	Time complexity	215
A.1.7	Proof of Theorem 1 and Corollary	216
A.1.8	Eliminating Transition Domains	219
A.1.9	Numerical Experiment Details	221
A.2	Appendix for Chapter 3	233
A.2.1	Preliminaries: first-order and second-order variations	233
A.2.2	Proof of Lemma 1.1	234
A.2.3	Proof of Theorem 2	234

A.2.4	Proof of Theorem 3	234
A.2.5	First- and second-order variations of $IB_\beta[p(z x)]$	235
A.2.6	Proof of Lemma 2.1	240
A.2.7	Proof of Theorem 4	240
A.2.8	What IB first learns at its onset of learning	245
A.2.9	Proof of Theorem 5	248
A.2.10	Proof of Corollary 5.1 and Corollary 5.2	250
A.2.11	β_0 hypercontractivity coefficient, contraction coefficient, $\beta_0[h(x)]$ and maximum correlation	251
A.2.12	Experiment Details	254
A.3	Appendix for Chapter 4	257
A.3.1	Calculus of variations at any order of $IB_\beta[p(z x)]$	257
A.3.2	Proof of Lemma 5.1	260
A.3.3	$G_\Theta[p_\theta(z x)]$ for parameterized distribution $p_\theta(z x)$	263
A.3.4	Proof of Theorem 6	267
A.3.5	Invariance of $\mathcal{G}[r(z x); p(z x)]$ to addition of a global representation	268
A.3.6	Proof of Theorem 7	269
A.3.7	Subset separation at phase transitions	275
A.3.8	MNIST Experiment Details	276
A.3.9	CIFAR10 Experiment Details	276
A.4	Appendix for Chapter 5	279
A.4.1	Binning can be practically lossless	279
A.4.2	More varying conditional probability boosts mutual information . .	282
A.5	Appendix for Chapter 6	284

A.5.1	Hyperparameter λ selection	284
A.5.2	Proof and analysis of the Minimum Predictive Information regularized risk	284
A.5.3	Upper bound for the mutual information-regularized risk	297
A.5.4	Implementation details for the methods	299
A.5.5	Implementation details for synthetic experiments	301
A.5.6	AUC-ROC table for synthetic experiment	302
A.5.7	Additional experiment: testing with model capacity variations	302
A.5.8	Details for the video game dataset	303
A.5.9	Implementation details for experiment with heart-rate vs. breath-rate	304
A.5.10	Additional experiment: rat EEG dataset	304
A.6	Appendix for Chapter 7	309
A.7	Appendix for Chapter 8	311
A.7.1	Proofs	311
A.7.2	Additional Figures	324
A.7.3	Additional Tables	324
A.7.4	Additional Related Work	326

List of Figures

2-5 In this mystery, a charged double pendulum moves through two different electric fields \mathbb{E}_1 and \mathbb{E}_2 , with a domain boundary corresponding to $\cos \theta_1 + \cos \theta_2 = 1.05$ (the black curve above left, where the lower charge crosses the \mathbb{E} -field boundary). The color of each dot represents the domain into which it is classified by a Newborn agent, and its area represents the description length of the error with which its position is predicted, for a precision floor $\epsilon \approx 0.006$. In this world, the Newborn agent has a domain prediction accuracy of 96.5%.	63
3-1 Accuracy for binary classification of MNIST digits 0 and 1 with 20% label noise and varying β . No learning happens for models trained at $\beta < 3.25$.	70
3-2 The Pareto frontier of the information plane, $I(X; Z)$ vs $I(Y; Z)$, for the binary classification of MNIST digits 0 and 1 with 20% label noise described in Sec. 3.1 and Fig. 3-1. For this problem, learning happens for models trained at $\beta > 3.25$. $H(Y) = 1$ bit since only two of ten digits are used, and $I(Y; Z) \leq I(X; Y) \approx 0.5$ bits $< H(Y)$ because of the 20% label noise. The true frontier is differentiable; the figure shows a variational approximation that places an upper bound on both informations, horizontally offset to pass through the origin.	80
3-3 Predicted vs. experimentally identified β_0 , for mixture of Gaussians with varying class-conditional noise rates.	85
3-4 $I(Y; Z)$ vs. β , for mixture of Gaussian datasets with different distances between the two mixture components. The vertical lines are $\beta_{0,\text{predicted}}$ computed by the R.H.S. of Eq. (3.8). As Eq. (3.8) does not make predictions w.r.t. class overlap, the vertical lines are always just above $\beta_{0,\text{predicted}} = 1$. However, as expected, decreasing the distance between the classes in X space also increases the true β_0 .	87

3-5	$I(Y; Z)$ vs. β for the MNIST binary classification with different hidden units per layer n and noise rates ρ : (upper left) $\rho = 0.02$, (upper right) $\rho = 0.1$, (lower left) $\rho = 0.2$, (lower right) $\rho = 0.3$. The vertical lines are β_0 estimated by different methods. $n = 128$ has insufficient capacity for the problem, so its observed learnability onset is pushed higher, similar to the class overlap case.	88
3-6	Histograms of the full MNIST training and validation sets according to $h(X)$. Note that both are bimodal, and the histograms are indistinguishable. In both cases, $h(x)$ has learned to separate most of the ones into the smaller mode, but difficult ones are in the wide valley between the two modes. See Figure 3-8 for all of the training images to the left of the red threshold line, as well as the first few images to the right of the threshold.	89
3-7	Plot of $I(Y; Z)$ vs β for CIFAR10 training set with 20% label noise. Each blue cross corresponds to a fully-converged model starting with independent initialization. The vertical black line corresponds to the predicted $\beta_0 = 1.0483$ using Alg. 1. The empirical $\beta_0 = 1.048$	90
3-8	The first 5776 MNIST training set digits when sorted by $h(x)$. The digits highlighted in red are above the threshold drawn in Figure 3-6.	91
4-1	CIFAR10 plots (a) showing the information plane, as well as β vs (b) $I(X; Z)$ and $I(Y; Z)$, and (c) accuracy, all on the training set with 20% label noise. The arrows point to empirically-observed phase transitions. The vertical lines correspond to phase transitions found with Alg. 2.	97
4-2	(a) $I(Y; Z^*)$ vs. β for a categorical dataset with $ X = Y = Z = 3$, where Z^* is given by $p_\beta^*(z x)$, and the vertical lines are the experimentally discovered phase transition points β_0^c and β_1^c . (b) $G[p_\beta^*(z x)]$ vs. β for the same dataset, and the path for Alg. 2, with β_0^c and β_1^c in (a) also plotted. The dataset is given in Fig. A-3.	109

5-2 Sample data from Section 5.3. Images from MNIST (top), Fashion-MNIST (middle) and CIFAR-10 are mapped into integers (group labels) $Z = f(X)$ retaining maximum mutual information with the class variable Y (ones/sevens, shirts/pullovers and cats/dogs, respectively) for 3, 5 and 5 groups, respectively. These mappings f correspond to Pareto frontier “corners”.	123
5-3 Essentially all information about Y is retained if W is binned into sufficiently narrow bins. Sorting the bins (left) to make the conditional probability monotonically increasing (right) changes neither this information nor the entropy.	124
5-4 The reason that the Pareto frontier can never be reached using non-contiguous bins is that a swapping parts of them against parts of an intermediate bin can increase $I(Z, X)$ while keeping $H(Z)$ constant. In this example, the binning function g assigns two separate W -intervals (top panel) to the same bin (bin 2) as seen in the bottom panel. The shaded rectangles have widths P_i , heights p_i and areas $P_{i1} = P_i p_1$. In the upper panel, the conditional probabilities p_i are monotonically increasing because they are averages of the monotonically increasing curve $p_1(w)$.	127
5-5 Contour plot of the function $W(x_1, x_2)$ computed both exactly using equation 5.27 (solid curves) and approximately using a neural network (dashed curves).	133
5-6 Cumulative distributions $F_i(w) \equiv P(W < w Y=i)$ are shown for the analytic (blue/dark grey), Fashion-MNIST (red/grey) and CIFAR-10 (orange/light grey) examples. Solid curves show the observed cumulative histograms of W from the neural network, and dashed curves show the fits defined by equation (5.31) and Table 5.2.	136

5-7 The solid curves show the actual conditional probability $P(Y=1 W)$ for CIFAR-10 (where the labels Y=1 and 2 correspond to “cat” and “dog”) and MNIST with 20% label noise (where the labels Y=1 and 2 correspond to “1” and “7”), respectively. The color-matched dashed curves show the conditional probabilities predicted by the neural network; the reason that they are not diagonal lines $P(Y=1 W) = W$ is that W has been reparametrized to have a uniform distribution. If the neural network classifiers were optimal, then solid and dashed curves would coincide.	136
5-8 The Pareto frontier for compressed versions $Z = g(X)$ of our four datasets X , showing the maximum attainable class information $I(Z, Y)$ for a given entropy $H(Z)$. The “corners” (dots) correspond to the maximum $I(Z, Y)$ attainable when binning the likelihood W into a given number of bins (2, 3, ..., 8 from right to left). The horizontal dotted lines show the maximum available information $I(X, Y)$ for each case, reflecting that there is simply less to learn in some examples than in others.	137
5-9 The Pareto frontier our analytic example is computed exactly with our method (solid curve) and approximately with the Blahut-Arimoto method (dots).	143
6-1 (a) Predictive strength W_{ji} inferred by our method in Section 6.3.2. The (j, i) element denotes the inferred causal strength from j to i . (e) True underlying causal relations are marked dark, with light color marking competing causal relations that are indistinguishable from data. Other subfigures are: directional strength inferred by (b) mutual information (c) transfer entropy (d) linear Granger (f) kernel Granger (g) elastic net (h) causal influence. . .	160

6-2 (a) Predictive strength W_{ji} inferred by our method with the heart-rate vs. breath-rate dataset, averaged over 50 initializations of f_θ . The shaded areas are the 95% confidence interval. (b) Upper: the filtered causality index vs. varying width of Gaussian kernel σ [MPS08b]; lower: transfer entropy vs. r , the length scale [Sch00]; (c) The causality index for breath→heart (lower) and heart→breath (upper) in [AMS04], where m is the maximum time lag (equivalent to our K).	161
7-1 Architecture of our Meta-Learning Autoencoder (MeLA). MeLA augments a pre-existing neural network architecture f_θ (right) with a meta-recognition model (left) that generates the model code \mathbf{z} based on a few examples \mathbf{X}' , \mathbf{Y}' , and a meta-generative model (middle) that generates the parameters θ of model f_θ based on the model code. f_θ , g_γ and m_μ are implemented as multilayer perceptron (MLP).	167
7-2 (a) MSE vs. number of gradient steps (with learning rate = 0.001, Adam optimizer) on 20,000 randomly sampled testing datasets, for MeLA, MAML, baseline (pretrained) and oracle. MeLA starts at MSE of 0.208 and gets down to 0.129 after 10 steps, while MAML starts at 3.05 and gets down to 0.208 after 5 steps. (b) Predictions after 0 gradient steps for an example test dataset (MAML is after 1 gradient step). The markers' size is proportional to the influence identified by MeLA. Also plotted is MeLA's prediction given only the top 3 influential examples. (c) To get a better prediction at $x^* = -4$ using only two examples at hand, MeLA requests the example at $x = -0.593$ from 8 candidate positions (vertical lines). (d) Improved estimate at $x^* = -4$ after obtaining the requested example.	174
7-3 (a) Examples of the polygon "bouncy-house" environments. (b) Mean Euclidean distance between target and prediction vs. rollout distance traveled on 1000 randomly generated testing environments.	176

7-4 Ball bouncing prediction by MeLA for an example testing dataset. Also plotted are the top 10 most influential training trajectories identified by MeLA, which are all near the vertices.	176
7-5 (a) Example MeLA prediction vs. true trajectory for 5 rollout steps, with an unseen testing environment without gradient steps. (b) Mean Euclidean distance between the center of mass (COM) of true trajectory and prediction vs. rollout distance traveled for MeLA, pretrained and oracle on 100 randomly generated testing environments.	177
8-1 Illustration of Rank Pruning with a logistic regression classifier (\mathcal{LR}_θ). (a) : The corrupted training set D_ρ with noise rates $\rho_1 = 0.4$ and $\rho_0 = 0.1$. Corrupted colored labels ($s = 1, s = 0$) are observed. $y (+, -)$ is hidden. (b) : The marginal distribution of D_ρ projected onto the x_p axis (indicated in (a)), and the \mathcal{LR}_θ 's estimated $g(x)$, from which $\hat{\rho}_1^{conf} = 0.4237$, $\hat{\rho}_0^{conf} = 0.1144$ are estimated. (c) : The pruned X_{conf}, s_{conf} . (d) : The classification result by Rank Pruning ($\hat{f} = \mathcal{LR}_\theta.\text{fit}(X_{conf}, s_{conf})$), ground truth classifier ($f = \mathcal{LR}_\theta.\text{fit}(X, y)$), and baseline classifier ($g = \mathcal{LR}_\theta.\text{fit}(X, s)$), with an accuracy of 94.16%, 94.16% and 78.83%, respectively.	194
8-2 Comparison of Rank Pruning with different noise ratios (π_1, ρ_1) on a synthetic dataset for varying separability d , dimension, added random noise and number of training examples. Default settings for Fig. 8-2, 8-3 and 8-4: $d = 4$, 2-dimension, 0% random noise, and 5000 training examples with $p_{y1} = 0.2$. The lines are an average of 200 trials.	195
8-3 Sum of absolute difference between theoretically estimated $\hat{\rho}_i^{thry}$ and empirical $\hat{\rho}_i$, $i = 0, 1$, with five different (π_1, ρ_1) , for varying separability d , dimension, and number of training examples. Note that no figure exists for percent random noise because the theoretical estimates in Eq. (8.8) do not address added noise examples.	196

8-4 Comparison of $\tilde{P}\tilde{N}$ methods for varying separability d , dimension, added random noise, and number of training examples for $\pi_1 = 0.5$, $\rho_1 = 0.5$ (given to all methods).	197
8-5 Rank Pruning $\hat{\rho}_1$ and $\hat{\pi}_1$ estimation consistency, averaged over all digits in MNIST. (a) Color depicts $\hat{\rho}_1 - \rho_1$ with $\hat{\rho}_1$ (upper) and theoretical $\hat{\rho}_1^{thry}$ (lower) in each block. (b) Color depicts $\hat{\pi}_1 - \pi_1$ with $\hat{\pi}_1$ (upper) and $\hat{\pi}_1^{thry}$ (lower) in each block.	198
9-1 Prospect of different aspects combined integrally together.	205
A-1 Points where forward and backward extrapolations agree (large black dots) are boundary points. The tangent vectors agree for region boundaries (upper example), but not for bounce boundaries (lower example).	220
A-2 Example of automatically determined boundary points, for region boundary points (green), bounce boundary points (black) and failed cases (red).	222
A-3 $p(y x)$ for the categorical dataset in Fig. 4-2 and Fig. A-4. The value in i^{th} row and j^{th} column denotes $p(y = j x = i)$. $p(x)$ is uniform.	275
A-4 (a) $I(Y; Z)$ vs. β for the dataset given in Fig. A-3. The phase transitions are marked with vertical dashed line, with $\beta_0^c = 2.065571$ and $\beta_1^c = 5.623333$. (b)-(e) Optimal $p_{\beta}^*(z x)$ for four values of β , i.e. (b) $\beta = 2.060$, (c) $\beta = 2.070$, (d) $\beta = 5.620$ (e) $\beta = 5.625$ (their β values are also marked in (a)), where each marker denotes $p(z x = i)$ for a given $i \in \{0, 1, 2\}$	277
A-5 Confusion matrix for MNIST experiment. The value in i^{th} row and j^{th} column denotes $p(\tilde{y} = j y = i)$ for the label noise.	278

List of Tables

2.1	AI Physicist strategies tested.	47
2.2	Summary of numerical results, taking the median over 40 mystery environments from Table A.1 (top part) and on 40 novel environments with varying fraction of random examples (bottom parts), where each world is run with 10 random initializations and taking the best performance. Accuracies refer to big regions only.	62
3.1	Full table of values used to generate Fig. 3-3.	86
5.1	Data distillation: the relationship between Principal Component Analysis (PCA), Canonical Correlation Analysis (CCA), nonlinear autoencoders and nonlinear latent representations.	116
5.2	Fits to the conditional probability distributions $P(W Y)$ for our experiments, in terms of the parameters a_i defined by equation (5.31).	134
6.1	Mean and standard deviation of AUC-PR (%) vs. N , over 10 random sampling of datasets. Bold font marks the top method for each N	159
8.1	Variable definitions and descriptions for $\tilde{P}\tilde{N}$ learning and PU learning. Related work contains a prominent author using each variable. ρ_1 is also referred to as <i>contamination</i> in PU learning literature.	182
8.2	Summary of state-of-the-art and selected general solutions to $\tilde{P}\tilde{N}$ and PU learning.	182

8.3 Comparison of F1 score for one-vs-rest MNIST and CIFAR-10 (averaged over all digits/images) using logistic regression. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if greater than non-RP models. Due to sensitivity to imperfect $g(x)$, <i>Liu16</i> often predicts the same label for all examples.	199
8.4 F1 score comparison on MNIST and CIFAR-10 using a CNN. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods.	200
A.1 Results for each of our first 40 mystery world benchmarks, as described in the section A.1.9. Each number is the best out of ten trials with random initializations (using seeds 0, 30, 60, 90, 120, 150, 180, 210, 240, 270), and refers to big domains only. Based on the “Unsolved domain” column, we count out of 40 worlds what’s the percentage Baseline, Newborn and AI Physicist completely solve (has unsolved domain of 0), which goes to the “Fraction of worlds solved” row in Table 2.2.	225
A.2 Same as previous table, but showing number of training epochs required to reach various MSE prediction accuracies. We record the metrics every 5 epochs, so all the epochs are multiples of 5. Note that the AI Physicist has superseded 10^{-2} MSE already by 0 epochs for some environments, showing that thanks to the lifelong learning strategy which proposes previously learned theories in novel environments, reasonably good predictions can sometimes be achieved even without gradient descent training.	226
A.3 Hyperparameter settings in the numerical experiments. For a fair comparison between Baseline and the other agents that can have up to 4 theories, the number of neurons in each layer of Baseline is larger so that the total number of parameters is roughly the same for all agents. The Baseline agent in Mystery worlds has leakyReLU activation to be able to account for different domains.	228

A.4	Class confusion matrix used in CIFAR10 experiments. The value in row i , column j means for class i , the probability of labeling it as class j . The mean confusion across the classes is 20%.	256
A.5	Class confusion matrix used in CIFAR10 experiments, reproduced from [WFCT19a]. The value in row i , column j means for class i , the probability of labeling it as class j . The mean confusion across the classes is 20%.	278

Chapter 1

Introduction

1.1 What is intelligence?

What is intelligence? This is a question that has been intriguing generations of scientists and artificial intelligence (AI) researchers. The understanding and building of intelligence not only is in itself an important question, but also has profound influence on the society, with the potential to help solve important problems. For example, it has helped scientists predict protein structure from genomic data with unprecedented precision [RJJ⁺18], simulate light scattering by multilayer nanoparticles and facilitate nanophotonic inverse design [PSJ⁺18], predict molecule properties in quantum chemistry [GSR⁺17b], identify central nervous system tumours [CJS⁺18], reconstruct neural circuit map [JKL⁺17], etc. But I believe we have just seen a tip of the iceberg of what it may achieve. Future intelligent machines may help solve important problems that benefit humanity as a whole, for example, design more efficient ways for space exploration, automatically discover new physics and new mathematics, identify mechanisms in biological systems and propose cure for disease, to name just a few.

Physics has been extending its scope of study from space and time to matter and energy in all its forms, from the subatomic to the cosmological, from the elementary to the complex, and from the inanimate to living organisms. The study of intelligence is another frontier physics should and can illuminate. The wisdom and strategies developed by generations of physicists

may help better understand and build intelligence, just as many of the techniques and perspectives in AI domain are inspired by physics, e.g. energy models [LCH⁺06], simulated annealing [KGV83, Čer85], Hamiltonian Monte Carlo (HMC) [DKPR87], critical behaviors in random Boolean expressions [KS94] and data representations [CJM⁺19], neural ordinary differential equations [CRBD18], fluctuation-dissipation relations for stochastic gradient descent [Yai19], to name just a few. In turn, the better intelligent algorithms can help solve important physics problems, e.g. quantum state reconstruction [CTMA19], phase transitions [CM17, Wan16, VNLH17], planetary dynamics [LK18] and particle physics [BSW14].

Before diving into studying it, it is important that we have a notion of what “intelligence” is. As R. J. Sternberg has put it [GZ87], “Viewed narrowly, there seem to be almost as many definitions of intelligence as there were experts asked to define it”. Still, there exist similarities among many of the definitions. Below I quote some of the prominent definitions:

- (1) “Intelligence measures an agent’s ability to achieve goals in a wide range of environments” [LH⁺07b].
- (2) “The intelligence of a system is a measure of its skill-acquisition efficiency over a scope of tasks, with respect to priors, experience, and generalization difficulty” [Cho19]. In this paper, the author argues that measuring skill by testing on the same kind of training task (“local generalization”) does not measure intelligence, since the skills can be bought by training with arbitrary number of examples or injecting prior knowledge by the developer, which should be on the orthogonal axis of intelligence. He then proposes a new measure of intelligence, which roughly translates to

$$\text{intelligence} := \mathbb{E} \left[\frac{\text{skill} \times (\text{generalization difficulty})}{\text{prior} + \text{experience}} \right]$$

where the generalization difficulty is defined as the ratio of the length of shortest program that solves the testing task given the shortest program that achieves optimal training-time performance over the situations in the curriculum, over the length of the shortest program that solves the testing task.

- (3) "... the ability of a system to act appropriately in an uncertain environment, where appropriate action is that which increases the probability of success, and success is the achievement of behavioral subgoals that support the system's ultimate goal" [Alb91].
- (4) "Intelligence is the ability to use optimally limited resources - including time - to achieve goals" [Kur00].
- (5) "Humans (machines) are intelligent to the extent that our (their) actions can be expected to achieve our (their) objectives." [Rus19].
- (6) "Intelligence is the power to rapidly find an adequate solution in what appears *a priori* (to observers) to be an immense search space" [LF92].
- (7) "Intelligence is the ability to process information properly in a complex environment. The criteria of properness are not predefined and hence not available beforehand. They are acquired as a result of the information processing." [Nak99].
- (8) "Intelligence means getting better over time" [Sch91].
- (9) "Intelligence = Ability to accomplish complex goals" [Teg17].
- (10) "Intelligence is the ability for an information processing system to adapt to its environment with insufficient knowledge and resources" [Wan95].
- (11) "Intelligence is not a single, unitary ability, but rather a composite of several functions. The term denotes that combination of abilities required for survival and advancement within a particular culture" [Ana92].
- (12) "...in its lowest terms intelligence is present where the individual animal, or human being, is aware, however dimly, of the relevance of his behaviour to an objective. Many definitions of what is indefinable have been attempted by psychologists, of which the least unsatisfactory are 1. the capacity to meet novel situations, or to learn to do so, by new adaptive responses and 2. the ability to perform tests or tasks, involving the grasping of relationships, the degree of intelligence being proportional to the complexity, or the abstractness, or both, of the relationship" [Col15].

- (13) “Intelligence is assimilation to the extent that it incorporates all the given data of experience within its framework . . . There can be no doubt either, that mental life is also accommodation to the environment. Assimilation can never be pure because by incorporating new elements into its earlier schemata the intelligence constantly modifies the latter in order to adjust them to new elements.” [Pia05].
- (14) “... certain set of cognitive capacities that enable an individual to adapt and thrive in any given environment they find themselves in, and those cognitive capacities include things like memory and retrieval, and problem solving and so forth. There’s a cluster of cognitive abilities that lead to successful adaptation to a wide range of environments” [Sim03].

We see that although the definitions vary, there exist similar aspects of the definitions. For example, (1) intelligence involves an agent’s interaction with the *environment*; (2) it is a property of the agent’s *information processing*; (3) it involves the agents ability to *achieve goals* or *solve tasks* in diverse environments, under environmental constraints (e.g. incomplete information, limited resources of computation, space or time).

To better understand intelligence, I believe the following two perspectives provide feasible routes. For the first perspective, instead of trying to directly understand and build intelligence using a single formula or definition, I believe a better approach is to understand its different aspects. Just as in the understanding of life, although once doubting whether biological mechanisms could ever explain the property of being alive, biologists gradually uncovered aspects of life, for example metabolism, homeostasis and reproduction. Physicist Erwin Schrödinger also introduced the idea of an “aperiodic crystal” containing genetic information [Sch92] that inspired the discovery of DNA. I think intelligence is a *system*. Just as a human is a system, with its body, hands, sensory organs, different parts of the brain (e.g. visual cortex, hippocampus, prefrontal cortex, etc.), intelligence should also have many aspects that work integrally to form intelligent behavior. By identifying and understanding the various aspects of intelligence (which we expand on in section 1.2) with humans as a blueprint, we may have a feasible route to fully understand intelligence.

The second perspective is that, inspired by the above common aspect in definitions of intelligence that agent can solve tasks in diverse environments under environmental constraints, I view intelligence as a result of the ability to solve a universal two-term trade-off in diverse settings, which may be a manifestation and approximation of the AIXI formalism [LH07a] under different scenarios, and resonant with the notion of “intelligence as compression” [LH07a]. In the two-term trade-off, we have one term that measures the agent’s performance on the task, and another term that constrains resources of the agent, usually in the form of limiting the complexity of the learned model or representation. The agent has to find the best solution to a task under such constraints. In machine learning, this usually comes in the form of regularization.

In the above two perspectives, physics and information have the potential to play a central role. Since intelligence involves an agent’s interaction with the environment, in order to achieve a diverse set of goals in the environment, the agent has to first understand the *physics* of the environment, so as to utilize it and direct the environment to its need, just as humans cannot build rockets without understanding Newton’s law of universal gravitation. To build intelligent agents, we can get inspirations from how generations of physicists study and model the different aspects of the world, from space and time to matter and energy, from the subatomic to the cosmological, from the elementary to the complex, and from the inanimate to living organisms. Therefore, the different aspects of intelligence, viewed from the first perspective, should benefit a lot from borrowing strategies and techniques from physics.

The second perspective, viewing intelligence as the ability to solve a universal two-term tradeoff in diverse settings, is also resonant with the central goal of physics. We want to find physics theories that not only can predict parts of Nature precisely, but are also *simple*, where the simplicity may be governed by Occam’s razor principle, or measured by Kolmogorov complexity [Kol63] or description length [Ris83], or their approximations.

Information, on the other hand, should also lie at the core of intelligence. As we have stated before, intelligence can be viewed as a specific form of information processing. Moreover, I believe that this specific form of information processing should be independent of the substrate on which it performs, be it a human brain, a computer, a coordinated group of

people, some emergent plasma behavior on the sun, or even an imaginary scenario of billions of ants doing the exact same information processing as a brain does. It is *structure* of the information processing that matters, not the substrate that performs it.¹ To understand the information processing that gives rise to intelligence, information theory², and the variational techniques developed thereupon, provide an ideal language and technique³ to measure, constrain and optimize the information flow, regardless of the architecture of the model, and the task the agent is solving. Therefore, information theory may provide a valuable tool in developing different aspects of intelligence, and may arise as a natural objective in the universal two-term tradeoff, in either measuring the task performance, or constraining the complexity of the model, as we shall see in the following two sections.

As a sidenote, physics and information are not unrelated. Instead, information plays important roles in many branches of physics. For example, thermodynamics involves the study of *entropy*⁴ and its interplay with other thermodynamic quantities; a generalized version of second law of thermodynamics [BRLW17] involves mutual information; quantum information studies the information processing and channels in quantum scenarios; to name just a few. Moreover, simplicity as measured by Kolmogorov complexity has a close connection with entropy [GV04].

The above two perspectives of intelligence, and the central role physics and information play, lay the foundation of my thesis. In the following two sections I will expand each perspective in more detail, while putting my thesis work into the picture.

1.2 First perspective: aspects of intelligence

Regarding intelligence as a system with many integral aspects working together, just as a computer or a human can be thought of as a system, I believe the following aspects are

¹This understanding is also resonant with the notion that consciousness is independent of the substrate, and only depend of the information processing [Teg17].

²Information theory studies the quantification, storage, and communication of information. It was originally introduced by Claude Shannon [Sha48a] and further developed by many others.

³There may be other techniques that address other aspects of the information processing, besides information theory.

⁴which is also called “self information”.

integral to intelligence:

Basic level:

- (1) **Sensory inputs**: it should be able to receive sensory inputs from the outside world.
- (2) **Intrinsic reward**: it should be able to generate intrinsic reward from interactions with the environment, directing its action.
- (3) **Memory**: it should have some form of memory.
- (4) **Actuators**: it should be able to influence the environment by its actuator.

High level:

- (1) **Working robustly**: it should be able to work robustly in noisy and complex real environments.
- (2) **Working intelligibly⁵**: we should be able to understand its goal and how it makes its decisions.
- (3) **Learning good representations**: it should be able to learn good internal representations for the tasks it is assigned to.
- (4) **Communication**: it should be able to communicate with human or other agents with succinct language.
- (5) **Predicting the future**: it should be able to learn to predict the future from the past, allowing it to understand how the world works, and better plan its action and achieving its goal.
- (6) **Causal inference**: it should be able to think in terms of cause and effect, and infer causal relations from observations or by performing experiments.
- (7) **Planning**: it should be able to plan its action to achieve its goal.
- (8) **Few-shot learning**: it should be able to learn to learn across tasks, allowing it to learn with few examples for novel tasks.

⁵Although human intelligence is often not intelligible, intelligibility is arguably a desirable trait for AI.

- (9) **Lifelong learning:** it should be able to continuously learn new things without forgetting the past learning. This includes lifelong learning of prediction models, and lifelong skill acquisition.
- (10) **Emotional intelligence:** it should have a theory of mind, and be able to perceive other agents' feeling within their frame of reference.
- (11) (Something we don't know yet)

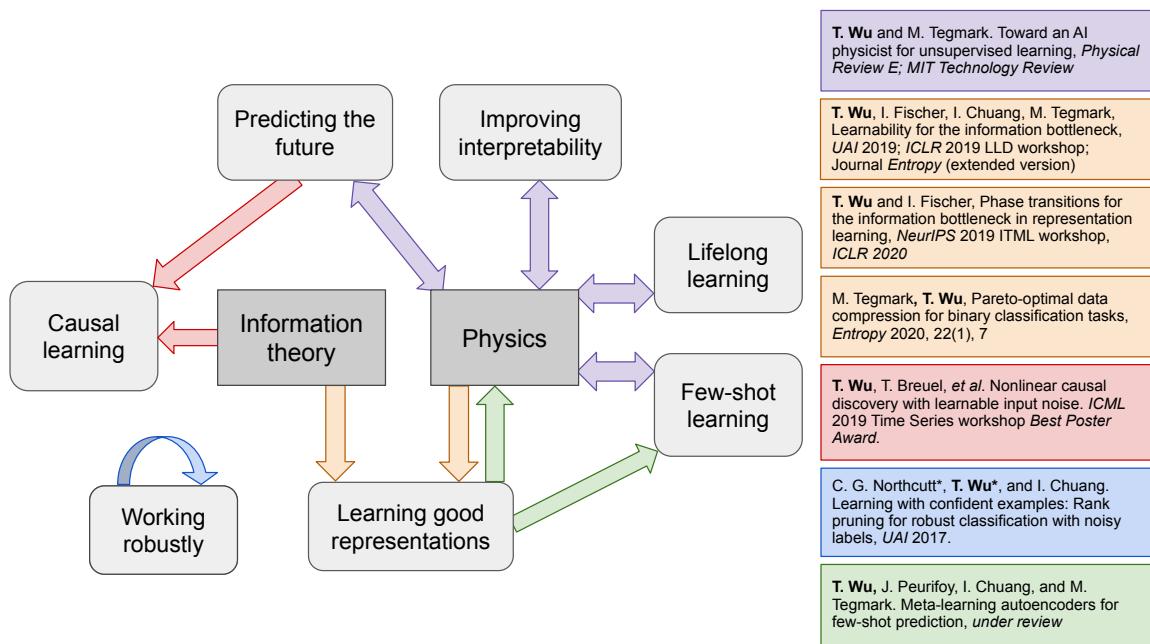


Figure 1-1: How different parts of my thesis address the aspects of intelligence.

I believe a seamless integration of the above aspects in a single architecture will bring us closer to understanding intelligence. Moreover, the above aspects are not independent. Instead, doing well on one aspect will probably help some others.

Based on the above understanding, I try in my thesis to combine aspects together if possible, and use one aspect to help another. My different thesis projects and their addressed aspects are shown in Fig. 1-1. As we see, there are many connections between different aspects, and one aspect can help another. For example, the causal learning project (red) uses predicting the future for causal learning; the meta-learning autoencoder project (green) applies learning good representations to achieve few-shot learning.

Moreover, from Fig. 1-1, we see that physics and information play a central role in connecting these different aspects. For example, the AI Physics project (purple) uses techniques inspired by physics, for predicting the future, improving interpretability, few-shot learning and lifelong learning aspects of intelligence, which in turn help the learning of physics theories. The information bottleneck projects (yellow) applies the analogy of phase transition in physics for understanding the phase transitions in representation learning, and whose objective uses information theory for representation learning.

1.3 Second perspective: universal two-term trade-off

As mentioned in Section 1.1, we may view intelligence as the ability to solve universal two-term trade-off in diverse settings. In fact, many machine learning objectives also have this format of two-term tradeoff, as follows:

$$L = (\text{Prediction loss}) + \beta \cdot \text{Complexity} \quad (1.1)$$

For example, for unsupervised learning, the variational autoencoder [KW13] has one term that encourages a small reconstruction loss, and another term that controls the KL-divergence between the posterior and the prior. For classification and regression, the Information Bottleneck [TPB00] has one term that encourages the latent representation Z to contain as much information about the target Y as possible, and another term that controls the information contained in Z about the input X . Similar formalisms can also be seen in InfoDropout [AS18a], L1 regularizations, Kolmogorov structure functions [VV04], and Least Angle Regression [EHJ⁺04], to name just a few.

In my thesis projects, this two-term tradeoff is also a recurring theme. Specifically,

- (1) AI Physicist (Chapter 2):

$$L = \text{DL}(\text{data}|\text{model}) + \text{DL}(\text{model})$$

- (2) Information Bottleneck (Chapter 3 and 4):

$$L = I(X; Z) - \beta I(Y; Z)$$

- (3) Pareto optimal data compression for binary classification (Chapter 5), equivalent to the following form:

$$L = H(Z) - \beta I(Y; Z)$$

- (4) Causal learning with minimum mutual information regularization (Chapter 6):

$$L = \text{MSE}(f(\tilde{X}); Y) + \lambda I(\tilde{X}; X)$$

From the expression of the two-term tradeoff (Eq. 1.1), we see that there is a hyperparameter β that tunes the importance of the complexity constraint relative to the prediction loss. In one extreme, when the incentive to minimize complexity is significantly greater than that to minimize the prediction loss, we would expect that the optimum under the objective L is a trivial solution with zero complexity. In the other extreme, we would expect that the optimum of the objective minimizes the prediction loss, but may use a very complicated model or representation, which may harm generalization and robustness [SST10, AFDM16]. Between these two extremes, interesting phase transitions have been observed, where key quantities, e.g. prediction accuracy, change in a discontinuous way [Tis18, SS17b, CGTW05, AS18a, VV04, RV18]. Usually, learning is at a specific β . Here, we are instead interested in understanding the full range of learning between the above two extremes, and specifically how the phase transitions depend on the structure of the task/dataset, the objective and the model. Moreover, there has been a long history of studying phase transition in physics. The intuition and strategies in physics may help the understanding of phase transitions in learning.

1.4 Roadmap

1.4.1 Scope and structure of the thesis

With the above motivation and perspectives in mind, this thesis tries to address some key questions in the above perspectives and aspects of intelligence. We do not attempt to solve intelligence in this thesis, since it is an extremely difficult and long-term endeavor that requires the efforts of researchers around the world, still needs many breakthroughs, and may be decades away [MB16, GSD⁺18]. Instead, by laying out the perspectives and aspects, and addressing some key questions therein, we hope to make small steps towards a better understanding of the questions and improvements in answering these questions. Moreover, the lessons learned in answering those questions may help future efforts that build upon them.

As we have identified the different aspects of intelligence in Section 1.2, each chapter of the thesis addresses a key question related to one or more aspects, or addresses the second perspective of the two-term tradeoff. We start with a simple question. Suppose that an agent, without knowing anything, experiences many environments each of which may have multiple physics laws, how can it make sense of the environments, and become better at learning over time? This is a simplified version of how a human or an agent may try to make sense of different environments, each of which may have parts of underlying models in common. Although it is a simplified version, it still poses a difficult challenge. Inspired by how physicists model the world, in Chapter 2 we introduce a learning paradigm that learns and manipulates *theories* as “atoms” of learning. We introduce four algorithms, differentiable-divide-and-conquer (DDAC), Occam’s razor with minimum description length (MDL), unification and lifelong learning of theories, and integrate them into a simple AI Physicist agent, and demonstrate its capability in few-shot learning, lifelong learning, improving interpretability in a diverse set of prototypical physics environments.

Next, we study a key question in the universal two-term tradeoff, i.e. when we vary the hyperparameter β that tunes the relative strength of controlling complexity and minimizing prediction loss, how do the phase transitions depend on the dataset, the objective and the

model? We focus our attention on the Information Bottleneck (IB) objective, which provides a principled method to balance compression and prediction in representation learning via information. In Chapter 3, we derive formulas that reveal how the learnability phase transition in IB depends on the structure of the dataset, which we analyze is determined by the *conspicuous subset*, i.e. the most confident, typical and large, and imbalanced subset of the examples. The formulas also provide a tool to measure model capacity in a task-specific manner. We demonstrate that our theory and algorithm match closely with the experimentally observed onset of learning in mixture of Gaussian, MNIST and CIFAR10 datasets. In Chapter 4, we generalize our approach to study the phase transitions for the full IB trade-off curve. We provide the first formula that gives the condition for the phase transitions in the most general setting, reveal that each phase transition is finding a (nonlinear) maximum correlation component between the input and target, orthogonal to the learned representation. We present an algorithm for discovering the phase transition points. We verify that our theory and algorithm accurately predict phase transitions in categorical datasets, predict the onset of learning new classes and class difficulty in MNIST, and predict prominent phase transitions.

In Chapter 5, we take a slightly different approach in studying the two-term tradeoff. In the scenario of the tradeoff between the $I(Y; Z)$ vs. $H(Z)$ in binary classification (X is the input, Y is the target, and Z is a representation of X), we prove that we can use binning a uniformized and sorted histogram of $P(Y|X)$ to achieve the Pareto frontier of the $I(Y; Z)$ vs. $H(Z)$ tradeoff curve. We apply our technique to MNIST, FashionMNIST and CIFAR10 datasets, and illustrate how it can be interpreted as an information-theoretically optimal image clustering algorithm.

An important aspect of intelligence is its ability to learn causal relations from observations. Different from mere associations, causal relations provide a succinct way to describe the underlying mechanism of the data-generating process. The learning of causal models is important especially under shifts in environments, where environmental factors may vary but the causal relations persist [dHJL19]. Learning causal relations can also help the agent answer *interventional* and *counterfactual* questions, an important aspect of human reasoning.

Furthermore, the understanding of causal relations between components of a system also constitutes an important aspect of scientific endeavor, including physics. In Chapter 6, we ask the question: given multiple time series, without intervention, how can an agent discover the underlying causal relations? To address this question, we introduce an algorithm that combines prediction and minimizing information from the input, for exploratory causal discovery from observational time series, and demonstrate its effectiveness in synthetic, video game, breath rate vs. heart rate and *C.elegans* datasets.

Suppose that an agent has experienced and solved many similar tasks. How can it utilize its past learning to solve new tasks with few examples? In Chapter 7 we introduce the Meta-Learning Autoencoders (MeLA) architecture, which uses learning task representations for few-shot learning. When given a new task, its meta-recognition model maps the task into a task representation, and its meta-generative model maps the task representation into the weights and biases of a task-specific model. We demonstrate MeLA’s effectiveness in physics prediction tasks, and show that it compares favorably with the state-of-the-art meta-learning algorithms.

Working robustly is another important aspect of intelligence. In the scenario of binary classification, where the labels are corrupted by an unknown noise process, how can a classifier still classify accurately as if the labels are not corrupted? In Chapter 8, we introduce Rank Pruning, a robust, time-efficient, general algorithm for both binary classification with noisy labels, and estimation of the fraction of mislabeling in the training sets. We prove that under certain assumptions, Rank Pruning achieves perfect noise estimation and equivalent expected risk as learning with correct labels, and provide closed-form solutions when those assumptions are relaxed. It improves the state-of-the-art of learning with noisy labels across F1 score, AUC-PR, and Error.

Finally, in Chapter 9, I conclude the thesis, and provide prospects for future works, particularly detailing the specific directions for future work building on my thesis. This thesis is just a start, and countless opportunities lie ahead. I am excited to embark on the exciting journey of understanding intelligence and applying it for solving problems in society.

1.4.2 My contributions

Chapters 2-8 correspond to separate published or submitted papers, presented as unchanged as possible. Since they report work done together with various co-authors, I will now detail my specific contributions.

For the AI Physicist project (Chapter 2), I proposed the main architecture of the AI physicist agent, developed the differentiable divide-and-conquer, unification and lifelong learning aspects of the agent, and performed extensive experiments for improving and validating the agent. Max Tegmark developed the Occam’s-razor-with-MDL aspect of the agent and created datasets and Mathematica scripts for evaluating the agent’s performance. The project would not have succeeded without close collaboration on proofs, algorithm development and writing, occasionally late into the night.

For the Learnability for the Information Bottleneck work (Chapter 3), inspired by Max’s suggestion of starting from simple scenarios, I discovered the Information Bottleneck learnability phase transition. Through extensive discussion with Ian Fischer, I developed the main theorems that relate the learnability phase transition to the structure of the dataset. I performed the experiments on synthetic and MNIST datasets, and Ian performed the CIFAR10 experiments. For the initial draft, I wrote the method, proof and experiment sections, and Ian wrote the introduction and related work sections. Ian, Ike and Max all provided valuable feedback for significantly improving the drafts.

For the phase transition for the Information Bottleneck project (Chapter 4), which was done during my internship at Google AI, I developed the main theories through extensive discussions with Ian and performing experiments together. I wrote the majority of the draft, and both Ian and I contributed significantly to the numerical experiments.

For the Pareto-optimal data compression project (Chapter 5), I mainly contributed to the experimental part, by running experiments to compute the Pareto frontier of MNIST, FashionMNIST and CIFAR10 datasets in binary classification scenarios. Max contributed to the motivation, theorems and writing of the paper, but we extensively discussed all aspects together.

For the causal learning work (Chapter 6), which was mainly done during my internship at NVIDIA Research, I contributed to the initial idea, algorithm development, and extensive experiments for evaluating the methods, with Thomas Breuel providing lots of guidance during the process. I wrote the initial draft and subsequent re-submissions, with Thomas Breuel and Jan Kautz providing valuable feedback for improving the draft. Michael Skuhersky contributed to the *C. elegans* experiment corresponding text in the re-submissions.

For the meta-learning autoencoder (MeLA) work (Chapter 7), I contributed to the initial idea, main experiment and writing of the draft. John Peurifoy contributed to the comparison experiments of the paper. Ike provided valuable ideas for influence identification and interactive learning aspects of the MeLA architecture, and both Ike and Max provided valuable guidance, feedback during the whole process, as well as editing.

For the Rank Pruning work (Chapter 8), I contributed to the proposal of the robust noise estimator and theory developments of the work, and Curtis Northcutt contributed to the initial ideas and initial experiments of the work. Both Curtis and I contributed significantly to the main experiments and writing of the paper, and Ike provided valuable guidance and feedback during the process.

Chapter 2

AI Physicist for few-shot, lifelong learning of physics

Imagine that you are an agent. As a start, you don't know anything about how the world works, but has been endowed with a perfect mechanism to learn. Suppose that you will experience many environments each of which may have multiple physics laws, how can you make sense of the environments, and become better at learning over time?

To address this question¹², we look at how physicists model the world. Inspired by four common strategies with a long history in physics: divide-and-conquer, Occam's razor, unification and lifelong learning, we propose a novel paradigm centered around the learning and manipulation of *theories*, which parsimoniously predict both aspects of the future (from past observations) and the domain in which these predictions are accurate. This is in sharp contrast with the standard approach of using a single model to learn everything. Specifically, we propose a novel generalized-mean-loss to encourage each theory to specialize in its comparatively advantageous domain, and a differentiable description length objective to downweight bad data and "snap" learned theories into simple symbolic formulas. Theories are stored in a "theory hub", which continuously unifies learned theories and can propose theories when encountering new environments. We test our implementation, the toy "AI

¹Published in *Physical Review E*, 100 (3), 033311, "Toward an artificial intelligence physicist for unsupervised learning", Wu, Tailin and Max Tegmark. arXiv: 1810.10525.

²The code is open-sourced at github.com/tailintalent/AI_physicist.

Physicist" learning agent, on a suite of increasingly complex physics environments. From unsupervised observation of trajectories through worlds involving random combinations of gravity, electromagnetism, harmonic motion and elastic bounces, our agent typically learns faster and produces mean-squared prediction errors about a billion times smaller than a standard feedforward neural net of comparable complexity, typically recovering integer and rational theory parameters exactly. Our agent successfully identifies domains with different laws of motion also for a nonlinear chaotic double pendulum in a piecewise constant force field.

2.1 Introduction

2.1.1 Motivation

The ability to predict, analyze and parsimoniously model observations is not only central to physics, but also a goal of unsupervised machine learning, which is a key frontier in artificial intelligence (AI) research [LBH15]. Despite impressive recent progress with artificial neural nets, they still get frequently outmatched by human researchers at such modeling, suffering from two drawbacks:

1. Different parts of the data are often generated by different mechanisms in different contexts. A big model that tries to fit all the data in one environment may therefore underperform in a new environment where some mechanisms are replaced by new ones, being inflexible and inefficient at combinatorial generalization [BHB⁺18].
2. Big models are generally hard to interpret, and may not reveal succinct and universal knowledge such as Newton's law of gravitation that explains only some aspects of the data. The pursuit of "intelligible intelligence" in place of inscrutable black-box neural nets is important and timely, given the growing interest in AI interpretability from AI users and policymakers, especially for AI components involved in decisions and infrastructure where trust is important [RDT15, AOS⁺16, BBC⁺17, KDV16].

Strategy	Definition
Divide-and-conquer	Learn multiple theories each of which specializes to fit <i>part</i> of the data very well
Occam's Razor	Avoid overfitting by minimizing description length, which can include replacing fitted constants by simple integers or fractions.
Unification	Try unifying learned theories by introducing parameters
Lifelong Learning	Remember learned solutions and try them on future problems

Table 2.1: AI Physicist strategies tested.

To address these challenges, we will borrow from physics the core idea of a *theory*, which parsimoniously predicts both aspects of the future (from past observations) and also the domain in which these predictions are accurate. This suggests an alternative to the standard machine-learning paradigm of fitting a single big model to all the data: instead, learning small theories one by one, and gradually accumulating and organizing them. This paradigm suggests the four specific approaches summarized in Table 2.1, which we combine into a toy “AI Physicist” learning agent: To find individual theories from complex observations, we use the divide-and-conquer strategy with multiple theories and a novel generalized-mean loss that encourages each theory to specialize in its own domain by giving larger gradients for better-performing theories. To find simple theories that avoid overfitting and generalize well, we use the strategy known as Occam’s razor, favoring simple theories that explain a lot, using a computationally efficient approximation of the minimum-description-length (MDL) formalism. To unify similar theories found in different environments, we use the description length for clustering and then learn a “master theory” for each class of theories. To accelerate future learning, we use a lifelong learning strategy where learned theories are stored in a theory hub for future use.

2.1.2 Goals & relation to prior work

The goal of the AI Physicist learning agent presented in this paper is quite limited, and does not even remotely approach the ambition level of problem solving by human physicists. The

latter is likely to be almost as challenging as artificial general intelligence, which most AI researchers guess remains decades away [MB16, GSD⁺18]. Rather, the goal of this paper is to take a very modest but useful step in that direction, combining the four physics-inspired strategies above.

Our approach complements other work on automatic program learning, such as neural program synthesis/induction [GWD14, SsWF15, RDF15, PMS⁺16, DUB⁺17, BSXT18] and symbolic program induction [Mug91, LD94, LJK10, ESLT15, DMAT13] and builds on prior machine-learning work on divide-and-conquer [CLRS09, Für99, GSR⁺17a], network simplification [Ris78, HS93, SHS01, GMP05, HMD15, HPTD15] and continuous learning [KPR⁺17, LH18, LPR17, NLBT17]. It is often said that babies are born scientists, and there is arguably evidence for use of all of these four strategies during childhood development as well [BSXT18].

There has been significant recent progress on AI-approaches specifically linked to physics, including physical scene understanding [YSB⁺18], latent physical properties [ZLWT18, BPL⁺16, CUTT16], learning physics simulators [WZW⁺17a], physical concept discovery [IMW⁺18], an intuitive physics engine [LUTG17], and the “Sir Isaac” automated adaptive inference agent [DN15]. Our AI Physicist is different and complementary in two fundamental ways that loosely correspond to the two motivations on the first page:

1. All of these papers learn one big model to fit all the data. In contrast, the AI Physicist learns many small models applicable in different domains, using the divide-and-conquer strategy.
2. Our primary focus is not on making approximate predictions or discovering latent variables, but on near-exact predictions and complete intelligibility. From the former perspective, it is typically irrelevant if a model parameter changes by a tiny amount, but from a physics perspective, one is quite interested to learn that gravity weakens like distance to the power 2 rather than 1.99999314.

We share this focus on intelligibility with a long tradition of research on computational scientific discovery [DLT07], including the Bacon system [Lan81] and its successors [LZ89],

which induced physical laws from observations and which also used a divide-and-conquer strategy. Other work has extended this paradigm to support discovery of differential equation models from multivariate time series [DT95, BES01, LGBS03, LA15].

The rest of this paper is organized as follows. In Section 2.2, we introduce the architecture of our AI Physicist learning agent, and the algorithms implementing the four strategies. We present the results of our numerical experiments using a suite of physics environment benchmarks in Section 2.3, and discuss our conclusions in Section IV, delegating supplementary technical details to a series of appendices.

2.2 Methods

Unsupervised learning of regularities in time series can be viewed as a supervised learning problem of predicting the future from the past. This paper focuses on the task of predicting the next state vector $\mathbf{y}_t \in \mathbf{R}^d$ in a sequence from the concatenation $\mathbf{x}_t = (\mathbf{y}_{t-T}, \dots, \mathbf{y}_{t-1})$ of the last T vectors. However, our AI Physicist formalism applies more generally to learning any function $\mathbf{R}^M \mapsto \mathbf{R}^N$ from examples. In the following we first define *theory*, then introduce a unified AI Physicist architecture implementing the four aforementioned strategies.

2.2.1 Definition of Theory

A theory \mathcal{T} is a 2-tuple (f, c) , where f is a prediction function that predicts \mathbf{y}_t when \mathbf{x}_t is within the theory's domain, and c is a domain sub-classifier which takes \mathbf{x}_t as input and outputs a logit of whether \mathbf{x}_t is inside this domain. When multiple theories are present, the sub-classifier c 's outputs are concatenated and fed into a softmax function, producing probabilities for which theory is applicable. Both f and c can be implemented by a neural net or symbolic formula, and can be set to learnable during training and fixed during prediction/validation.

This definition draws inspirations from physics theories (conditional statements), such as

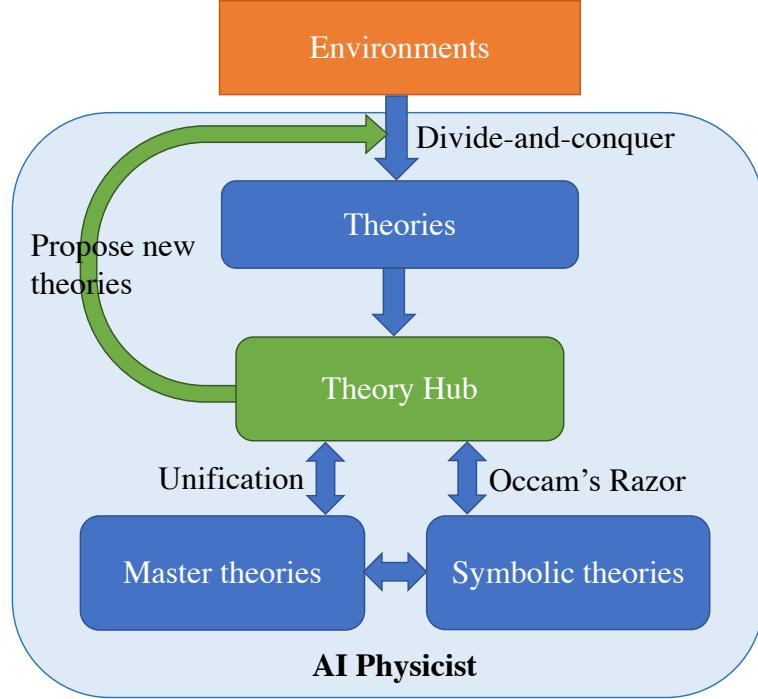


Figure 2-1: AI Physicist Architecture

“a ball not touching anything (*condition*) with vertical velocity and height (v_0, h_0) will at time t later have $\mathbf{y} \equiv (v, h) = (v_0 - gt, h_0 + v_0t - gt^2/2)$ (*prediction function*)”. For our AI Physicist, theories constitute its “atoms” of learning, as well as the building blocks for higher-level manipulations.

2.2.2 Divide-and-Conquer

2.2.3 AI Physicist Architecture Overview

Figure 2-1 illustrates the architecture of the AI Physicist learning agent. At the center is a theory hub which stores the learned and organized theories. When encountering a new environment, the agent first inspects the hub and proposes old theories that help account for parts of the data as well as randomly initialized new theories for the rest of the data. All these theories are trained via our divide-and-conquer strategy, first jointly with our generalized-mean loss then separately to fine-tune each theory in its domain (section 2.2.2). Successful theories along with the corresponding data are added to the theory hub.

The theory hub has two organizing strategies: (1) Applying Occam’s razor, it snaps the learned theories, in the form of neural nets, into simpler symbolic formulas (section 2.2.4). (2) Applying unification, it clusters and unifies the symbolic theories into master theories (section 2.2.5). The symbolic and master theories can be added back into the theory hub, improving theory proposals for new environments. The detailed AI Physicist algorithm is presented in a series of appendices. It has polynomial time complexity, as detailed in Appendix A.1.6.

Conventionally, a function \mathbf{f} mapping $\mathbf{x}_t \mapsto \mathbf{y}_t$ is learned by parameterizing \mathbf{f} by some parameter vector $\boldsymbol{\theta}$ that is adjusted to minimize a loss (empirical risk)

$$\mathcal{L} \equiv \sum_t \ell[\mathbf{f}(\mathbf{x}_t), \mathbf{y}_t], \quad (2.1)$$

where ℓ is some non-negative distance function quantifying how far each prediction is from the target, typically satisfying $\ell(\mathbf{y}, \mathbf{y}) = 0$. In contrast, a physicist observing an unfamiliar environment does typically *not* try to predict everything with one model, instead starting with an easier question: is there any part or aspect of the world that can be described? For example, when Galileo famously tried to model the motion of swinging lamps in the Pisa cathedral, he completely ignored everything else, and made no attempts to simultaneously predict the behavior of sound waves, light rays, weather, or subatomic particles. In this spirit, we allow multiple competing theories $\mathcal{T} = \{\mathbf{T}_i\} = \{(\mathbf{f}_i, c_i)\}$, $i = 1, 2, \dots, M$, to specialize in different domains, with a novel generalized-mean loss

$$\mathcal{L}_\gamma \equiv \sum_t \left(\frac{1}{M} \sum_{i=1}^M \ell[\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t]^\gamma \right)^{1/\gamma} \quad (2.2)$$

When $\gamma < 0$, the loss \mathcal{L}_γ will be dominated by whichever prediction function \mathbf{f}_i fits each data point best. This dominance is controlled by γ , with $\mathcal{L}_\gamma \rightarrow \min_i \ell[\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t]$ in the limit where $\gamma \rightarrow -\infty$. This means that the best way to minimize \mathcal{L}_γ is for each \mathbf{f}_i to specialize by further improving its accuracy for the data points where it already outperforms the other theories. The following Theorem 1 formalizes the above intuition, stating that under mild conditions for the loss function $\ell(\cdot, \cdot)$, the generalized-mean loss gives larger

gradient w.r.t. the error $|\hat{\mathbf{y}}_t - \mathbf{y}_t|$ for theories that perform better, so that a gradient-descent loss minimization encourages specialization.

Theorem 1. Let $\hat{\mathbf{y}}_t^{(i)} \equiv \mathbf{f}_i(\mathbf{x}_t)$ denote the prediction of the target \mathbf{y}_t by the function \mathbf{f}_i , $i = 1, 2, \dots, M$. Suppose that $\gamma < 0$ and $\ell(\hat{\mathbf{y}}_t, \mathbf{y}_t) = \ell(|\hat{\mathbf{y}}_t - \mathbf{y}_t|)$ for a monotonically increasing function $\ell(u)$ that vanishes on $[0, u_0]$ for some $u_0 \geq 0$, with $\ell(u)^\gamma$ differentiable and strictly convex for $u > u_0$.

Then if $0 < \ell(\hat{\mathbf{y}}_t^{(i)}, \mathbf{y}_t) < \ell(\hat{\mathbf{y}}_t^{(j)}, \mathbf{y}_t)$, we have

$$\left| \frac{\partial \mathcal{L}_\gamma}{\partial u_t^{(i)}} \right| > \left| \frac{\partial \mathcal{L}_\gamma}{\partial u_t^{(j)}} \right|, \quad (2.3)$$

where $u_t^{(i)} \equiv |\hat{\mathbf{y}}_t^{(i)} - \mathbf{y}_t|$.

Appendix A.1.7 gives the proof, and also shows that this theorem applies to mean-squared-error (MSE) loss $\ell(u) = u^2$, mean-absolute-error loss $\ell(u) = |u|$, Huber loss and our description-length loss from the next section.

We find empirically that the simple choice $\gamma = -1$ works quite well, striking a good balance between encouraging specialization for the best theory and also giving some gradient for theories that currently perform slightly worse. We term this choice \mathcal{L}_{-1} the “harmonic loss”, because it corresponds to the harmonic mean of the losses for the different theories. Based on the harmonic loss, we propose an unsupervised differentiable divide-and-conquer (DDAC) algorithm (Alg. 7 in Appendix A.1.2) that simultaneously learns prediction functions $\{\mathbf{f}_i\}$ and corresponding domain classifiers $\{c_i\}$ from observations.

Our DDAC method’s combination of multiple prediction modules into a single prediction is reminiscent of AdaBoost [FS97]. While AdaBoost gradually upweights those modules that best predict *all* the data, DDAC instead identifies complementary modules that each predict some *part* of the data best, and encourages these modules to simplify and improve by specializing on these respective parts.

2.2.4 Occam’s Razor

The principle of Occam’s razor, that simpler explanations are better, is quite popular among physicists. This preference for parsimony helped dispense with phlogiston, aether and other superfluous concepts.

Our method therefore incorporates the minimum-description-length (MDL) formalism [Ris78, GMP05], which provides an elegant mathematical implementation of Occam’s razor. It is rooted in Solomonoff’s theory of inference [Sol64] and is linked to Hutter’s AIXI approach to artificial general intelligence [Hut00]. The description length (DL) of a dataset \mathbf{D} is defined as the number of bits required to describe it. For example, if regularities are discovered that enable data compression, then the corresponding description length is defined as the number of bits of the program that produces \mathbf{D} as its output (including both the code bits and the compressed data bits). In our context of predicting a time series, this means that the description length is the number of bits required to describe the theories used plus the number of bits required to store all prediction errors. Finding the optimal data compression and hence computing the MDL is a famous hard problem that involves searching an exponentially large space, but any discovery reducing the description length is a step in the right direction, and provably avoids the overfitting problem that plagues many alternative machine-learning strategies [Ris78, GMP05].

The end-goal of the AI Physicist is to discover theories \mathcal{T} minimizing the total description length, given by

$$\text{DL}(\mathcal{T}, \mathbf{D}) = \text{DL}(\mathcal{T}) + \sum_t \text{DL}(\mathbf{u}_t), \quad (2.4)$$

where $\mathbf{u}_t = \hat{\mathbf{y}}_t - \mathbf{y}_t$ is the prediction error at time step t . By discovering simple theories that can each account for parts of the data very well, the AI Physicist strives to make both $\text{DL}(\mathcal{T})$ and $\sum_t \text{DL}(\mathbf{u}_t)$ small.

Physics has enjoyed great success in its pursuit of simpler theories using rather vague definitions of simplicity. In the this spirit, we choose to compute the description length DL not exactly, but using an approximate heuristic that is numerically efficient, and significantly simpler than more precise versions such as [Ris83], paying special attention to rational

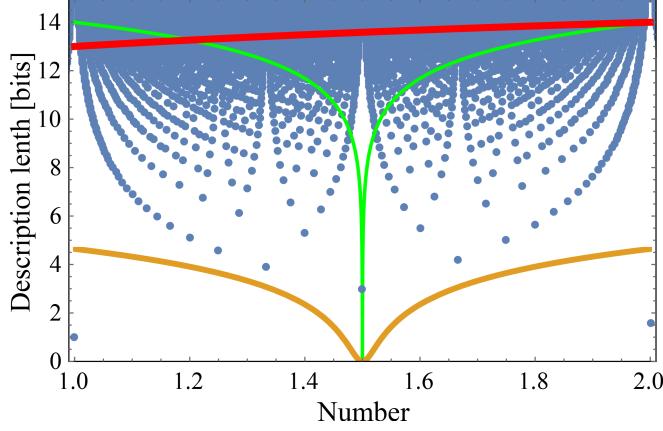


Figure 2-2: The description length DL is shown for real numbers p with $\epsilon = 2^{-14}$ (rising curve) and for rational numbers (dots). Occam’s Razor favors lower DL, and our MDL rational approximation of a real parameter p is the lowest point after taking these “model bits” specifying the approximate parameter and adding the “data bits” \mathcal{L} required to specify the prediction error made. The two symmetric curves illustrate the simple example where $\mathcal{L} = \log_+ \left(\frac{x-x_0}{\epsilon} \right)$ for $x_0 = 1.4995$, $\epsilon = 2^{-14}$ and 0.02, respectively.

numbers since they appear in many physics theories. We compute the DL of both theories \mathbf{T} and prediction errors \mathbf{u}_t as the sum of the DL of all numbers that specify them, using the following conventions for the DL of integers, rational numbers and real numbers. Our MDL implementation differs from popular machine-learning approaches whose goal is efficiency and generalizability [HvC93, HMD15, BO18] rather than intelligibility.

The number of binary digits required to specify a natural number $n = 1, 2, 3, \dots$ is approximately $\log_2 n$, so we define $\text{DL}(n) \equiv \log_2 n$ for natural numbers. For an integer m , we define

$$\text{DL}(m) \equiv \log_2(1 + |m|). \quad (2.5)$$

For a rational number $q = m/n$, the description length is the sum of that for its integer numerator and (natural number) denominator, as illustrated in Figure 2-2:

$$\text{DL}\left(\frac{m}{n}\right) = \log_2[(1 + |m|)n]. \quad (2.6)$$

For a real number r and a numerical precision floor ϵ , we define

$$\text{DL}(r) = \log_+ \left(\frac{r}{\epsilon} \right), \quad (2.7)$$

where the function

$$\log_+(x) \equiv \frac{1}{2} \log_2 (1 + x^2) \quad (2.8)$$

is plotted in Figure 2-2. Since $\log_+(x) \approx \log_2 x$ for $x \gg 1$, $\text{DL}(r)$ is approximately the description length of the integer closest to r/ϵ . Since $\log_+(x) \propto x^2$ for $x \ll 1$, $\text{DL}(r)$ simplifies to a quadratic (mean-squared-error) loss function below the numerical precision, which will prove useful below.³

Note that as long as all prediction absolute errors $|u_i| \gg \epsilon$ for some dataset,

minimizing the total description length $\sum_i \text{DL}(u_i)$ instead of the MSE $\sum_i u_i^2$ corresponds to minimizing the geometric mean instead of the arithmetic mean of the squared errors, which encourages focusing more on improving already well-fit points. $\sum_i \text{DL}(u_i)$ drops by 1 bit whenever one prediction error is halved, which can typically be achieved by fine-tuning the fit for many valid data points that are already well predicted while increasing DL for bad or extraneous points at most marginally.

For numerical efficiency, our AI Physicist minimizes the description length of equation (2.4) in two steps: 1) All model parameters are set to trainable real numbers, and the DDAC algorithm is applied to minimize the harmonic loss \mathcal{L}_{-1} with $\ell(\mathbf{u}) \equiv \sum_i \text{DL}(u_i)$ using equation (2.7) and the annealing procedure for the precision floor described in Appendix A.1.2. 2) Some model parameters are replaced by rational numbers as described below, followed by re-optimization of the other parameters. The idea behind the second step is that if a physics experiment or neural net training produces a parameter $p = 1.4999917$, it would be natural to interpret this as a hint, and to check if $p = 3/2$ gives an equally acceptable fit to the data, reducing total DL. We implement step 2 using continued fraction expansion as described in Appendix A.1.3 and illustrated in Figure 2-3.

2.2.5 Unification

Physicists aspire not only to find simple theories that explain aspects of the world accurately, but also to discover underlying similarities between theories and *unify* them. For example,

³Natural alternative definitions of $\log_+(x)$ include $\log_2(1 + |x|)$, $\log_2 \max(1, |x|)$, $(\ln 2)^{-1} \sinh^{-1} |x|$ and $(2 \ln 2)^{-1} \sinh^{-1}(x^2)$. Unless otherwise specified, we choose $\epsilon = 2^{-32}$ in our experiments.

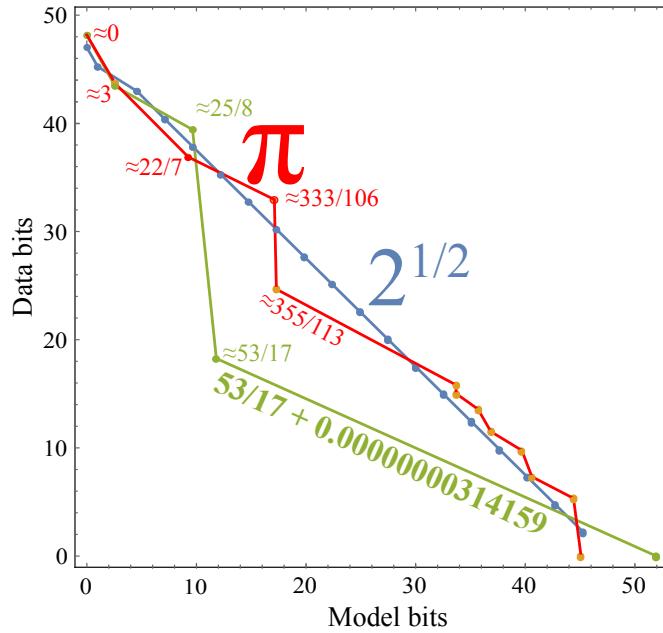


Figure 2-3: Illustration of our minimum-description-length (MDL) analysis of the parameter vector $\mathbf{p} = \{\pi, \sqrt{2}, 3.43180632382353\}$. We approximate each real number r as a fraction a_k/b_k using the first k terms of its continued fraction expansion, and for each integer $k = 1, \dots$, we plot the number of “data bits” required to encode the prediction error $r - a_k/b_k$ to 14 decimal places versus the number of “model bits” required to encode the rational approximation a_k/b_k , as described in the text. We then select the point with smallest bit sum (furthest down/left from the diagonal) as our first approximation candidate to test. Generic irrational numbers are incompressible; the total description length (model bits+data bits) is roughly independent of k as is seen for π and $\sqrt{2}$, corresponding to a line of slope -1 around which there are small random fluctuations. In contrast, the green/light grey curve (bottom) is for a parameter that is anomalously close to a rational number, and the curve reveals this by the approximation $53/17$ reducing the total description length (model+data bits) by about 16 bits.

when James Clerk Maxwell corrected and unified four key formulas describing electricity and magnetism into his eponymous equations ($dF = 0$, $d \star F = J$ in differential form notation), he revealed the nature of light and enabled the era of wireless communication.

Here we make a humble attempt to automate part of this process. The goal of the unification is to output a master theory $\mathcal{T} = \{(f_p, \cdot)\}$, such that varying the parameter vector $p \in \mathbb{R}^n$ can generate a continuum of theories (f_p, \cdot) including previously discovered ones. For example, Newton's law of gravitation can be viewed as a master theory unifying the gravitational force formulas around different planets by introducing a parameter p corresponding to planet mass. Einstein's special relativity can be viewed as a master theory unifying the approximate formulas for $v \ll c$ and $v \approx c$ motion.

We perform unification by first computing the description length $dl^{(i)}$ of the prediction function f_i (in symbolic form) for each theory i and performing clustering on $\{dl^{(i)}\}$. Unification is then achieved by discovering similarities and variations between the symbolic formulas in each cluster, retaining the similar patterns, and introducing parameters in place of the parameters that vary as detailed in Appendix A.1.4.

2.2.6 Lifelong Learning

Isaac Newton once said “If I have seen further it is by standing on the shoulders of giants”, emphasizing the utility of building on past discoveries. At a more basic level, our past experiences enable us humans to model new environments much faster than if we had to re-acquire all our knowledge from scratch. We therefore embed a lifelong learning strategy into the architecture of the AI Physicist. As shown in Fig. 2-1 and Alg. 6, the theory hub stores successfully learned theories, organizes them with our Occam's razor and unification algorithms (reminiscent of what humans do while dreaming and reflecting), and when encountering new environments, uses its accumulated knowledge to propose new theories that can explain parts of the data. This both ensures that past experiences are not forgotten and enables faster learning in novel environments. The detailed algorithms for proposing and adding theories are in Appendix A.1.5.

2.3 Results of Numerical Experiments

2.3.1 Physics Environments

We test our algorithms on two suites of benchmarks, each with increasing complexity. In all cases, the goal is to predict the two-dimensional motion as accurately as possible. One suite involves chaotic and highly nonlinear motion of a charged double pendulum in two adjacent electric fields. The other suite involves balls affected by gravity, electromagnetic fields, springs and bounce-boundaries, as exemplified in Figure 2-4. Within each spatial region, the force corresponds to a potential energy function $V \propto (ax + by + c)^n$ for some constants a, b, c , where $n = 0$ (no force), $n = 1$ (uniform electric or gravitational field), $n = 2$ (spring obeying Hooke's law) or $n = \infty$ (ideal elastic bounce), and optionally involves also a uniform magnetic field. The environments are summarized in Table A.2.

2.3.2 Numerical Results

In the mystery world example of Figure 2-4, after the DDAC algorithm 7 taking the sequence of coordinates as the only input, we see that the AI Physicist has learned to simultaneously predict the future position of the ball from the previous two, and classify without external supervision the observed inputs into four big physics domains. The predictions are seen to be more accurate deep inside the domains (tiny dots) than near boundaries (larger dots) where transitions and bounces create small domains with laws of motion that are harder to infer because of complexity and limited data. Because these small domains can be automatically inferred and eliminated once the large ones are known as described in Appendix A.1.8, all accuracy benchmarks quoted below refer to points in the large domains only.

After DDAC, the AI Physicist performs Occam's-razor-with-MDL (Alg. 8) on the learned theories. As an example, it discovers that the motion deep inside the lower-left quadrant obeys the difference equation parameterized by a learned 3-layer neural net, which after the

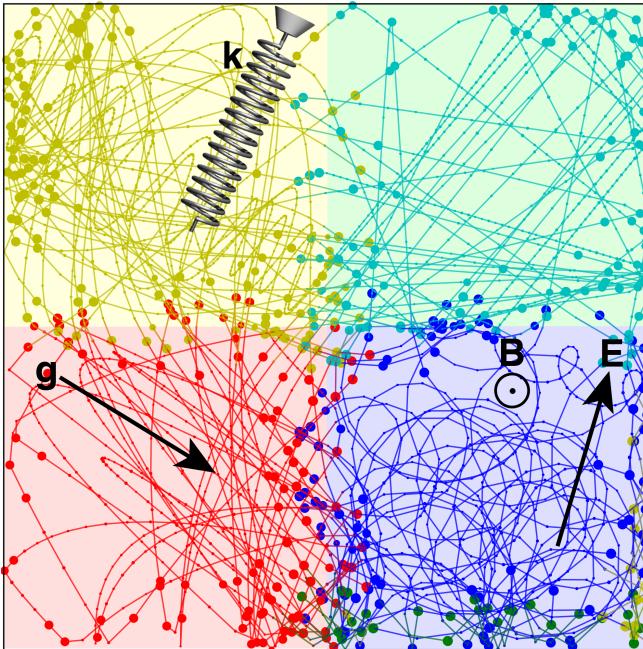


Figure 2-4: In this sample mystery world, a ball moves through a harmonic potential (upper left quadrant), a gravitational field (lower left) and an electromagnetic field (lower right quadrant) and bounces elastically from four walls. The only input to the AI Physicist is the sequence of dots (ball positions); the challenge is to learn all boundaries and laws of motion (predicting each position from the previous two). The color of each dot represents the domain into which it is classified by \mathbf{c} , and its area represents the description length of the error with which its position is predicted ($\epsilon = 10^{-6}$) after the DDAC (differentiable divide-and-conquer) algorithm; the AI Physicist tries to minimize the total area of all dots.

first collapseLayer transformation simplifies to

$$\hat{\mathbf{y}}_t = \begin{pmatrix} -0.99999994 & 0.00000006 & 1.99999990 & -0.00000012 \\ -0.00000004 & -1.00000000 & 0.00000004 & 2.00000000 \end{pmatrix} \mathbf{x}_t + \begin{pmatrix} 0.01088213 \\ -0.00776199 \end{pmatrix},$$

with $\text{DL}(\mathbf{f}) = 212.7$ and $\sum_t \text{DL}(\mathbf{u}_t) = 2524.1$. The snapping stage thereafter simplifies this to

$$\hat{\mathbf{y}}_t = \begin{pmatrix} -1 & 0 & 2 & 0 \\ 0 & -1 & 0 & 2 \end{pmatrix} \mathbf{x}_t + \begin{pmatrix} 0.010882 \\ -0.007762 \end{pmatrix}. \quad (2.9)$$

which has lower description length in both model bits ($\text{DL}(\mathbf{f}) = 55.6$) and data bits ($\sum_t \text{DL}(\mathbf{u}_t) = 2519.6$) and gets transformed to the symbolic expressions

$$\begin{aligned} \hat{x}_{t+2} &= 2x_{t+1} - x_t + 0.010882, \\ \hat{y}_{t+2} &= 2y_{t+1} - y_t - 0.007762, \end{aligned} \quad (2.10)$$

where we have written the 2D position vector $\mathbf{y} = (x, y)$ for brevity. During unification (Alg. A.1.4), the AI Physicist discovers multiple clusters of theories based on the DL of each theory, where one cluster has DL ranging between 48.86 and 55.63, which it unifies into a master theory \mathbf{f}_p with

$$\begin{aligned} \hat{x}_{t+2} &= 2x_{t+1} - x_t + p_1, \\ \hat{y}_{t+2} &= 2y_{t+1} - y_t + p_2, \end{aligned}$$

effectively discovering a “gravity” master theory out of the different types of environments it encounters. If so desired, the difference equations (2.11) can be automatically generalized to the more familiar-looking differential equations

$$\ddot{x} = g_x,$$

$$\ddot{y} = g_y,$$

where $g_i \equiv p_i(\Delta t)^2$, and both the Harmonic Oscillator Equation and Lorentz Force Law of electromagnetism can be analogously auto-inferred from other master theories learned.

Many mystery domains in our test suite involve laws of motion whose parameters include both rational and irrational numbers. To count a domain as “solved” below, we use the very stringent requirement that any rational numbers (including integers) must be discovered *exactly*, while irrational numbers must be recovered with accuracy 10^{-4} .

We apply our AI Physicist to 40 mystery worlds in sequence (Appendix A.1.9). After this training, we apply it to a suite of 40 additional worlds to test how it learns different numbers of examples. The results are shown in tables A.1 and A.2, and Table 2.2 summarizes these results using the median over worlds. For comparison, we also show results for two simpler agents with similar parameter count: a “baseline” agent consisting of a three-layer feedforward MSE-minimizing leakyReLU network and a “newborn” AI Physicist that has not seen any past examples and therefore cannot benefit from the lifelong learning strategy.

We see that the newborn agent outperforms baseline on all the tabulated measures, and that the AI Physicist does still better. Using all data, the Newborn agent and AI Physicist are able to predict with mean-squared prediction error below 10^{-13} , more than nine orders of magnitude below baseline. Moreover, the Newborn and AI Physicist agents are able to simultaneously learn the domain classifiers with essentially perfect accuracy, without external supervision. Both agents are able to solve above 90% of all the 40 mystery worlds according to our stringent criteria.

The main advantage of the AI Physicist over the Newborn agent is seen to be its learning speed, attaining given accuracy levels faster, especially during the early stage of learning. Remarkably, for the subsequent 40 worlds, the AI Physicist reaches 0.01 MSE within 35 epochs using as little as 1% of the data, performing almost as well as with 50% of the data much better than the Newborn agent. This illustrates that the lifelong learning strategy enables the AI Physicist to learn much faster in novel environments with less data. This is much like an experienced scientist can solve new problems way faster than a beginner by building on prior knowledge about similar problems.

Our double-pendulum mysteries (Appendix A.1.9) are more challenging for all the agents,

Benchmark	Baseline	Newborn	AI Physicist
\log_{10} mean-squared error	-3.89	-13.95	-13.88
Classification accuracy	67.56%	100.00%	100.00%
Fraction of worlds solved	0.00%	90.00%	92.50%
Description length for f	11,338.7	198.9	198.9
Epochs until 10^{-2} MSE	95	83	15
Epochs until 10^{-4} MSE	6925	330	45
Epochs until 10^{-6} MSE	∞	5403	3895
Epochs until 10^{-8} MSE	∞	6590	5100
<hr/>			
\log_{10} MSE			
using 100% of data	-3.78	-13.89	-13.89
using 50% of data	-3.84	-13.76	-13.81
using 10% of data	-3.16	-7.38	-10.54
using 5% of data	-3.06	-6.06	-6.20
using 1% of data	-2.46	-3.69	-3.95
<hr/>			
Epochs until 10^{-2} MSE			
using 100% of data	95	80	15
using 50% of data	190	152.5	30
using 10% of data	195	162.5	30
using 5% of data	205	165	30
using 1% of data	397.5	235	35

Table 2.2: Summary of numerical results, taking the median over 40 mystery environments from Table A.1 (top part) and on 40 novel environments with varying fraction of random examples (bottom parts), where each world is run with 10 random initializations and taking the best performance. Accuracies refer to big regions only.

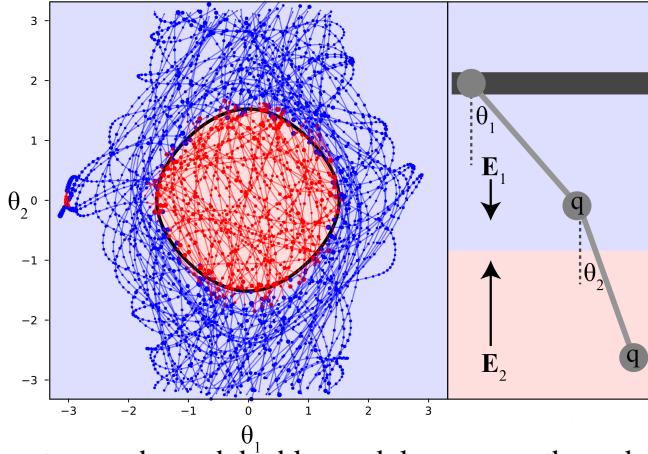


Figure 2-5: In this mystery, a charged double pendulum moves through two different electric fields E_1 and E_2 , with a domain boundary corresponding to $\cos \theta_1 + \cos \theta_2 = 1.05$ (the black curve above left, where the lower charge crosses the E -field boundary). The color of each dot represents the domain into which it is classified by a Newborn agent, and its area represents the description length of the error with which its position is predicted, for a precision floor $\epsilon \approx 0.006$. In this world, the Newborn agent has a domain prediction accuracy of 96.5%.

because the motion is more nonlinear and indeed chaotic. Although none of our double-pendulum mysteries get exactly solved according to our very stringent above-mentioned criterion, Figure 2-5 illustrates that the Newborn agent does a good job: it discovers the two domains and classifies points into them with an accuracy of 96.5%. Overall, the Newborn agent has a median best accuracy of 91.0% compared with the baseline of 76.9%. The MSE prediction error is comparable to the baseline performance ($\sim 4 \times 10^{-4}$) in the median, since both architectures have similar large capacity. We analyze this challenge and opportunities for improvement below.

2.4 Conclusions

We have presented a toy “AI Physicist” unsupervised learning agent centered around the learning and manipulation of theories, which in polynomial time learns to parsimoniously predict both aspects of the future (from past observations) and the domain in which these predictions are accurate.

2.4.1 Key findings

Testing it on a suite of mystery worlds involving random combinations of gravity, electromagnetism, harmonic motion and elastic bounces, we found that its divide-and-conquer and Occam’s razor strategies effectively identified domains with different laws of motion and reduced the mean-squared prediction error billionfold, typically recovering integer and rational theory parameters exactly. These two strategies both encouraged prediction functions to specialize: the former on the domains they handled best, and the latter on the data points within their domain that they handled best. Adding the lifelong learning strategy greatly accelerated learning in novel environments.

2.4.2 What has been learned?

Returning to the broader context of unsupervised learning from Section 2.1 raises two important questions: what is the *difficulty* of the problems that our AI physicist solved, and what is the *generality* of our paradigm?

In terms of *difficulty*, our solved physics problems are clearly on the easier part of the spectrum, so if we were to have faced the *supervised* learning problem where the different domains were pre-labeled, the domain learning would have been a straightforward classification task and the forecasting task could have been easily solved by a standard feedforward neural network. Because the real world is generally unlabeled, we instead tackled the more difficult problem where boundaries of multiple domains had to be learned concurrently with the dynamical evolution rules in a fully unsupervised fashion. The dramatic performance improvement over a traditional neural network seen in Table A.1 reflects the power of the divide-and-conquer and Occam’s razor strategies, and their robustness is indicated by the fact that unsupervised domain discovery worked well even for the two-field non-linear double-pendulum system whose dynamic is notoriously chaotic and whose domain boundary is the curved rhomboid $\cos \theta_1 + \cos \theta_2 = 1.05$.

In terms of *generality*, our core contribution lies in the AI physicist paradigm we propose (combining divide-and-conquer, Occam’s razor, unification and lifelong learning), not in

the specific implementation details. Here we draw inspiration from the history of the Turing Machine: Turing’s initial implementation of a universal computer was very inefficient for all but toy problems, but his framework laid out the essential architectural components that subsequent researchers developed into today’s powerful computers. What has been learned is that our AI physicist paradigm outperforms traditional deep learning on a test suite of problems even though it is a fully general paradigm that is was not designed specifically for these problems. For example, it is defined to work for an arbitrary number of input spatial dimensions, spatial domains, past time steps used, boundaries of arbitrary shapes, and evolution laws of arbitrary complexity.

From the above-mentioned successes and failures of our paradigm, we have also learned about promising opportunities for improvement of the implementation which we will now discuss. First of all, the more modest success in the double-pendulum experiments illustrated the value of learned theories being *simple*: if they are highly complex, they are less likely to unify or generalize to future environments, and the correspondingly complex baseline model will have less incentive to specialize because it has enough expressive power to approximate the motion in all domains at once. It will therefore be valuable to improve techniques for simplifying complex learned neural nets. The specific implementation details for the Occam’s Razor toolkit would then change, but the principle and numerical objective would remain the same: reducing their total description length from equation (2.4). There are many promising opportunities for this using techniques from the Monte-Carlo-Markov-Chain-based and genetic techniques [RMS⁺17], reinforcement learning [ZL16, BGNR16] and symbolic regression [SL09, UT19] literature to simplify and shrink the model architecture. Also, it will be valuable and straightforward to generalize our implementation to simplify not only the prediction functions, but also the classifiers, for example to find sharp domain boundaries composed of hyperplanes or other simple surfaces.

Analogously, there are many ways in which the unification and life-long learning toolkits can be improved while staying within our AI physicist paradigm. For example, unification can undoubtedly be improved by using more sophisticated clustering techniques for grouping the learned theories with similar ones. Life-long learning can probably be made more efficient

by using better methods for determining which previous theories to try when faced by new data, for example by training a separate neural network to perform this prediction task.

2.4.3 Outlook

In summary, these and other improvements to the algorithms that implement our AI Physicist paradigm could enable future unsupervised learning agents to learn simpler and more accurate models faster from fewer examples, and also to discover accurate symbolic expressions for more complicated physical systems. More broadly, AI has been used with great success to tackle problems in diverse areas of physics, ranging from quantum state reconstruction [CTMA19] to phase transitions [CM17, Wan16, VNLH17], planetary dynamics [LK18] and particle physics [BSW14]. We hope that building on the ideas of this paper may one day enable AI to help us discover entirely novel physical theories from data.

Acknowledgements: This work was supported by the The Casey and Family Foundation, the Ethics and Governance of AI Fund, the Foundational Questions Institute and the Rothberg Family Fund for Cognitive Science. We thank Isaac Chuang, John Peurifoy and Marin Soljačić for helpful discussions and suggestions, and the Center for Brains, Minds, and Machines (CBMM) for hospitality.

Chapter 3

Learnability phase transition: onset of learning

As I have stated in the Introduction (Chapter 1), the ability to solve a universal two-term tradeoff: a term on task performance and a term on controlling complexity, provides an important perspective in intelligence. When the relative strength between the two terms vary, how do the learning bahave? In this chapter, and Chapter 4 and Chapter 5, we set out to address this question. Specifically, we study the two-term tradeoff in the Information Bottleneck paradigm [TPB00], whose objective is based on information, and provides an insightful and principled approach for balancing compression and prediction for representation learning. This chapter and Chapter 4 will focus on the phase transitions, which like the phase transitions in physics, have key quantities for the system changing in a discontinuous way. Chapter 5 will directly study the two-term tradeoff curve in the binary classification cases, where we introduce a method for directly compute the Pareto frontier.

In this chapter¹, we focus on understanding the learnability transition in IB. The IB objective $I(X; Z) - \beta I(Y; Z)$ employs a Lagrange multiplier β to tune the trade-off between compression and prediction. However, in practice, not only is β chosen empirically without theoretical guidance, there is also a lack of theoretical understanding between β , learnability,

¹This extended version is published in *Entropy* 2019, 21(10), 924, “Learnability for the information bottleneck”, Wu, Tailin, Ian Fischer, Isaac Chuang, Max Tegmark. Also published at Conference on Uncertainty in Artificial Intelligence (UAI 2019) and presented at ICLR 2019 LLD workshop as spotlight. arXiv: 1907.07331.

the intrinsic nature of the dataset and model capacity. In this chapter, we show that if β is improperly chosen, learning cannot happen – the trivial representation $P(Z|X) = P(Z)$ becomes the global minimum of the IB objective. We show how this can be avoided, by identifying a sharp phase transition between the unlearnable and the learnable which arises as β is varied. This phase transition defines the concept of IB-Learnability. We prove several sufficient conditions for IB-Learnability, which provides theoretical guidance for choosing a good β . We further show that IB-learnability is determined by the largest *confident*, *typical*, and *imbalanced subset* of the examples (the *conspicuous subset*), and discuss its relation with model capacity. We give practical algorithms to estimate the minimum β for a given dataset. We also empirically demonstrate our theoretical conditions with analyses of synthetic datasets, MNIST, and CIFAR10.

3.1 Introduction

[TPB00] introduced the *Information Bottleneck* (IB) objective function which learns a representation Z of observed variables (X, Y) that retains as little information about X as possible, but simultaneously captures as much information about Y as possible:

$$\min \text{IB}_\beta(X, Y; Z) = \min[I(X; Z) - \beta I(Y; Z)] \quad (3.1)$$

$I(\cdot)$ is the mutual information. The hyperparameter β controls the trade-off between compression and prediction, in the same spirit as Rate-Distortion Theory [Sha48b], but with a learned representation function $P(Z|X)$ that automatically captures some part of the “semantically meaningful” information, where the semantics are determined by the observed relationship between X and Y . The IB framework has been extended to and extensively studied in a variety of scenarios, including Gaussian variables [CGTW05], meta-Gaussians [RR12], continuous variables via variational methods [AFDM16, CMT16, Fis18], deterministic scenarios [SS17a, KTVK19], geometric clustering [SS17b], and is used for learning invariant and disentangled representations in deep neural nets [AS18a, AS18b].

From the IB objective (Eq. 3.1) we see that when $\beta \rightarrow 0$ it will encourage $I(X; Z) = 0$

which leads to a trivial representation Z that is independent of X , while when $\beta \rightarrow +\infty$, it reduces to a maximum likelihood objective (e.g. in classification, it reduces to cross-entropy loss). Therefore, as we vary β from 0 to $+\infty$, there must exist a point β_0 at which IB starts to learn a nontrivial representation where Z contains information about X .

As an example, we train multiple variational information bottleneck (VIB) models on binary classification of MNIST [LBBH98] digits 0 and 1 with 20% label noise at different β . The accuracy vs. β is shown in Fig. 3-1. We see that when $\beta < 3.25$, no learning happens, and the accuracy is the same as random guessing. Beginning with $\beta > 3.25$, there is a clear phase transition where the accuracy sharply increases, indicating the objective is able to learn a nontrivial representation. In general, we observe that different datasets and model capacity will result in different β_0 at which IB starts to learn a nontrivial representation. How does β_0 depend on the aspects of the dataset and model capacity, and how can we estimate it? What does an IB model learn at the onset of learning? Answering these questions may provide a deeper understanding of IB in particular, and learning on two observed variables in general.

In this work, we begin to answer the above questions. Specifically:

- We introduce the concept of *IB-Learnability*, and show that when we vary β , the IB objective will undergo a phase transition from the inability to learn to the ability to learn (Section 3.3).
- Using the second-order variation, we derive sufficient conditions for IB-Learnability, which provide upper bounds for the learnability threshold β_0 (Section 3.4).
- We show that IB-Learnability is determined by the largest *confident*, *typical*, and *imbalanced subset* of the examples (the *conspicuous subset*), reveal its relationship with the slope of the Pareto frontier at the origin on the information plane $I(X; Z)$ vs. $I(Y; Z)$, and discuss its relation to model capacity (Section 3.5).
- We prove a deep relationship between IB-Learnability, our upper bounds on β_0 , the hypercontractivity coefficient, the contraction coefficient, and the maximum correlation (Section 3.5).

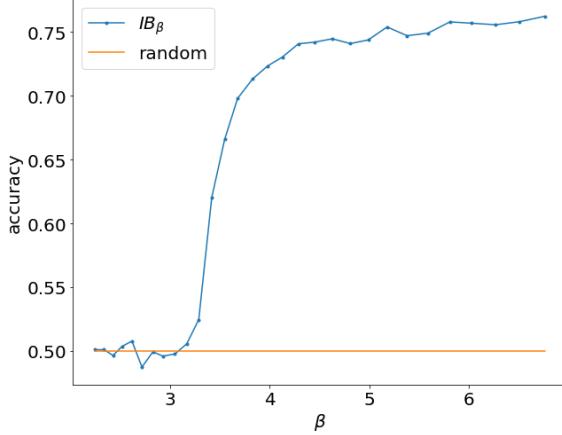


Figure 3-1: Accuracy for binary classification of MNIST digits 0 and 1 with 20% label noise and varying β . No learning happens for models trained at $\beta < 3.25$.

We also present an algorithm for estimating the onset of IB-Learnability and the conspicuous subset, which provide us with a tool for understanding a key aspect of the learning problem (X, Y) (Section 3.6).

Finally, we use our main results to demonstrate on synthetic datasets, MNIST [LBBH98] and CIFAR10 [KH09] that the theoretical prediction for IB-Learnability closely matches experiment, and show the conspicuous subset our algorithm discovers (Section 3.7).

3.2 Related Work

The seminal IB work [TPB00] provides a tabular method for exactly computing the optimal encoder distribution $P(Z|X)$ for a given β and cardinality of the discrete representation, $|Z|$. They did not consider the IB learnability problem as addressed in this work.

[CGTW05] presents the Gaussian Information Bottleneck (GIB) for learning a multivariate Gaussian representation Z of (X, Y) , assuming that both X and Y are also multivariate Gaussians. Under GIB, they derive analytic formula for the optimal representation as a noisy linear projection to eigenvectors of the normalized regression matrix $\Sigma_{x|y}\Sigma_x^{-1}$, and the learnability threshold β_0 is then given by $\beta_0 = \frac{1}{1-\lambda_1}$ where λ_1 is the largest eigenvalue of the matrix $\Sigma_{x|y}\Sigma_x^{-1}$. This work provides deep insights about relations between the dataset,

β_0 and optimal representations in the Gaussian scenario, but the restriction to multivariate Gaussian datasets limits the generality of the analysis

Another analytic treatment of IB is given in [RR12], which reformulates the objective in terms of the copula functions. As with the GIB approach, this formulation restricts the form of the data distributions – the copula functions for the joint distribution (X, Y) are assumed to be known, which is unlikely in practice.

[SS17a] presents the Deterministic Information Bottleneck (DIB), which minimizes the coding cost of the representation, $H(Z)$, rather than the transmission cost, $I(X; Z)$ as in IB. This approach learns hard clusterings with different code entropies that vary with β . In this case, it is clear that a hard clustering with minimal $H(Z)$ will result in a single cluster for all of the data, which is the DIB trivial solution. No analysis is given beyond this fact to predict the actual onset of learnability, however.

The first amortized IB objective is in the Variational Information Bottleneck (VIB) of [AFDM16]. VIB replaces the exact, tabular approach of IB with variational approximations of the classifier distribution ($P(Y|Z)$) and marginal distribution ($P(Z)$). This approach cleanly permits learning a stochastic encoder, $P(Z|X)$, that is applicable to any $x \in \mathcal{X}$, rather than just the particular X seen at training time. The cost of this flexibility is the use of variational approximations that may be less expressive than the tabular method. Nevertheless, in practice, VIB learns easily and is simple to implement, so we rely on VIB models for our experimental confirmation.

Closely related to IB is the recently proposed Conditional Entropy Bottleneck (CEB) [Fis18]. CEB attempts to explicitly learn the Minimum Necessary Information (MNI), defined as the point in the information plane where $I(X; Y) = I(X; Z) = I(Y; Z)$. The MNI point may not be achievable even in principle for a particular dataset. However, the CEB objective provides an explicit estimate of how closely the model is approaching the MNI point by observing that a necessary condition for reaching the MNI point occurs when $I(X; Z|Y) = 0$. The CEB objective $I(X; Z|Y) - \gamma I(Y; Z)$ is equivalent to IB at $\gamma = \beta + 1$, so our analysis of IB-Learnability applies equally to CEB.

[KTVK19] shows that when Y is a deterministic function of X , the “corner point” of the IB

curve (where $I(X; Y) = I(X; Z) = I(Y; Z)$) is the unique optimizer of the IB objective for all $0 < \beta' < 1$ (with the parameterization of [KTVK19], $\beta' = 1/\beta$), which they consider to be a “trivial solution”.

However, their use of the term “trivial solution” is distinct from ours. They are referring to the observation that all points on the IB curve contain uninteresting interpolations between two different but valid solutions on the optimal frontier, rather than demonstrating a non-trivial trade-off between compression and prediction as expected when varying the IB Lagrangian. Our use of “trivial” refers to whether IB is capable of learning at all given a certain dataset and value of β .

[AS18b] apply the IB Lagrangian to the weights of a neural network, yielding InfoDropout. In [AS18a], the authors give a deep and compelling analysis of how the IB Lagrangian can yield invariant and disentangled representations. They do not, however, consider the question of the onset of learning, although they are aware that not all models will learn a non-trivial representation. More recently, [AMS18] repurpose the InfoDropout IB Lagrangian as a Kolmogorov Structure Function to analyze the ease with which a previously-trained network can be fine-tuned for a new task. While that work is tangentially related to learnability, the question it addresses is substantially different from our investigation of the onset of learning.

Our work is also closely related to the hypercontractivity coefficient [AGKN13, PW17], defined as $\sup_{Z-X-Y} \frac{I(Y; Z)}{I(X; Z)}$, which by definition equals the inverse of β_0 , our IB-learnability threshold. In [AGKN13], the authors prove that the hypercontractivity coefficient equals the contraction coefficient $\eta_{KL}(P_{Y|X}, P_X)$, and [KGK⁺17] propose a practical algorithm to estimate $\eta_{KL}(P_{Y|X}, P_X)$, which provides a measure for potential influence in the data. Although our goal is different, the sufficient conditions we provide for IB-Learnability are also lower bounds for the hypercontractivity coefficient.

3.3 IB-Learnability

We are given instances of (x, y) drawn from a distribution with probability (density) $P(X, Y)$ with support of $\mathcal{X} \times \mathcal{Y}$, where unless otherwise stated, both X and Y can be discrete or

continuous variables. (X, Y) is our *training data*, and may be characterized by different types of noise. The nature of this training data and the choice of β will be sufficient to predict the transition from unlearnable to learnable.

We can learn a representation Z of X with conditional probability² $p(z|x)$, such that X, Y, Z obey the Markov chain $Z \leftarrow X \leftrightarrow Y$. Eq. 3.1 above gives the IB objective with Lagrange multiplier β , $\text{IB}_\beta(X, Y; Z)$, which is a functional of $p(z|x)$: $\text{IB}_\beta(X, Y; Z) = \text{IB}_\beta[p(z|x)]$. The IB learning task is to find a conditional probability $p(z|x)$ that minimizes $\text{IB}_\beta(X, Y; Z)$. The larger β , the more the objective favors making a good prediction for Y . Conversely, the smaller β , the more the objective favors learning a concise representation.

How can we select β such that the IB objective learns a useful representation? In practice, the selection of β is done empirically. Indeed, [TPB00] recommends “sweeping β ”. In this paper, we provide theoretical guidance for choosing β by introducing the concept of IB-Learnability and providing a series of IB-learnable conditions.

Def. 1. *(X, Y) is IB_β -learnable if there exists a Z given by some $p_1(z|x)$, such that $\text{IB}_\beta(X, Y; Z)|_{p_1(z|x)} < \text{IB}_\beta(X, Y; Z)|_{p(z|x)=p(z)}$, where $p(z|x) = p(z)$ characterizes the trivial representation where $Z = Z_{\text{trivial}}$ is independent of X .*

If $(X; Y)$ is IB_β -learnable, then when $\text{IB}_\beta(X, Y; Z)$ is globally minimized, it will *not* learn a trivial representation. On the other hand, if $(X; Y)$ is not IB_β -learnable, then when $\text{IB}_\beta(X, Y; Z)$ is globally minimized, it may learn a trivial representation.

Trivial solutions. Definition 1 defines trivial solutions in terms of representations where $I(X; Z) = I(Y; Z) = 0$. Another type of trivial solution occurs when $I(X; Z) > 0$ but $I(Y; Z) = 0$. This type of trivial solution is not directly achievable by the IB objective, as $I(X; Z)$ is minimized, but it can be achieved by construction or by chance. It is possible that starting learning from $I(X; Z) > 0, I(Y; Z) = 0$ could result in access to non-trivial solutions not available from $I(X; Z) = 0$. We do not attempt to investigate this type of trivial solution in this work.

²We use capital letters X, Y, Z for random variables and lowercase x, y, z to denote the instance of variables, with $P(\cdot)$ and $p(\cdot)$ denoting their probability or probability density, respectively.

Necessary condition for IB_β -Learnability. From Definition 1, we can see that IB_β -Learnability for any dataset $(X; Y)$ requires $\beta > 1$. In fact, from the Markov chain $Z \leftarrow X \leftrightarrow Y$, we have $I(Y; Z) \leq I(X; Z)$ via the data-processing inequality. If $\beta \leq 1$, then since $I(X; Z) \geq 0$ and $I(Y; Z) \geq 0$, we have that $\min(I(X; Z) - \beta I(Y; Z)) = 0 = \text{IB}_\beta(X, Y; Z_{\text{trivial}})$. Hence (X, Y) is not IB_β -learnable for $\beta \leq 1$.

Due to the reparameterization invariance of mutual information, we have the following theorem for IB_β -Learnability:

Lemma 1.1. *Let $X' = g(X)$ be an invertible map (if X is a continuous variable, g is additionally required to be continuous). Then (X, Y) and (X', Y) have the same IB_β -Learnability.*

The proof for Lemma 1.1 is in Appendix A.2.2. Lemma 1.1 implies a favorable property for any condition for IB_β -Learnability: the condition should be invariant to invertible mappings of X . We will inspect this invariance in the conditions we derive in the following sections.

3.4 Sufficient conditions for IB_β -Learnability

Given (X, Y) , how can we determine whether it is IB_β -learnable? To answer this question, we derive a series of sufficient conditions for IB_β -Learnability, starting from its definition. The conditions are in increasing order of practicality, while sacrificing as little generality as possible.

Firstly, Theorem 2 characterizes the IB_β -Learnability range for β , with proof in Appendix A.2.3:

Theorem 2. *If (X, Y) is IB_{β_1} -learnable, then for any $\beta_2 > \beta_1$, it is IB_{β_2} -learnable.*

Based on Theorem 2, the range of β such that (X, Y) is IB_β -learnable has the form $\beta \in (\beta_0, +\infty)$. Thus, β_0 is the *threshold* of IB -Learnability.

Lemma 2.1. *$p(z|x) = p(z)$ is a stationary solution for $\text{IB}_\beta(X, Y; Z)$.*

The proof in Appendix A.2.6 shows that both first-order variations $\delta I(X; Z) = 0$ and $\delta I(Y; Z) = 0$ vanish at the trivial representation $p(z|x) = p(z)$, so $\delta \text{IB}_\beta[p(z|x)] = 0$ at the trivial representation.

Lemma 2.1 yields our strategy for finding sufficient conditions for learnability: find conditions such that $p(z|x) = p(z)$ is not a local minimum for the functional $\text{IB}_\beta[p(z|x)]$. Based on the necessary condition for the minimum (Appendix A.2.4), we have the following theorem³:

Theorem 3 (Suff. Cond. 1). *A sufficient condition for (X, Y) to be IB_β -learnable is that there exists a perturbation function⁴ $h(z|x)$ with $\int h(z|x) dz = 0$, such that the second-order variation $\delta^2 \text{IB}_\beta[p(z|x)] < 0$ at the trivial representation $p(z|x) = p(z)$.*

The proof for Theorem 3 is given in Appendix A.2.4. Intuitively, if $\delta^2 \text{IB}_\beta[p(z|x)]|_{p(z|x)=p(z)} < 0$, we can always find a $p'(z|x) = p(z|x) + \epsilon \cdot h(z|x)$ in the neighborhood of the trivial representation $p(z|x) = p(z)$, such that $\text{IB}_\beta[p'(z|x)] < \text{IB}_\beta[p(z|x)]$, thus satisfying the definition for IB_β -Learnability.

To make Theorem 3 more practical, we perturb $p(z|x)$ around the trivial solution $p'(z|x) = p(z|x) + \epsilon \cdot h(z|x)$, and expand $\text{IB}_\beta[p(z|x) + \epsilon \cdot h(z|x)] - \text{IB}_\beta[p(z|x)]$ to the second order of ϵ . We can then prove Theorem 4:

Theorem 4 (Suff. Cond. 2). *A sufficient condition for (X, Y) to be IB_β -learnable is X and Y are not independent, and*

$$\beta > \inf_{h(x)} \beta_0[h(x)] \tag{3.2}$$

where the functional $\beta_0[h(x)]$ is given by

$$\beta_0[h(x)] = \frac{\mathbb{E}_{x \sim p(x)}[h(x)^2] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}{\mathbb{E}_{y \sim p(y)} \left[(\mathbb{E}_{x \sim p(x|y)}[h(x)])^2 \right] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}$$

³The theorems in this paper deal with learnability w.r.t. true mutual information. If parameterized models are used to approximate the mutual information, the limitation of the model capacity will translate into more uncertainty of Y given X , viewed through the lens of the model.

⁴so that the perturbed probability (density) is $p'(z|x) = p(z|x) + \epsilon \cdot h(z|x)$. Also, for integrals, whenever a variable W is discrete, we can simply replace the integral ($\int \cdot dw$) by summation ($\sum_w \cdot$).

Moreover, we have that $(\inf_{h(x)} \beta[h(x)])^{-1}$ is a lower bound of the slope of the Pareto frontier in the information plane $I(Y; Z)$ vs. $I(X; Z)$ at the origin.

The proof is given in Appendix A.2.7, which also shows that if $\beta > \inf_{h(x)} \beta_0[h(x)]$ in Theorem 4 is satisfied, we can construct a perturbation function $h(z|x) = h^*(x)h_2(z)$ with $h^*(x) = \arg \min_{h(x)} \beta_0[h(x)]$, $\int h_2(z)dz = 0$, $\int \frac{h_2^2(z)}{p(z)}dz > 0$ for some $h_2(z)$, such that $h(z|x)$ satisfies Theorem 3. It also shows that the converse is true: if there exists $h(z|x)$ such that the condition in Theorem 3 is true, then Theorem 4 is satisfied⁵, i.e. $\beta > \inf_{h(x)} \beta_0[h(x)]$. Moreover, letting the perturbation function $h(z|x) = h^*(x)h_2(z)$ at the trivial solution, we have

$$p_\beta(y|x) = p(y) + \epsilon^2 C_z (h^*(x) - \bar{h}_x^*) \int p(x,y) (h^*(x) - \bar{h}_x^*) dx \quad (3.3)$$

where $p_\beta(y|x)$ is the estimated $p(y|x)$ by IB for a certain β , $\bar{h}_x^* = \int h^*(x)p(x)dx$, and $C_z = \int \frac{h_2^2(z)}{p(z)}dz > 0$ is a constant. This shows how the $p_\beta(y|x)$ by IB explicitly depends on $h^*(x)$ at the onset of learning. The proof is provided in Appendix A.2.8.

Theorem 4 suggests a method to estimate β_0 : we can parameterize $h(x)$ e.g. by a neural network, with the objective of minimizing $\beta_0[h(x)]$. At its minimization, $\beta_0[h(x)]$ provides an upper bound for β_0 , and $h(x)$ provides a *soft clustering* of the examples corresponding to a nontrivial perturbation of $p(z|x)$ at $p(z|x) = p(z)$ that minimizes $\text{IB}_\beta[p(z|x)]$.

Alternatively, based on the property of $\beta_0[h(x)]$, we can also use a specific functional form for $h(x)$ in Eq. (3.2), and obtain a stronger sufficient condition for IB_β -Learnability. But we want to choose $h(x)$ as near to the infimum as possible. To do this, we note the following characteristics for the R.H.S of Eq. (3.2):

- We can set $h(x)$ to be nonzero if $x \in \Omega_x$ for some region $\Omega_x \subset \mathcal{X}$ and 0 otherwise.

⁵We do not claim that any $h(z|x)$ satisfying Theorem 3 can be decomposed to $h^*(x)h_2(z)$ at the onset of learning. But from the equivalence of Theorems 3 and 4 as explained above, when there exists an $h(z|x)$ such that Theorem 3 is satisfied, we can always construct an $h'(z|x) = h^*(x)h_2(z)$ that also satisfies Theorem 3.

Then we obtain the following sufficient condition:

$$\beta > \inf_{h(x), \Omega_x \subset \mathcal{X}} \frac{\frac{\mathbb{E}_{x \sim p(x), x \in \Omega_x} [h(x)^2]}{\left(\mathbb{E}_{x \sim p(x), x \in \Omega_x} [h(x)] \right)^2} - 1}{\int \frac{dy}{p(y)} \left(\frac{\mathbb{E}_{x \sim p(x), x \in \Omega_x} [p(y|x)h(x)]}{\mathbb{E}_{x \sim p(x), x \in \Omega_x} [h(x)]} \right)^2 - 1} \quad (3.4)$$

- The numerator of the R.H.S. of Eq. (3.4) attains its minimum when $h(x)$ is a constant within Ω_x . This can be proved using the Cauchy-Schwarz inequality: $\langle u, u \rangle \langle v, v \rangle \geq \langle u, v \rangle^2$, setting $u(x) = h(x)\sqrt{p(x)}$, $v(x) = \sqrt{p(x)}$, and defining the inner product as $\langle u, v \rangle = \int u(x)v(x)dx$. Therefore, the numerator of the R.H.S. of Eq. (3.4) $\geq \frac{1}{\int_{x \in \Omega_x} p(x)} - 1$, and attains equality when $\frac{u(x)}{v(x)} = h(x)$ is constant.

Based on these observations, we can let $h(x)$ be a nonzero constant inside some region $\Omega_x \subset \mathcal{X}$ and 0 otherwise, and the infimum over an arbitrary function $h(x)$ is simplified to infimum over $\Omega_x \subset \mathcal{X}$, and we obtain a sufficient condition for IB_β -Learnability, which is a key result of this paper:

Theorem 5 (Conspicuous Subset Suff. Cond.). *A sufficient condition for (X, Y) to be IB_β -learnable is X and Y are not independent, and*

$$\beta > \inf_{\Omega_x \subset \mathcal{X}} \beta_0(\Omega_x) \quad (3.5)$$

where

$$\beta_0(\Omega_x) = \frac{\frac{1}{p(\Omega_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]}$$

Ω_x denotes the event that $x \in \Omega_x$, with probability $p(\Omega_x)$.

$(\inf_{\Omega_x \subset \mathcal{X}} \beta_0(\Omega_x))^{-1}$ gives a lower bound of the slope of the Pareto frontier in the information plane $I(Y; Z)$ vs. $I(X; Z)$ at the origin.

The proof is given in Appendix A.2.9. In the proof we also show that this condition is invariant to invertible mappings of X .

3.5 Discussion

3.5.1 The conspicuous subset determines β_0 .

From Eq. (3.5), we see that three characteristics of the subset $\Omega_x \subset \mathcal{X}$ lead to low β_0 : **(1) confidence:** $p(y|\Omega_x)$ is large; **(2) typicality and size:** the number of elements in Ω_x is large, or the elements in Ω_x are typical, leading to a large probability of $p(\Omega_x)$; **(3) imbalance:** $p(y)$ is small for the subset Ω_x , but large for its complement. In summary, β_0 will be determined by the largest *confident*, *typical* and *imbalanced subset* of examples, or an equilibrium of those characteristics. We term Ω_x at the minimization of $\beta_0(\Omega_x)$ the *conspicuous subset*.

3.5.2 Multiple phase transitions.

Based on this characterization of Ω_x , we can hypothesize datasets with multiple learnability phase transitions. Specifically, consider a region Ω_{x0} that is small but “typical”, consists of all elements confidently predicted as y_0 by $p(y|x)$, and where y_0 is the least common class. By construction, this Ω_{x0} will dominate the infimum in Eq. (3.5), resulting in a small value of β_0 . However, the remaining $\mathcal{X} - \Omega_{x0}$ effectively form a new dataset, \mathcal{X}_1 . At exactly β_0 , we may have that the current encoder, $p_0(z|x)$, has no mutual information with the remaining classes in \mathcal{X}_1 ; i.e., $I(Y_1; Z_0) = 0$. In this case, Definition 1 applies to $p_0(z|x)$ with respect to $I(X_1; Z_1)$. We might expect to see that, at β_0 , learning will plateau until we get to some $\beta_1 > \beta_0$ that defines the phase transition for \mathcal{X}_1 . Clearly this process could repeat many times, with each new dataset \mathcal{X}_i being distinctly more difficult to learn than \mathcal{X}_{i-1} .

3.5.3 Similarity to information measures.

The denominator of $\beta_0(\Omega_x)$ in Eq. (3.5) is closely related to mutual information. Using the inequality $x - 1 \geq \log(x)$ for $x > 0$, it becomes:

$$\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right] \geq \mathbb{E}_{y \sim p(y|\Omega_x)} \left[\log \frac{p(y|\Omega_x)}{p(y)} \right] = \tilde{I}(\Omega_x; Y)$$

where $\tilde{I}(\Omega_x; Y)$ is the mutual information “density” at $\Omega_x \subset \mathcal{X}$. Of course, this quantity is also $\mathbb{D}_{\text{KL}}[p(y|\Omega_x) || p(y)]$, so we know that the denominator of Eq. (3.5) is non-negative. Incidentally, $\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]$ is the density of “rational mutual information” ([LT16a]) at Ω_x .

Similarly, the numerator of $\beta_0(\Omega_x)$ is related to the self-information of Ω_x :

$$\frac{1}{p(\Omega_x)} - 1 \geq \log \frac{1}{p(\Omega_x)} = -\log p(\Omega_x) = h(\Omega_x)$$

so we can estimate β_0 as:

$$\beta_0 \simeq \inf_{\Omega_x \subset \mathcal{X}} \frac{h(\Omega_x)}{\tilde{I}(\Omega_x; Y)} \quad (3.6)$$

Since Eq. (3.6) uses upper bounds on both the numerator and the denominator, it does not give us a bound on β_0 , only an estimate.

3.5.4 Estimating model capacity.

The observation that a model cannot distinguish between cluster overlap in the data and its own lack of capacity gives an interesting way to use IB-Learnability to measure the capacity of a set of models relative to the task they are being used to solve. For example, for a classification task, we can use different model classes to estimate $p(y|x)$. For each such trained model, we can estimate the corresponding IB-learnability threshold β_0 . A model with smaller capacity than the task needs will translate to more uncertainty in $p(y|\Omega_x)$, resulting in a larger β_0 . On the other hand, models that give the same β_0 as each other all have the same capacity relative to the task, even if we would otherwise expect them to have

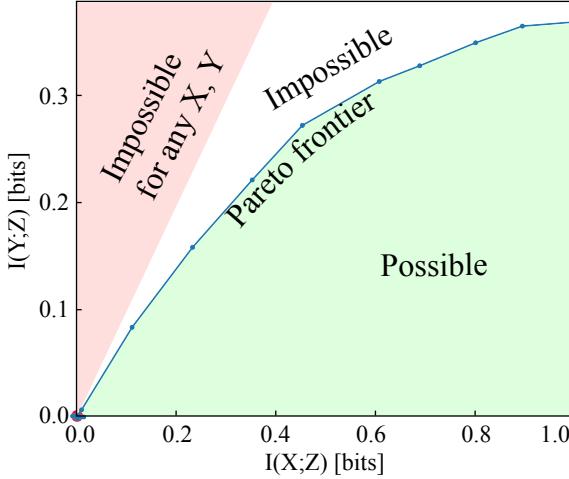


Figure 3-2: The Pareto frontier of the information plane, $I(X; Z)$ vs $I(Y; Z)$, for the binary classification of MNIST digits 0 and 1 with 20% label noise described in Sec. 3.1 and Fig. 3-1. For this problem, learning happens for models trained at $\beta > 3.25$. $H(Y) = 1$ bit since only two of ten digits are used, and $I(Y; Z) \leq I(X; Y) \approx 0.5$ bits $< H(Y)$ because of the 20% label noise. The true frontier is differentiable; the figure shows a variational approximation that places an upper bound on both informations, horizontally offset to pass through the origin.

very different capacities. For example, if two deep models have the same core architecture, but one has twice the number of parameters at each layer, and they both yield the same β_0 , their capacities are equivalent with respect to the task. Thus, β_0 provides a way to measure model capacity in a task-specific manner.

3.5.5 Learnability and the Information Plane.

Many of our results can be interpreted in terms of the geometry of the Pareto frontier illustrated in Fig. 3-2, which describes the trade-off between increasing $I(Y; Z)$ and decreasing $I(X; Z)$. At any point on this frontier that minimizes $\text{IB}_\beta^{\min} \equiv \min I(X; Z) - \beta I(Y; Z)$, the frontier will have slope β^{-1} if it is differentiable. If the frontier is also concave (has negative second derivative), then this slope β^{-1} will take its maximum β_0^{-1} at the origin, which implies IB_β -Learnability for $\beta > \beta_0$, so that the threshold for IB_β -Learnability is simply the inverse slope of the frontier at the origin. More generally, as long as the Pareto frontier is differentiable, the threshold for IB_β -learnability is the inverse of its maximum

slope. Indeed, Theorem 4 and Theorem 5 give lower bounds of the slope of the Pareto frontier at the origin.

3.5.6 IB-Learnability, hypercontractivity, and maximum correlation.

IB-Learnability and its sufficient conditions we provide harbor a deep connection with hypercontractivity and maximum correlation:

$$\frac{1}{\beta_0} = \xi(X; Y) = \eta_{\text{KL}} \geq \sup_{h(x)} \frac{1}{\beta_0[h(x)]} = \rho_m^2(X; Y) \quad (3.7)$$

which we prove in Appendix A.2.11. Here $\rho_m(X; Y) \equiv \max_{f,g} \mathbb{E}[f(X)g(Y)]$ s.t. $\mathbb{E}[f(X)] = \mathbb{E}[g(Y)] = 0$ and $\mathbb{E}[f^2(X)] = \mathbb{E}[g^2(Y)] = 1$ is the *maximum correlation* [Hir35, Geb41], $\xi(X; Y) \equiv \sup_{Z-X-Y} \frac{I(Y; Z)}{I(X; Z)}$ is the *hypercontractivity coefficient*, and $\eta_{\text{KL}}(p(y|x), p(x)) \equiv \sup_{r(x) \neq p(x)} \frac{\mathbb{D}_{\text{KL}}(r(y)||p(y))}{\mathbb{D}_{\text{KL}}(r(x)||p(x))}$ is the *contraction coefficient*. Our proof relies on [AGKN13]’s proof $\xi(X; Y) = \eta_{\text{KL}}$. Our work reveals the deep relationship between IB-Learnability and these earlier concepts and provides additional insights about what aspects of a dataset give rise to high maximum correlation and hypercontractivity: the most confident, typical, imbalanced subset of (X, Y) .

3.6 Estimating the IB-Learnability Condition

Theorem 5 not only reveals the relationship between the learnability threshold for β and the least noisy region of $P(Y|X)$, but also provides a way to practically estimate β_0 , both in the general classification case, and in more structured settings.

3.6.1 Estimation Algorithm

Based on Theorem 5, for general classification tasks we suggest Algorithm 1 to empirically estimate an upper-bound $\tilde{\beta}_0 \geq \beta_0$, as well as discovering the conspicuous subset that

determines β_0 .

We approximate the probability of each example $p(x_i)$ by its empirical probability, $\hat{p}(x_i)$. E.g., for MNIST, $p(x_i) = \frac{1}{N}$, where N is the number of examples in the dataset. The algorithm starts by first learning a maximum likelihood model of $p_\theta(y|x)$, using e.g. feed-forward neural networks. It then constructs a matrix $P_{y|x}$ and a vector p_y to store the estimated $p(y|x)$ and $p(y)$ for all the examples in the dataset. To find the subset Ω such that the $\tilde{\beta}_0$ is as small as possible, by previous analysis we want to find a *conspicuous* subset such that its $p(y|x)$ is large for a certain class j (to make the denominator of Eq. (3.5) large), and containing as many elements as possible (to make the numerator small).

We suggest the following heuristics to discover such a conspicuous subset. For each class j , we sort the rows of $(P_{y|x})$ according to its probability for the pivot class j by decreasing order, and then perform a search over $i_{\text{left}}, i_{\text{right}}$ for $\Omega = \{i_{\text{left}}, i_{\text{left}} + 1, \dots, i_{\text{right}}\}$. Since $\tilde{\beta}_0$ is large when Ω contains too few or too many elements, the minimum of $\tilde{\beta}_0^{(j)}$ for class j will typically be reached with some intermediate-sized subset, and we can use binary search or other discrete search algorithm for the optimization. The algorithm stops when $\tilde{\beta}_0^{(j)}$ does not improve by tolerance ε . The algorithm then returns the $\tilde{\beta}_0$ as the minimum over all the classes $\tilde{\beta}_0^{(1)}, \dots, \tilde{\beta}_0^{(N)}$, as well as the conspicuous subset that determines this $\tilde{\beta}_0$.

After estimating $\tilde{\beta}_0$, we can then use it for learning with IB, either directly, or as an anchor for a region where we can perform a much smaller sweep than we otherwise would have. This may be particularly important for very noisy datasets, where β_0 can be very large.

3.6.2 Special Cases for Estimating β_0

Theorem 5 may still be challenging to estimate, due to the difficulty of making accurate estimates of $p(\Omega_x)$ and searching over $\Omega_x \subset \mathcal{X}$. However, if the learning problem is more structured, we may be able to obtain a simpler formula for the sufficient condition.

Class-conditional label noise.

Classification with noisy labels is a common practical scenario. An important noise model is that the labels are randomly flipped with some hidden class-conditional probabilities and we only observe the corrupted labels. This problem has been studied extensively [AL88, NDRT13a, LT16b, XXY⁺15a, NWC17]. If IB is applied to this scenario, how large β do we need? The following corollary provides a simple formula.

Corollary 5.1. *Suppose that the true class labels are y^* , and the input space belonging to each y^* has no overlap. We only observe the corrupted labels y with class-conditional noise $p(y|x, y^*) = p(y|y^*)$, and Y is not independent of X . We have that a sufficient condition for IB_β -Learnability is:*

$$\beta > \inf_{y^*} \frac{\frac{1}{p(y^*)} - 1}{\sum_y \frac{p(y|y^*)^2}{p(y)} - 1} \quad (3.8)$$

We see that under class-conditional noise, the sufficient condition reduces to a discrete formula which only depends on the noise rates $p(y|y^*)$ and the true class probability $p(y^*)$, which can be accurately estimated via e.g. [NWC17]. Additionally, if we know that the noise is class-conditional, but the observed β_0 is greater than the R.H.S. of Eq. (3.8), we can deduce that there is overlap between the true classes. The proof of Corollary 5.1 is provided in Appendix A.2.10.

Deterministic relationships.

Theorem 5 also reveals that β_0 relates closely to whether Y is a deterministic function of X , as shown by Corollary 5.2:

Corollary 5.2. *Assume that Y contains at least one value y such that its probability $p(y) > 0$. If Y is a deterministic function of X and not independent of X , then a sufficient condition for IB_β -Learnability is $\beta > 1$.*

The assumption in the corollary 5.2 is satisfied by classification, and certain regression

problems.⁶ This corollary generalizes the result in [KTVK19] which only proves it for classification problems. Combined with the necessary condition $\beta > 1$ for any dataset (X, Y) to be IB_β -learnable (Section 3.3), we have that under the assumption, if Y is a deterministic function of X , then a necessary and sufficient condition for IB_β -learnability is $\beta > 1$; i.e., its β_0 is 1. The proof of Corollary 5.2 is provided in Appendix A.2.10.

Therefore, in practice, if we find that $\beta_0 > 1$, we may infer that Y is not a deterministic function of X . For a classification task, we may infer that either some classes have overlap, or the labels are noisy. However, recall that finite models may add effective class overlap if they have insufficient capacity for the learning task, as mentioned in Section 3.4. This may translate into a higher observed β_0 , even when learning deterministic functions.

3.7 Experiments

To test how the theoretical conditions for IB_β -learnability match with experiment, we apply them to synthetic data with varying noise rates and class overlap, MNIST binary classification with varying noise rates, and CIFAR10 classification, comparing with the β_0 found experimentally. We also compare with the algorithm in [KGK⁺17] for estimating the hypercontractivity coefficient ($=1/\beta_0$) via the contraction coefficient η_{KL} . Experiment details are in Section A.2.12.

3.7.1 Synthetic Dataset Experiments

We construct a set of datasets from 2D mixtures of 2 Gaussians as X and the identity of the mixture component as Y . We simulate two practical scenarios with these datasets: **(1)** noisy labels with class-conditional noise, and **(2)** class overlap. For (1), we vary the class-conditional noise rates. For (2), we vary class overlap by tuning the distance between the Gaussians. For each experiment, we sweep β with exponential steps, and observe $I(X; Z)$

⁶The following scenario does not satisfy this assumption: for certain regression problems where Y is a continuous random variable and the probability density function $p_Y(y)$ is bounded, then for any y , the probability $P(Y = y)$ has measure 0.

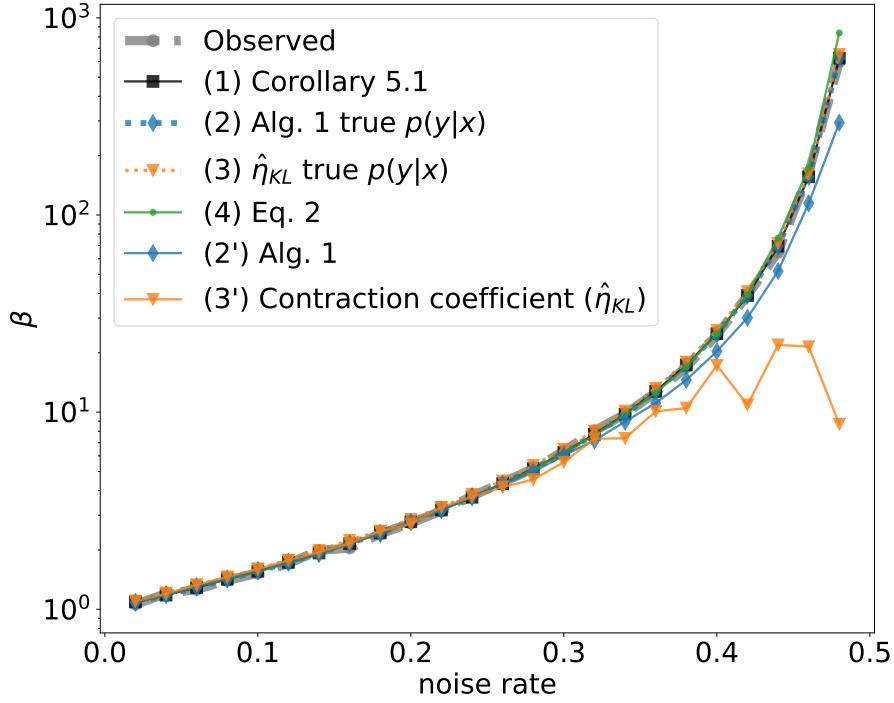


Figure 3-3: Predicted vs. experimentally identified β_0 , for mixture of Gaussians with varying class-conditional noise rates.

and $I(Y; Z)$. We then compare the empirical β_0 indicated by the onset of above-zero $I(X; Z)$ with predicted values for β_0 .

Classification with class-conditional noise. In this experiment, we have a mixture of Gaussian distribution with 2 components, each of which is a 2D Gaussian with diagonal covariance matrix $\Sigma = \text{diag}(0.25, 0.25)$. The two components have distance 16 (hence virtually no overlap) and equal mixture weight. For each x , the label $y \in \{0, 1\}$ is the identity of which component it belongs to. We create multiple datasets by randomly flipping the labels y with a certain noise rate $\rho = P(y = 0|y^* = 1) = P(y = 1|y^* = 0)$. For each dataset, we train VIB models across a range of β , and observe the onset of learning via random $I(X; Z)$ (Observed). To test how different methods perform in estimating β_0 , we apply the following methods: **(1)** Corollary 5.1, since this is classification with class-conditional noise, and the two true classes have virtually no overlap; **(2)** Alg. 1 with true $p(y|x)$; **(3)** The algorithm in [KGK⁺17] that estimates $\hat{\eta}_{KL}$, provided with true $p(y|x)$;

Table 3.1: Full table of values used to generate Fig. 3-3.

Noise rate	Observed	(2) Alg. 1		(3) $\hat{\eta}_{\text{KL}}$		(4) Eq. 3.2	(2') Alg. 1	(3') $\hat{\eta}_{\text{KL}}$
		(1) Corollary 5.1	true $p(y x)$	true $p(y x)$	(4) Eq. 3.2			
0.02	1.06	1.09	1.09	1.10	1.08	1.08	1.10	1.10
0.04	1.20	1.18	1.18	1.21	1.18	1.19	1.20	1.20
0.06	1.26	1.29	1.29	1.33	1.30	1.31	1.33	1.33
0.08	1.40	1.42	1.42	1.45	1.42	1.43	1.46	1.46
0.10	1.52	1.56	1.56	1.60	1.55	1.58	1.60	1.60
0.12	1.70	1.73	1.73	1.78	1.71	1.73	1.77	1.77
0.14	1.99	1.93	1.93	1.99	1.90	1.91	1.95	1.95
0.16	2.04	2.16	2.16	2.24	2.15	2.15	2.16	2.16
0.18	2.41	2.44	2.44	2.49	2.43	2.42	2.49	2.49
0.20	2.74	2.78	2.78	2.86	2.76	2.77	2.71	2.71
0.22	3.15	3.19	3.19	3.29	3.19	3.21	3.29	3.29
0.24	3.75	3.70	3.70	3.83	3.71	3.75	3.72	3.72
0.26	4.40	4.34	4.34	4.48	4.35	4.31	4.17	4.17
0.28	5.16	5.17	5.17	5.37	5.12	4.98	4.55	4.55
0.30	6.34	6.25	6.25	6.49	6.24	6.03	5.58	5.58
0.32	8.06	7.72	7.72	8.02	7.63	7.19	7.33	7.33
0.34	9.77	9.77	9.77	10.13	9.74	8.95	7.37	7.37
0.36	12.58	12.76	12.76	13.21	12.51	11.11	10.09	10.09
0.38	16.91	17.36	17.36	17.96	16.97	14.55	10.49	10.49
0.40	24.66	25.00	25.00	25.99	25.01	20.36	17.27	17.27
0.42	39.08	39.06	39.06	40.85	39.48	30.12	10.89	10.89
0.44	64.82	69.44	69.44	71.80	76.48	51.95	21.95	21.95
0.46	163.07	156.25	156.26	161.88	173.15	114.57	21.47	21.47
0.48	599.45	625.00	625.00	651.47	838.90	293.90	8.69	8.69

(4) $\beta_0[h(x)]$ in Eq. (3.2); (2') Alg. 1 with $p(y|x)$ estimated by a neural net; (3') $\hat{\eta}_{\text{KL}}$ with the same $p(y|x)$ as in (2'). The results are shown in Fig. 3-3 and in Table 3.1.

From Fig. 3-3 and Table 3.1 we see the following. **(A)** When using the true $p(y|x)$, both Alg. 1 and $\hat{\eta}_{\text{KL}}$ generally upper bound the empirical β_0 , and Alg. 1 is generally tighter. **(B)** When using the true $p(y|x)$, Alg. 1 and Corollary 5.1 give the same result. **(C)** Comparing Alg. 1 and $\hat{\eta}_{\text{KL}}$ both of which use the same empirically estimated $p(y|x)$, both approaches provide good estimation in the low-noise region; however, in the high-noise region, Alg. 1 gives more precise values than $\hat{\eta}_{\text{KL}}$, indicating that Alg. 1 is more robust to the estimation error of $p(y|x)$. **(D)** Eq. (3.2) empirically upper bounds the experimentally observed β_0 , and

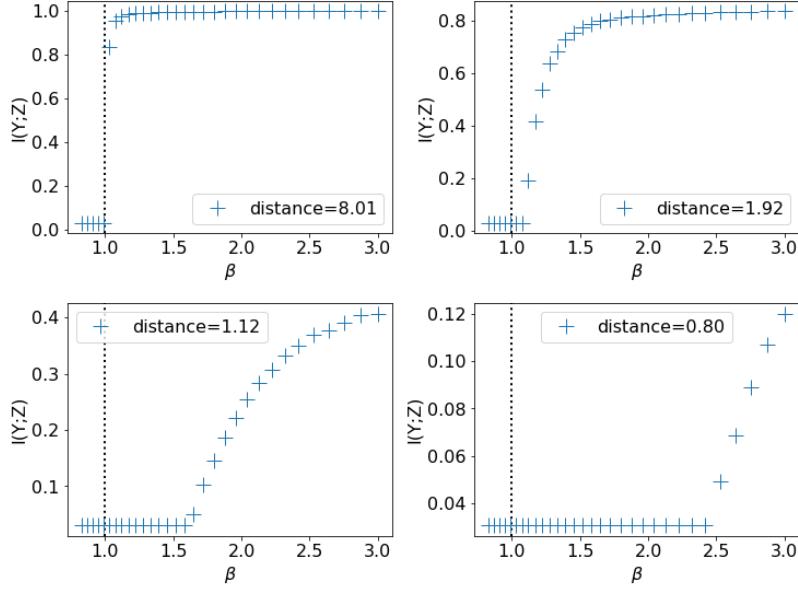


Figure 3-4: $I(Y; Z)$ vs. β , for mixture of Gaussian datasets with different distances between the two mixture components. The vertical lines are $\beta_{0,\text{predicted}}$ computed by the R.H.S. of Eq. (3.8). As Eq. (3.8) does not make predictions w.r.t. class overlap, the vertical lines are always just above $\beta_{0,\text{predicted}} = 1$. However, as expected, decreasing the distance between the classes in X space also increases the true β_0 .

gives almost the same result as theoretical estimation in Corollary 5.1 and Alg. 1 with the true $p(y|x)$. In the classification setting, this approach doesn't require any learned estimate of $p(y|x)$, as we can directly use the empirical $p(y)$ and $p(x|y)$ from SGD mini-batches.

This experiment also shows that for dataset where the signal-to-noise is small, β_0 can be very high. Instead of blindly sweeping β , our result can provide guidance for setting β so learning can happen.

Classification with class overlap. In this experiment, we test how different amounts of overlap among classes influence β_0 . We use the mixture of Gaussians with two components, each of which is a 2D Gaussian with diagonal covariance matrix $\Sigma = \text{diag}(0.25, 0.25)$. The two components have weights 0.6 and 0.4. We vary the distance between the Gaussians from 8.0 down to 0.8 and observe the $\beta_{0,\text{exp}}$. Since we don't add noise to the labels, if there were no overlap and a deterministic map from X to Y , we would have $\beta_0 = 1$ by Corollary 5.2. The more overlap between the two classes, the more uncertain Y is given X . By Eq. 3.5

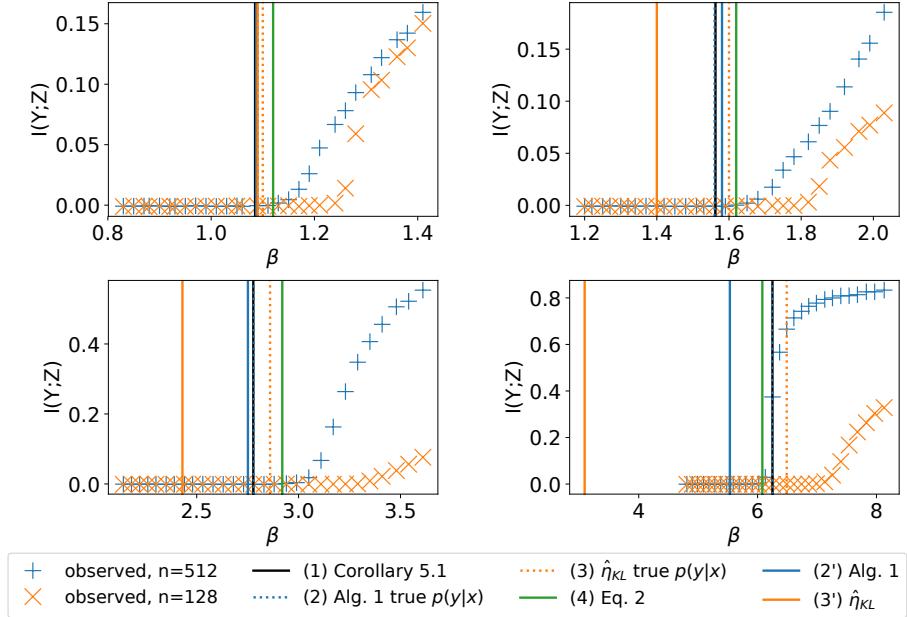


Figure 3-5: $I(Y;Z)$ vs. β for the MNIST binary classification with different hidden units per layer n and noise rates ρ : (upper left) $\rho = 0.02$, (upper right) $\rho = 0.1$, (lower left) $\rho = 0.2$, (lower right) $\rho = 0.3$. The vertical lines are β_0 estimated by different methods. $n = 128$ has insufficient capacity for the problem, so its observed learnability onset is pushed higher, similar to the class overlap case.

we expect β_0 to be larger, which is corroborated in Fig. 3-4.

3.7.2 MNIST Experiments

We perform binary classification with digits 0 and 1, and as before, add class-conditional noise to the labels with varying noise rates ρ . To explore how the model capacity influences the onset of learning, for each dataset we train two sets of VIB models differing only by the number of neurons in their hidden layers of the encoder: one with $n = 512$ neurons, the other with $n = 128$ neurons. As we describe in Section 3.4, insufficient capacity will result in more uncertainty of Y given X from the point of view of the model, so we expect the observed β_0 for the $n = 128$ model to be larger. This result is confirmed by the experiment (Fig. 3-5). Also, in Fig. 3-5 we plot β_0 given by different estimation methods. We see that the observations (A), (B), (C) and (D) in Section 3.7.1 still hold.

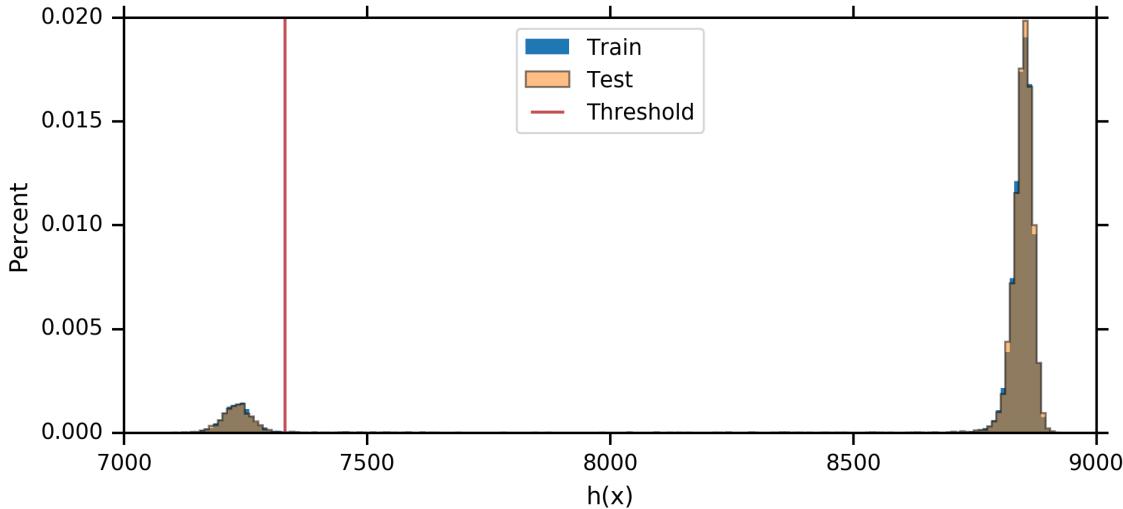


Figure 3-6: Histograms of the full MNIST training and validation sets according to $h(X)$. Note that both are bimodal, and the histograms are indistinguishable. In both cases, $h(x)$ has learned to separate most of the ones into the smaller mode, but difficult ones are in the wide valley between the two modes. See Figure 3-8 for all of the training images to the left of the red threshold line, as well as the first few images to the right of the threshold.

3.7.3 MNIST Experiments using Equation 3.2

To see what IB learns at its onset of learning for the full MNIST dataset, we optimize Eq. (3.2) w.r.t. the full MNIST dataset, and visualize the clustering of digits by $h(x)$. Eq. (3.2) can be optimized using SGD using any differentiable parameterized mapping $h(x) : \mathcal{X} \rightarrow \mathbb{R}$. In this case, we chose to parameterize $h(x)$ with a PixelCNN++ architecture [vdOKE⁺16, SKCK17], as PixelCNN++ is a powerful autoregressive model for images that gives a scalar output (normally interpreted as $\log p(x)$). Eq. (3.2) should generally give two clusters in the output space, as discussed in Section 3.4. In this setup, smaller values of $h(x)$ correspond to the subset of the data that is easiest to learn. Fig. 3-6 shows two strongly separated clusters, as well as the threshold we choose to divide them. Fig. 3-8 shows the first 5,776 MNIST training examples as sorted by our learned $h(x)$, with the examples above the threshold highlighted in red. We can clearly see that our learned $h(x)$ has separated the “easy” one (1) digits from the rest of the MNIST training set.

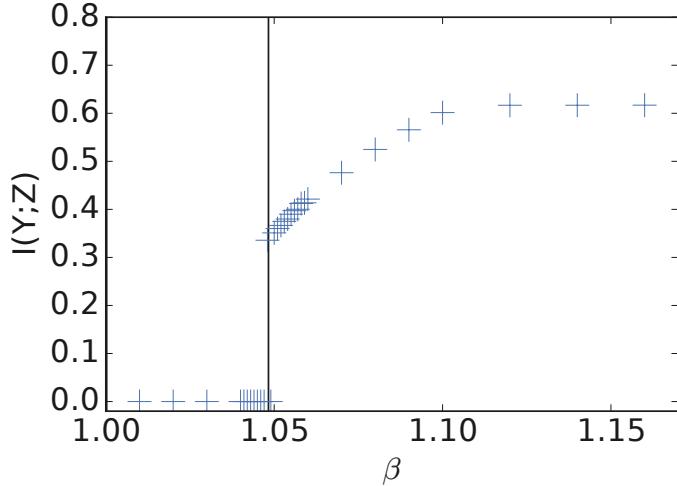


Figure 3-7: Plot of $I(Y; Z)$ vs β for CIFAR10 training set with 20% label noise. Each blue cross corresponds to a fully-converged model starting with independent initialization. The vertical black line corresponds to the predicted $\beta_0 = 1.0483$ using Alg. 1. The empirical $\beta_0 = 1.048$.

3.7.4 CIFAR10 Forgetting Experiments

For CIFAR10 [KH09], we study how *forgetting* varies with β . In other words, given a VIB model trained at some high β_2 , if we anneal it down to some much lower β_1 , what $I(Y; Z)$ does the model converge to? Using Alg. 1, we estimated $\beta_0 = 1.0483$ on a version of CIFAR10 with 20% label noise, where the $P_{y|x}$ is estimated by maximum likelihood training with the same encoder and classifier architectures as used for VIB. For the VIB models, the lowest β with performance above chance was $\beta = 1.048$, a very tight match with the estimate from Alg. 1. See Appendix A.2.12 for details.

3.8 Conclusion

In this paper, we have presented theoretical results for predicting the onset of learning, and have shown that it is determined by the conspicuous subset of the training examples. We gave a practical algorithm for predicting the transition as well as discovering this subset, and showed that those predictions are accurate, even in cases of extreme label noise. We proved a deep connection between IB-learnability, our upper bounds on β_0 , the hypercontractivity

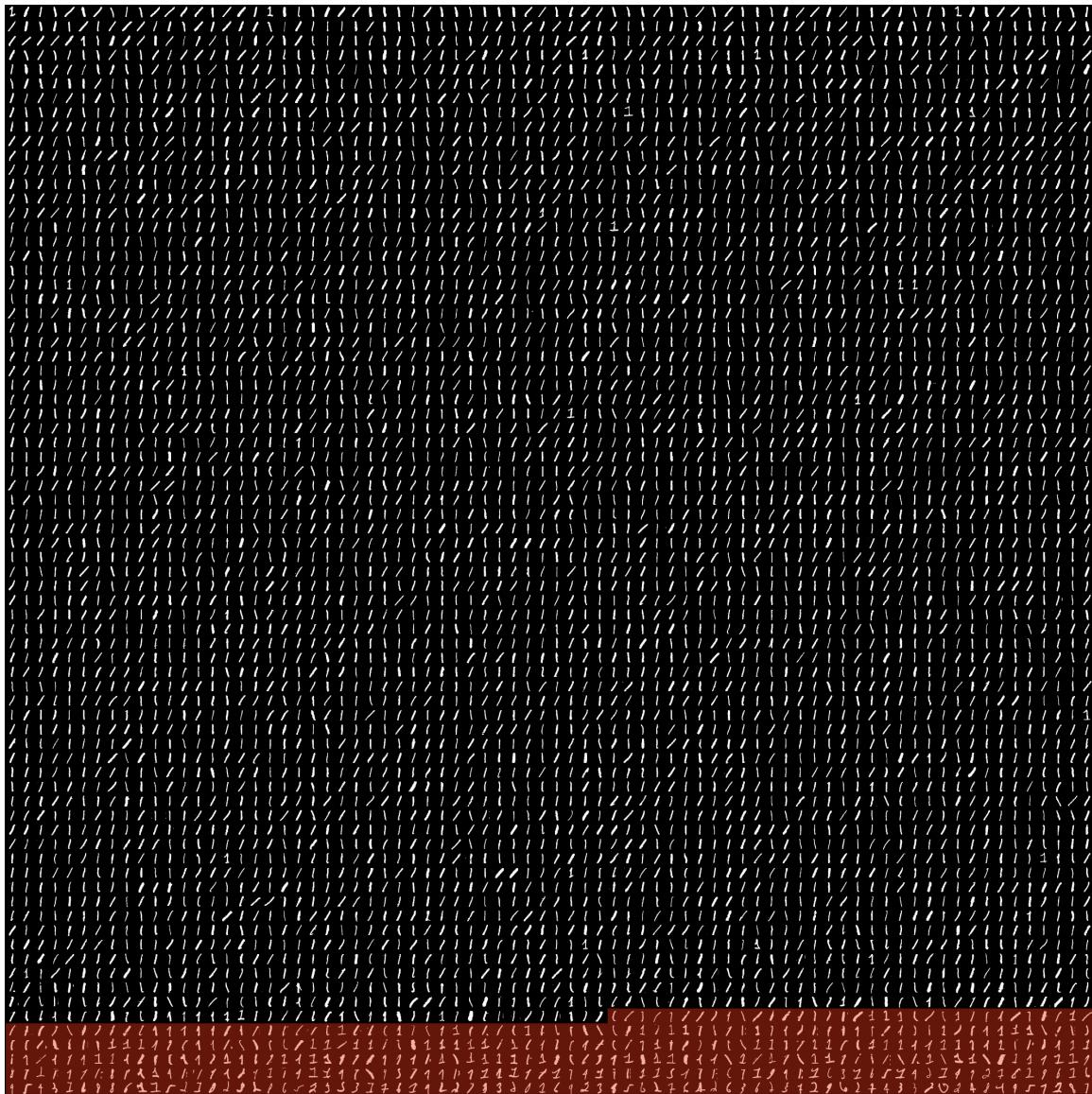


Figure 3-8: The first 5776 MNIST training set digits when sorted by $h(x)$. The digits highlighted in red are above the threshold drawn in Figure 3-6.

coefficient, the contraction coefficient, and the maximum correlation. We believe that these results provide a deeper understanding of IB, as well as a tool for analyzing a dataset by discovering its conspicuous subset, and a tool for measuring model capacity in a task-specific manner.

Our work also raises other questions, such as whether there are other phase transitions in learnability that might be identified. We hope to address some of those questions in future work.

Algorithm 1 Estimating the upper bound for β_0 and identifying the conspicuous subset.

Require: Dataset $\mathbf{D} = \{(x_i, y_i)\}, i = 1, 2, \dots, N$. The number of classes is C .

Require ε : tolerance for estimating β_0

1: Learn a maximum likelihood model $p_\theta(y|x)$ using the dataset \mathbf{D} .

2: Construct matrix $(P_{y|x})$ such that $(P_{y|x})_{ij} = p_\theta(y = y_j|x = x_i)$.

3: Construct vector $p_y = (p_{y1}, \dots, p_{yC})$ such that $p_{yj} = \frac{1}{N} \sum_{i=1}^N (P_{y|x})_{ij}$.

4: **for** j **in** $\{1, 2, \dots, C\}$:

5: $P_{y|x}^{(\text{sort}j)} \leftarrow$ Sort the rows of $P_{y|x}$ in decreasing values of $(P_{y|x})_{ij}$.

6: $\tilde{\beta}_0^{(j)}, \Omega^{(j)} \leftarrow$ Search $i_{\text{left}}, i_{\text{right}}$ until $\tilde{\beta}_0^{(j)} = \text{Get}\beta(P_{y|x}, p_y, \Omega)$ is minimal with tolerance ε ,

where $\Omega = \{i_{\text{left}}, i_{\text{left}} + 1, \dots, i_{\text{right}}\}$.

7: **end for**

8: $j^* \leftarrow \arg \min_j \{\tilde{\beta}_0^{(j)}\}, j = 1, 2, \dots, N$.

9: $\tilde{\beta}_0 \leftarrow \tilde{\beta}_0^{(j^*)}$.

10: $P_{y|x}^{(\tilde{\beta}_0)} \leftarrow$ the rows of $P_{y|x}^{(\text{sort}j^*)}$ indexed by $\Omega^{(j^*)}$.

11: **return** $\tilde{\beta}_0, P_{y|x}^{(\tilde{\beta}_0)}$

subroutine $\text{Get}\beta(P_{y|x}, p_y, \Omega)$:

s1: $N \leftarrow$ number of rows of $P_{y|x}$.

s2: $C \leftarrow$ number of columns of $P_{y|x}$.

s3: $n \leftarrow$ number of elements of Ω .

s4: $(p_{y|\Omega})_j \leftarrow \frac{1}{n} \sum_{i \in \Omega} (P_{y|x})_{ij}, j = 1, 2, \dots, C$.

s5: $\tilde{\beta}_0 \leftarrow \frac{\frac{N}{n} - 1}{\sum_j \left[\frac{(p_{y|\Omega})_j^2}{p_{yj}} - 1 \right]}$

s6: **return** $\tilde{\beta}_0$

Chapter 4

Intermediate phase transitions

In Chapter 3, we have studied the first phase transition in Information Bottleneck (IB), the learnability transition, in detail. In general, when tuning the relative strength between compression and prediction terms in IB, how do the two terms behave, and what's their relationship with the dataset and the learned representation? In this chapter¹, we set out to answer this question by studying multiple phase transitions in the IB objective: $\text{IB}_\beta[p(z|x)] = I(X; Z) - \beta I(Y; Z)$, where sudden jumps of $\frac{dI(Y;Z)}{d\beta}$ and prediction accuracy are observed with increasing β . We introduce a definition for IB phase transitions as a qualitative change of the IB loss landscape, and show that the transitions correspond to the onset of learning new classes. Using second-order calculus of variations, we derive a formula that provides a practical condition for IB phase transitions, and draw its connection with the Fisher information matrix for parameterized models. We provide two perspectives to understand the formula, revealing that each IB phase transition is finding a component of maximum (nonlinear) correlation between X and Y orthogonal to the learned representation, in close analogy with canonical-correlation analysis (CCA) in linear settings. Based on the theory, we present an algorithm for discovering phase transition points. Finally, we verify that our theory and algorithm accurately predict phase transitions in categorical datasets, predict the onset of learning new classes and class difficulty in MNIST, and predict

¹The paper “Phase transitions for the information bottleneck in representation learning” is published as a conference paper at *ICLR* 2020. A short version is presented at *NeurIPS* 2019 Workshop on Information Theory and Machine Learning. Authors: Wu, Tailin and Ian Fischer. arXiv:2001.01878.

prominent phase transitions in CIFAR10 experiments.

4.1 Introduction

The Information Bottleneck (IB) objective [TPB00]:

$$\text{IB}_\beta[p(z|x)] := I(X; Z) - \beta I(Y; Z) \quad (4.1)$$

explicitly trades off model compression ($I(X; Z)$, $I(\cdot; \cdot)$ denoting mutual information) with predictive performance ($I(Y; Z)$) using the Lagrange multiplier β , where X, Y are observed random variables, and Z is a learned representation of X . The IB method has proved effective in a variety of scenarios, including improving the robustness against adversarial attacks [AFDM16, Fis18], learning invariant and disentangled representations [AS18a, AS18b], underlying information-based geometric clustering [SS17b], improving the training and performance in adversarial learning [PKT⁺18], and facilitating skill discovery [SGL⁺19] and learning goal-conditioned policy [GIS⁺19] in reinforcement learning.

From Eq. (4.1) we see that when $\beta \rightarrow 0$ it will encourage $I(X; Z) = 0$ which leads to a trivial representation Z that is independent of X , while when $\beta \rightarrow +\infty$, it reduces to a maximum likelihood objective² that does not constrain the information flow. Between these two extremes, how will the IB objective behave? Will prediction and compression performance change smoothly, or do there exist interesting transitions in between? In [WFCT19a], the authors observe and study the learnability transition, i.e. the β value such that the IB objective transitions from a trivial global minimum to learning a nontrivial representation. They also show how this first phase transition relates to the structure of the dataset. However, to answer the full question, we need to consider the full range of β .

Motivation. To get a sense of how $I(Y; Z)$ and $I(X; Z)$ vary with β , we train Variational Information Bottleneck (VIB) models [AFDM16] on the CIFAR10 dataset [KH09], where each experiment is at a different β and random initialization of the model. Fig. 4-1 shows

²For example, in classification, it reduces to cross-entropy loss.

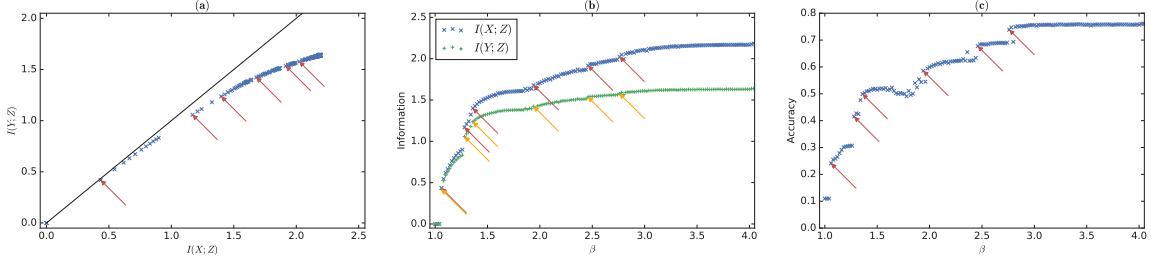


Figure 4-1: CIFAR10 plots (a) showing the information plane, as well as β vs (b) $I(X; Z)$ and $I(Y; Z)$, and (c) accuracy, all on the training set with 20% label noise. The arrows point to empirically-observed phase transitions. The vertical lines correspond to phase transitions found with Alg. 2.

the $I(X; Z)$, $I(Y; Z)$ and accuracy vs. β , as well as $I(Y; Z)$ vs. $I(X; Z)$ for CIFAR10 with 20% label noise (see Appendix A.2.12 for details).

From Fig. 4-1(b)(c), we see that as we increase β , instead of going up smoothly, both $I(X; Z)$ and $I(Y; Z)$ show multiple phase transitions, where the slopes $\frac{dI(X;Z)}{d\beta}$ and $\frac{dI(Y;Z)}{d\beta}$ are discontinuous and the accuracy has discrete jumps. The observation lets us refine our question: When do the phase transitions occur, and how do they depend on the structure of the dataset? These questions are important, since answering them will help us gain a better understanding of the IB objective and its close interplay with the dataset and the learned representation.

Moreover, the IB objective belongs to a general form of two-term trade-offs in many machine learning objectives: $L = \text{Prediction-loss} + \beta \cdot \text{Complexity}$, where the complexity term generally takes the form of regularization. Usually, learning is set at a specific β . Many more insights can be gained if we understand the behavior of the prediction loss and model complexity with varying β , and how they depend on the dataset. The techniques developed to address the question in the IB setting may also help us understand the two-term tradeoff in other learning objectives.

Contributions. In this work, we begin to address the above question in IB settings. Specifically:

- We identify a *qualitative* change of the IB loss landscape w.r.t. $p(z|x)$ for varying β

as IB phase transitions (Section 4.3).

- Based on the definition, we introduce a quantity $G[p(z|x)]$ and use it to prove a theorem giving a practical condition for IB phase transitions. We further reveal the connection between $G[p(z|x)]$ and the Fisher information matrix when $p(z|x)$ is parameterized by θ (Section 4.3).
- We reveal the close interplay between the IB objective, the dataset and the learned representation, by showing that in IB, each phase transition corresponds to learning a new nonlinear component of maximum correlation between X and Y , orthogonal to the previously-learned Z , and each with decreasing strength (Section 4.4).

To the best of our knowledge, our work provides the first theoretical formula to address IB phase transitions in the most general setting. In addition, we present an algorithm for iteratively finding the IB phase transition points (Section 4.5). We show that our theory and algorithm give tight matches with the observed phase transitions in categorical datasets, predict the onset of learning new classes and class difficulty in MNIST, and predict prominent transitions in CIFAR10 experiments (Section 4.6).

4.2 Related Work

The Information Bottleneck Method [TPB00] provides a tabular method based on the Blahut-Arimoto (BA) Algorithm [Bla72] to numerically solve the IB functional for the optimal encoder distribution $P(Z|X)$, given the trade-off parameter β and the cardinality of the representation variable Z . This work has been extended in a variety of directions, including to the case where all three variables X, Y, Z are multivariate Gaussians [CGTW05], cases of variational bounds on the IB and related functionals for amortized learning [AFDM16, AS18a, Fis18], and a more generalized interpretation of the constraint on model complexity as a Kolmogorov Structure Function [AMS18]. Previous theoretical analyses of IB include [RR12], which looks at IB through the lens of copula functions, and [SST10], which starts to tackle the question of how to bound generalization with IB.

We will make practical use of the original IB algorithm, as well as the amortized bounds of the Variational Information Bottleneck [AFDM16] and the Conditional Entropy Bottleneck [Fis18].

Phase transitions, where key quantities change discontinuously with varying relative strength in the two-term trade-off, have been observed in many different learning domains, for multiple learning objectives. In [RV18], the authors observe phase transitions in the latent representation of β -VAE for varying β . [SS17b] utilize the kink angle of the phase transitions in the Deterministic Information Bottleneck (DIB) [SS17a] to determine the optimal number of clusters for geometric clustering. [TW19] explicitly considers critical points in binary classification tasks using a discrete information bottleneck with a non-convex Pareto-optimal frontier. In [AS18a], the authors observe a transition on the tradeoff of $I(\theta; X, Y)$ vs. $H(Y|X, \theta)$ in InfoDropout. Under IB settings, [CGTW05] study the Gaussian Information Bottleneck, and analytically solve the critical values $\beta_i^c = \frac{1}{1-\lambda_i}$, where λ_i are eigenvalues of the matrix $\Sigma_{x|y}\Sigma_x^{-1}$, and Σ_x is the covariance matrix. This work provides valuable insights for IB, but is limited to the special case that X , Y and Z are jointly Gaussian. Phase transitions in the general IB setting have also been observed, which [Tis18] describes as “information bifurcation”. In [WFCT19a], the authors study the first phase transition, i.e. the learnability phase transition, and provide insights on how the learnability depends on the dataset. Our work is the first work that addresses all the IB phase transitions in the most general setting, and provides theoretical insights on the interplay between the IB objective, its phase transitions, the dataset, and the learned representation.

4.3 Formula for IB phase transitions

4.3.1 Definitions

Let $X \in \mathcal{X}, Y \in \mathcal{Y}, Z \in \mathcal{Z}$ be random variables denoting the input, target and representation, respectively, having a joint probability distribution $p(X, Y, Z)$, with $\mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$ its support. X , Y and Z satisfy the Markov chain $Z - X - Y$, i.e. Y and Z are conditionally independent given X . We assume that the integral (or summing if X , Y or Z are discrete

random variables) is on $\mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$. We use x , y and z to denote the instances of the respective random variables. The above settings are used throughout the paper. We can view the IB objective $\text{IB}_\beta[p(z|x)]$ (Eq. 4.1) as a functional of the encoding distribution $p(z|x)$. To prepare for the introduction of IB phase transitions, we first define *relative perturbation function* and *second variation*, as follows.

Def. 2. Relative perturbation function: For $p(z|x)$, its relative perturbation function $r(z|x)$ is a bounded function that maps $\mathcal{X} \times \mathcal{Z}$ to \mathbb{R} and satisfies $\mathbb{E}_{z \sim p(z|x)}[r(z|x)] = 0$. Formally, define $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}} := \{r(z|x) : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbb{R} \mid \mathbb{E}_{z \sim p(z|x)}[r(z|x)] = 0, \text{and } \exists M > 0 \text{ s.t. } \forall X \in \mathcal{X}, Z \in \mathcal{Z}, |r(z|x)| \leq M\}$. We have that $r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ iff $r(z|x)$ is a relative perturbation function of $p(z|x)$. The perturbed probability (density) is $p'(z|x) = p(z|x)(1 + \epsilon \cdot r(z|x))$ for some $\epsilon > 0$.

Def. 3. Second variation: Let functional $F[f(x)]$ be defined on some normed linear space \mathcal{R} . Let us add a perturbative function $\epsilon \cdot h(x)$ to $f(x)$, and now the functional $F[f(x) + \epsilon \cdot h(x)]$ can be expanded as

$$\begin{aligned}\Delta F[f(x)] &= F[f(x) + \epsilon \cdot h(x)] - F[f(x)] \\ &= \varphi_1[\epsilon \cdot h(x)] + \varphi_2[\epsilon \cdot h(x)] + \varphi_r[\epsilon \cdot h(x)] \|\epsilon \cdot h(x)\|^2\end{aligned}$$

such that $\lim_{\epsilon \rightarrow 0} \varphi_r[\epsilon \cdot h(x)] = 0$, where $\|\cdot\|$ denotes the norm, $\varphi_1[\epsilon \cdot h(x)] = \epsilon \frac{dF[f(x)]}{d\epsilon}$ is a linear functional of $\epsilon \cdot h(x)$, and is called the first variation, denoted as $\delta F[f(x)]$. $\varphi_2[\epsilon \cdot h(x)] = \frac{1}{2} \epsilon^2 \frac{d^2 F[f(x)]}{d\epsilon^2}$ is a quadratic functional of $\epsilon \cdot h(x)$, and is called the second variation, denoted as $\delta^2 F[f(x)]$.

We can think of the perturbation function $\epsilon \cdot h(x)$ as an infinite-dimensional “vector” (x being the indices), with ϵ being its amplitude and $h(x)$ its direction. With the above preparations, we define the IB phase transition as a change in the local curvature on the global minimum of $\text{IB}_\beta[p(z|x)]$.

Def. 4. IB phase transitions: Let $r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ be a perturbation function of $p(z|x)$, $p_\beta^*(z|x)$ denote the optimal solution of $\text{IB}_\beta[p(z|x)]$ at β , where the IB functional $\text{IB}[\cdot]$ is

defined in Eq. (4.1). The IB phase transitions β_i^c are the β values satisfying the following two conditions:

$$(1) \forall r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}, \delta^2 \text{IB}_\beta[p(z|x)] \Big|_{p_\beta^*(z|x)} \geq 0;$$

$$(2) \lim_{\beta' \rightarrow \beta^+} \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} \delta^2 \text{IB}_{\beta'}[p(z|x)] \Big|_{p_\beta^*(z|x)} = 0^-.$$

Here β^+ and 0^- denote one-sided limits.

We can understand the $\delta^2 \text{IB}_\beta[p(z|x)]$ as a local ‘‘curvature’’ of the IB objective IB_β (Eq. 4.1) w.r.t. $p(z|x)$, along some relative perturbation $r(z|x)$. A phase transition occurs when the convexity of $\text{IB}_\beta[p(z|x)]$ w.r.t. $p(z|x)$ changes from a minimum to a saddle point in the neighborhood of its optimal solution $p_\beta^*(z|x)$ as β increases from β_c to $\beta_c + 0^+$. This means that there exists a perturbation to go downhill and find a better minimum. We validate this definition empirically below.

4.3.2 Condition for IB phase transitions

The definition for IB phase transition (Definition 4) indicates the important role $\delta^2 \text{IB}_\beta[p(z|x)]$ plays on the optimal solution in providing the condition for phase transitions. To concretize it and prepare for a more practical condition for IB phase transitions, we expand $\text{IB}_\beta[p(z|x)(1 + \epsilon \cdot r(z|x))]$ to the second order of ϵ , giving:

Lemma 5.1. *For $\text{IB}_\beta[p(z|x)]$, the condition of $\forall r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}, \delta^2 \text{IB}_\beta[p(z|x)] \geq 0$ is equivalent to $\beta \leq G[p(z|x)]$. The threshold function $G[p(z|x)]$ is given by:*

$$G[p(z|x)] := \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} \mathcal{G}[r(z|x); p(z|x)]$$

$$\mathcal{G}[r(z|x); p(z|x)] := \frac{\mathbb{E}_{x,z \sim p(x,z)} [r^2(z|x)] - \mathbb{E}_{z \sim p(z)} \left[(\mathbb{E}_{x \sim p(x|z)} [r(z|x)])^2 \right]}{\mathbb{E}_{y,z \sim p(y,z)} \left[(\mathbb{E}_{x \sim p(x|y,z)} [r(z|x)])^2 \right] - \mathbb{E}_{z \sim p(z)} \left[(\mathbb{E}_{x \sim p(x|z)} [r(z|x)])^2 \right]} \quad (4.2)$$

The proof is given in Appendix A.3.2, in which we also give Eq. (A.39) for empirical estimation. Note that Lemma 5.1 is very general and can be applied to any $p(z|x)$, not only at the optimal solution $p_\beta^*(z|x)$.

The Fisher Information matrix. In practice, the encoder $p_{\theta}(z|x)$ is usually parameterized by some parameter vector $\theta = (\theta_1, \theta_2, \dots, \theta_k)^T \in \Theta$, e.g. weights and biases in a neural net, where Θ is the parameter field. An infinitesimal change of $\theta' \leftarrow \theta + \Delta\theta$ induces a relative perturbation $\epsilon \cdot r(z|x) \simeq \Delta\theta^T \frac{\partial \log p_{\theta}(z|x)}{\partial \theta}$ on $p_{\theta}(z|x)$, from which we can compute the threshold function $G_{\Theta}[p_{\theta}(z|x)]$:

Lemma 5.2. *For $\text{IB}_{\beta}[p_{\theta}(z|x)]$ objective, the condition of $\forall \Delta\theta \in \Theta, \delta^2 \text{IB}_{\beta}[p_{\theta}(z|x)] \geq 0$ is equivalent to $\beta \leq G_{\Theta}[p_{\theta}(z|x)]$, where*

$$G_{\Theta}[p_{\theta}(z|x)] := \inf_{\Delta\theta \in \Theta} \frac{\Delta\theta^T (\mathcal{I}_{Z|X}(\theta) - \mathcal{I}_Z(\theta)) \Delta\theta}{\Delta\theta^T (\mathcal{I}_{Z|Y}(\theta) - \mathcal{I}_Z(\theta)) \Delta\theta} = \lambda_{\max}^{-1} \quad (4.3)$$

where $\mathcal{I}_Z(\theta) := \int dz p_{\theta}(z) \left(\frac{\partial \log p_{\theta}(z)}{\partial \theta} \right) \left(\frac{\partial \log p_{\theta}(z)}{\partial \theta} \right)^T$ is the Fisher information matrix of θ for Z , $\mathcal{I}_{Z|X}(\theta) := \int dx dz p(x)p_{\theta}(z|x) \left(\frac{\partial \log p_{\theta}(z|x)}{\partial \theta} \right) \left(\frac{\partial \log p_{\theta}(z|x)}{\partial \theta} \right)^T$, $\mathcal{I}_{Z|Y}(\theta) := \int dy dz p(y)p_{\theta}(z|y) \left(\frac{\partial \log p_{\theta}(z|y)}{\partial \theta} \right) \left(\frac{\partial \log p_{\theta}(z|y)}{\partial \theta} \right)^T$ are the conditional Fisher information matrix [Zeg15] of θ for Z conditioned on X and Y , respectively. λ_{\max} is the largest eigenvalue of $C^{-1} (\mathcal{I}_{Z|Y}(\theta) - \mathcal{I}_Z(\theta)) (C^T)^{-1}$ with v_{\max} the corresponding eigenvector, where CC^T is the Cholesky decomposition of the matrix $\mathcal{I}_{Z|X}(\theta) - \mathcal{I}_Z(\theta)$, and v_{\max} is the eigenvector for λ_{\max} . The infimum is attained at $\Delta\theta = (C^T)^{-1} v_{\max}$.

The proof is in appendix A.3.3. We see that for parameterized encoders $p_{\theta}(z|x)$, each term of $G[p(z|x)]$ in Eq. (4.2) can be replaced by a bilinear form with the Fisher information matrix of the respective variables. Although this lemma is not required to understand the more general setting of Lemma 5.1, where the model is described in a functional space, Lemma 5.2 helps understand $G[p(z|x)]$ for parameterized models, which permits directly linking the phase transitions to the model's parameters.

Phase Transitions. Now we introduce Theorem 6 that gives a concrete and practical condition for IB phase transitions, which is the core result of the paper:

Theorem 6. *The IB phase transition points $\{\beta_i^c\}$ as defined in Definition 4 are given by the roots of the following equation:*

$$G[p_{\beta}^*(z|x)] = \beta \quad (4.4)$$

where $G[p(z|x)]$ is given by Eq. (4.2) and $p_\beta^*(z|x)$ is the optimal solution of $\text{IB}_\beta[p(z|x)]$ at β .

We can understand Eq. (4.4) as the condition when $\delta^2 \text{IB}_\beta[p(z|x)]$ is *about* to be able to be negative at the optimal solution $p_\beta^*(z|x)$ for a given β . The proof for Theorem 6 is given in Appendix A.3.4. In Section 4.4, we will analyze Theorem 6 in detail.

4.4 Understanding the formula for IB phase transitions

In this section we set out to understand $G[p(z|x)]$ as given by Eq. (4.2) and the phase transition condition as given by Theorem 6, from the perspectives of Jensen's inequality and representational maximum correlation.

4.4.1 Jensen's Inequality

The condition for IB phase transitions given by Theorem 6 involves $G[p(z|x)] = \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} \mathcal{G}[r(z|x); p(z|x)]$ which is in itself an optimization problem. We can understand $G[p(z|x)] = \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} \frac{A-C}{B-C}$ in Eq. (4.2) using Jensen's inequality:

$$\underbrace{\mathbb{E}_{x,z \sim p(x,z)}[r^2(z|x)]}_A \geq \underbrace{\mathbb{E}_{y,z \sim p(y,z)} \left[(\mathbb{E}_{x \sim p(x|y,z)}[r(z|x)])^2 \right]}_B \geq \underbrace{\mathbb{E}_{z \sim p(z)} \left[(\mathbb{E}_{x \sim p(x|z)}[r(z|x)])^2 \right]}_C \quad (4.5)$$

The equality between A and B holds when the perturbation $r(z|x)$ is constant w.r.t. x for any z ; the equality between B and C holds when $\mathbb{E}_{x \sim p(x|y,z)}[r(z|x)]$ is constant w.r.t. y for any z . Therefore, the minimization of $\frac{A-C}{B-C}$ encourages the relative perturbation function $r(z|x)$ to be as constant w.r.t. x as possible (minimizing intra-class difference), but as different w.r.t. different y as possible (maximizing inter-class difference), resulting in a *clustering* of the values of $r(z|x)$ for different examples x according to their class y . Because of this clustering property in classification problems, we conjecture that there are at most

$|\mathcal{Y}| - 1$ phase transitions, where $|\mathcal{Y}|$ is the number of classes, with each phase transition differentiating one or more classes.

4.4.2 Representational Maximum Correlation

Under certain conditions we can further simplify $G[p(z|x)]$ and gain a deeper understanding of it. Firstly, inspired by maximum correlation [AGKN13], we introduce two new concepts, *representational maximum correlation* and *conditional maximum correlation*, as follows.

Def. 5. Given a joint distribution $p(X, Y)$, and a representation Z satisfying the Markov chain $Z - X - Y$, the representational maximum correlation $\rho_r(X, Y; Z)$ is defined as

$$\rho_r(X, Y; Z) := \sup_{(f(x,z), g(y,z)) \in \mathbf{S}_1} \mathbb{E}_{x,y,z \sim p(x,y,z)}[f(x, z)g(y, z)] \quad (4.6)$$

where $\mathbf{S}_1 = \{(f : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbf{R}, g : \mathcal{Y} \times \mathcal{Z} \rightarrow \mathbf{R}) \mid f, g \text{ bounded, and } \mathbb{E}_{x \sim p(x|z)}[f(x, z)] = \mathbb{E}_{y \sim p(y|z)}[g(y, z)] = 0, \mathbb{E}_{x,z \sim p(x,z)}[f^2(x, z)] = \mathbb{E}_{y,z \sim p(y,z)}[g^2(y, z)] = 1\}$.

The conditional maximum correlation $\rho_m(X, Y|Z)$ is defined as:

$$\rho_m(X, Y|Z) := \sup_{(f(x), g(y)) \in \mathbf{S}_2} \mathbb{E}_{x,y \sim p(x,y|z)}[f(x)g(y)] \quad (4.7)$$

where $\mathbf{S}_2 = \{(f : \mathcal{X} \rightarrow \mathbf{R}, g : \mathcal{Y} \rightarrow \mathbf{R}) \mid f, g \text{ bounded, and } \forall z \in \mathcal{Z} : \mathbb{E}_{x \sim p(x|z)}[f(x)] = \mathbb{E}_{y \sim p(y|z)}[g(y)] = 0, \mathbb{E}_{x \sim p(x|z)}[f^2(x)] = \mathbb{E}_{y \sim p(y|z)}[g^2(y)] = 1\}$.

We prove the following Theorem 7, which expresses $G[p(z|x)]$ in terms of representational maximum correlation and related quantities, with proof given in Appendix A.3.6.

Theorem 7. Define $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)} := \{r(z|x) : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbf{R} \mid r \text{ bounded}\}$. If $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)}$ and $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ satisfy: $\forall r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)}$, there exists³ $r_1(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$, $s(z) \in \{s(z) : \mathcal{Z} \rightarrow \mathbf{R} \mid s \text{ bounded}\}$ s.t. $r(z|x) = r_1(z|x) + s(z)$, then we have:

³For discrete X, Z such that the cardinality $|\mathcal{Z}| \geq |\mathcal{X}|$, this is generally true since in this scenario, $h(x, z)$ and $s(z)$ have $|\mathcal{X}||\mathcal{Z}| + |\mathcal{Z}|$ unknown variables, but the condition has only $|\mathcal{X}||\mathcal{Z}| + |\mathcal{X}|$ linear equations. The difference between $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ and $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)}$ is that $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)}$ does not have the requirement of $\mathbb{E}_{p(z|x)}[r(z|x)] = 0$. Combined with Lemma 13.3, this condition allows us to replace $r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ by $r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)}$ in Eq. (4.2).

(i) *The representation maximum correlation and G :*

$$G[p(z|x)] = \frac{1}{\rho_r^2(X, Y; Z)} \quad (4.8)$$

(ii) *The representational maximum correlation and conditional maximum correlation:*

$$\rho_r(X, Y; Z) = \sup_{Z \in \mathcal{Z}} [\rho_m(X, Y|Z)] \quad (4.9)$$

(iii) *When Z is continuous, an optimal relative perturbation function $r(z|x)$ for $G[p(z|x)]$ is given by*

$$r^*(z|x) = h^*(x) \sqrt{\frac{\delta(z - z^*)}{p(z)}} \quad (4.10)$$

where $z^* = \arg \max_{z \in \mathcal{Z}} \rho_m(X, Y|Z = z)$, and $h^*(x)$ is the optimal solution for the learnability threshold function $h^*(x) = \arg \min_{h(x) \in \{h: \mathcal{X} \rightarrow \mathbf{R} \mid h \text{ bounded}\}} \beta_0[h(x)]$ with $p(X, Y|Z = z^*)$ ($\beta_0[h(x)]$ is given in Theorem 4 of [WFCT19a]).

(iv) *For discrete X, Y and Z , we have*

$$\rho_r(X, Y; Z) = \max_{Z \in \mathcal{Z}} \sigma_2(Z) \quad (4.11)$$

where $\sigma_2(Z)$ is the second largest singular value of the matrix $Q_{X,Y|Z} := \left(\frac{p(x,y|z)}{\sqrt{p(x|z)p(y|z)}} \right)_{x,y} = \left(\frac{p(x,y)}{\sqrt{p(x)p(y)}} \sqrt{\frac{p(z|x)}{p(z|y)}} \right)_{x,y}$.

Theorem 7 furthers our understanding of $G[p(z|x)]$ and the phase transition condition (Theorem 6), which we elaborate as follows.

Discovering maximum correlation in the orthogonal space of a learned representation:

Intuitively, the representational maximum correlation measures the maximum linear correlation between $f(X, Z)$ and $g(Y, Z)$ among all real-valued functions f, g , under the constraint

that $f(X, Z)$ is “orthogonal” to $p(X|Z)$ and $g(Y, Z)$ is “orthogonal” to $p(Y|Z)$. Theorem 7 (i) reveals that $G[p(z|x)]$ is the inverse square of this representational maximum correlation. Theorem 7 (ii) further shows that $G[p(z|x)]$ is finding a specific z^* on which maximum (nonlinear) correlation between X and Y conditioned on Z can be found. Combined with Theorem 6, we have that when we continuously increase β , for the optimal representation Z_β^* given by $p_\beta^*(z|x)$ at β , $\rho_r(X, Y; Z_\beta^*)$ shall monotonically decrease due to that X and Y has to find their maximum correlation on the orthogonal space of an increasingly better representation Z_β^* that captures more information about X . A phase transition occurs when $\rho_r(X, Y; Z_\beta^*)$ reduces to $\frac{1}{\sqrt{\beta}}$, after which as β continues to increase, $\rho_r(X, Y; Z_\beta^*)$ will try to find maximum correlation between X and Y orthogonal to the full previously learned representation. This is reminiscent of canonical-correlation analysis (CCA) [Hot92] in linear settings, where components with decreasing linear maximum correlation that are orthogonal to previous components are found one by one. In comparison, we show that in IB, each phase transition corresponds to learning a new *nonlinear* component of maximum correlation between X and Y in Z , orthogonal to the previously-learned Z . In the case of classification where different classes may have different difficulty (e.g. due to label noise or support overlap), we should expect that classes that are less difficult as measured by a larger maximum correlation between X and Y are learned earlier.

Conspicuous subset conditioned on a single z : Furthermore, we show in (iii) that an optimal relative perturbation function $r(z|x)$ can be decomposed into a product of two factors, a $\sqrt{\frac{\delta(z-z^*)}{p(z)}}$ factor that only focus on perturbing a specific point z^* in the representation space, and an $h^*(x)$ factor that is finding the “conspicuous subset” [WFCT19a], i.e. the most confident, large, typical, and imbalanced subset in the X space for the distribution $(X, Y) \sim p(X, Y|z^*)$.

Singular values In categorical settings, (iv) reveals a connection between $G[p(z|x)]$ and the singular value of the $Q_{X,Y|Z}$ matrix. Due to the property of SVD, we know that the square of the singular values of $Q_{X,Y|Z}$ equals the non-negative eigenvalue of the matrix $Q_{X,Y|Z}^T Q_{X,Y|Z}$. Then the phase transition condition in Theorem 6 is equivalent to

a (nonlinear) eigenvalue problem. This is resonant with previous analogy with CCA in linear settings, and is also reminiscent of the linear eigenvalue problem in Gaussian IB [CGTW05].

4.5 Algorithm for phase transitions discovery in classification

As a consequence of the theoretical analysis above, we are able to derive an algorithm to efficiently estimate the phase transitions for a given model architecture and dataset. This algorithm also permits us to empirically confirm some of our theoretical results in Section 4.6.

Typically, classification involves high-dimensional inputs X . Without sweeping the full range of β where at each β it is a full learning problem, it is in general a difficult task to estimate the phase transitions. In Algorithm 2, we present a two-stage approach.

In the first stage, we train a single maximum likelihood neural network f_θ with the same encoder architecture as in the (variational) IB to estimate $p(y|x)$, and obtain an $N \times C$ matrix $p(y|x)$, where N is the number of examples in the dataset and C is the number of classes. In the second stage, we perform an iterative algorithm w.r.t. G and β , alternatively, to converge to a phase transition point.

Specifically, for a given β , we use a Blahut-Arimoto type IB algorithm [TPB00] to efficiently reach IB optimal $p_\beta^*(z|x)$ at β , then use SVD (with the formula given in Theorem 7 (iv)) to efficiently estimate $G[p_\beta^*(z|x)]$ at β (step 8). We then use the $G[p_\beta^*(z|x)]$ value as the new β and do it again (step 7 in the next iteration). At convergence, we will reach the phase transition point given by $G[p_\beta^*(z|x)] = \beta$ (Theorem 6). After convergence as measured by patience parameter K , we slightly increase β by δ (step 13), so that the algorithm can discover the subsequent phase transitions.

4.6 Empirical study

We quantitatively and qualitatively test the ability of our theory and Algorithm 2 to provide good predictions for IB phase transitions. We first verify them in fully categorical settings, where X, Y, Z are all discrete, and we show that the phase transitions can correspond to learning new classes as we increase β . We then test our algorithm on versions of the MNIST and CIFAR10 datasets with added label noise.

4.6.1 Categorical dataset

For categorical datasets, X and Y are discrete, and $p(X)$ and $p(Y|X)$ are given. To test Theorem 6, we use the Blahut-Arimoto IB algorithm to compute the optimal $p_\beta^*(z|x)$ for each β . $I(Y; Z^*)$ vs. β is plotted in Fig. 4-2 (a). There are two phase transitions at β_0^c and β_1^c . For each β and the corresponding $p_\beta^*(z|x)$, we use the SVD formula (Theorem 7) to compute $G[p_\beta^*(z|x)]$, shown in Fig. 4-2 (b). We see that $G[p_\beta^*(z|x)] = \beta$ at exactly the observed phase transition points β_0^c and β_1^c . Moreover, starting at $\beta = 1$, Alg. 1 converges to each phase transition points within few iterations. Our other experiments with random categorical datasets show similarly tight matches.

Furthermore, in Appendix A.3.7 we show that the phase transitions correspond to the onset of separation of $p(z|x)$ for subsets of X that correspond to different classes. This supports our conjecture from Section 4.4.1 that there are at most $|\mathcal{Y}| - 1$ phase transitions in classification problems.

4.6.2 MNIST dataset

For continuous X , how does our algorithm perform, and will it reveal aspects of the dataset? We first test our algorithm in a 4-class MNIST with noisy labels⁴, whose confusion matrix and experimental settings are given in Appendix A.3.8. Fig. 4-3 (a) shows the path Alg. 2

⁴We use 4 classes since it is simpler than the full 10 classes, but still potentially possesses phase transitions. We use noisy label to mimic realistic settings where the data may be noisy and also to have controllable difficulty for different classes.

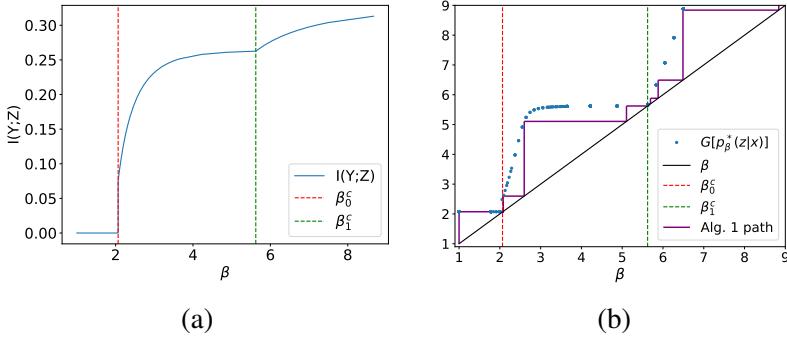


Figure 4-2: (a) $I(Y; Z^*)$ vs. β for a categorical dataset with $|X| = |Y| = |Z| = 3$, where Z^* is given by $p_\beta^*(z|x)$, and the vertical lines are the experimentally discovered phase transition points β_0^c and β_1^c . (b) $G[p_\beta^*(z|x)]$ vs. β for the same dataset, and the path for Alg. 2, with β_0^c and β_1^c in (a) also plotted. The dataset is given in Fig. A-3.

takes. We see again that in each phase Alg. 2 converges to the phase transition points within a few iterations, and it discovers in total 3 phase transition points. Similar to the categorical case, we expect that each phase transition corresponds to the onset of learning a new class, and that the last class is much harder to learn due to a larger separation of β . Therefore, this class should have a much larger label noise so that it is hard to capture this component of maximum correlation between X and Y , as analyzed in representational maximum correlation (Section 4.4.2). Fig. 4-3 (b) plots the per-class accuracy with increasing β for running the Conditional Entropy Bottleneck [Fis18] (another variational bound on IB). We see that the first two predicted phase transition points β_0^c, β_1^c closely match the observed onset of learning class 3 and class 0. Class 1 is observed to learn earlier than expected, possibly due to the gap between the variational IB objective and the true IB objective in continuous settings. By looking at the confusion matrix for the label noise (Fig. A-5), we see that the ordering of onset of learning: class 2, 3, 0, 1, corresponds exactly to the decreasing diagonal element $p(\tilde{y} = 1|y = 1)$ (increasing noise) of the classes, and as predicted, class 1 has a much smaller diagonal element $p(\tilde{y} = 1|y = 1)$ than the other three classes, which makes it much more difficult to learn. This ordering of classes by difficulty is what our representational maximum correlation predicts.

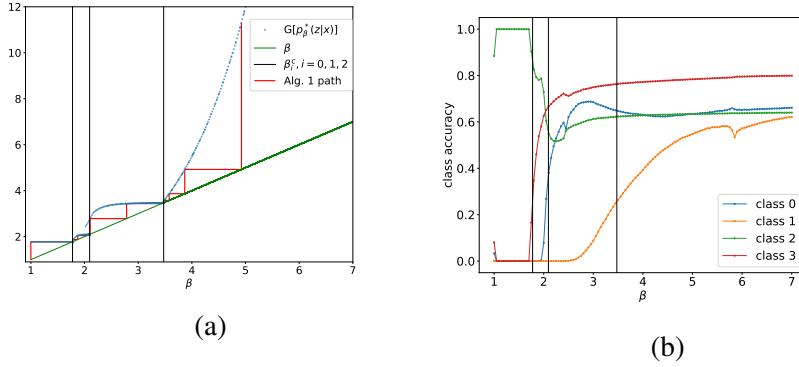


Figure 4-3: (a) Path of Alg. 2 starting with $\beta = 1$, where the maximum likelihood model f_θ is using the same encoder architecture as in the CEB model. This stairstep path shows that Alg. 2 is able to ignore very large regions of β , while quickly and precisely finding the phase transition points. Also plotted is an accumulation of $G[p_\beta^*(z|x)]$ vs. β by running Alg. 1 with varying starting β (blue dots). (b) Per-class accuracy vs. β , where the accuracy at each β is from training an independent CEB model on the dataset. The per-class accuracy denotes the fraction of correctly predicted labels by the CEB model for the observed label \hat{y} .

4.6.3 CIFAR10 dataset

Finally, we investigate the CIFAR10 experiment from Section 4.1. The details of the experimental setup are described in Appendix A.3.9. This experiment stretches the current limits of our discrete approximation to the underlying continuous representation being learned by the models. Nevertheless, we can see in Fig. 4-4 that many of the visible empirical phase transitions are tightly identified by Alg. 2. Particularly, the onset of learning is predicted quite accurately; the large interval between the predicted $\beta_3 = 1.21$ and $\beta_4 = 1.61$ corresponds well to the continuous increase of $I(X; Z)$ and $I(Y; Z)$ at the same interval. And Alg. 1 is able to identify many dense transitions not obviously seen by just looking at $I(Y; Z)$ vs. β curve alone. Alg. 1 predicts 9 phase transitions, exactly equal to $|\mathcal{Y}| - 1$ for CIFAR10.

4.7 Conclusion

In this work, we observe and study the phase transitions in IB as we vary β . We introduce the definition for IB phase transitions, and based on it derive a formula that gives a practical

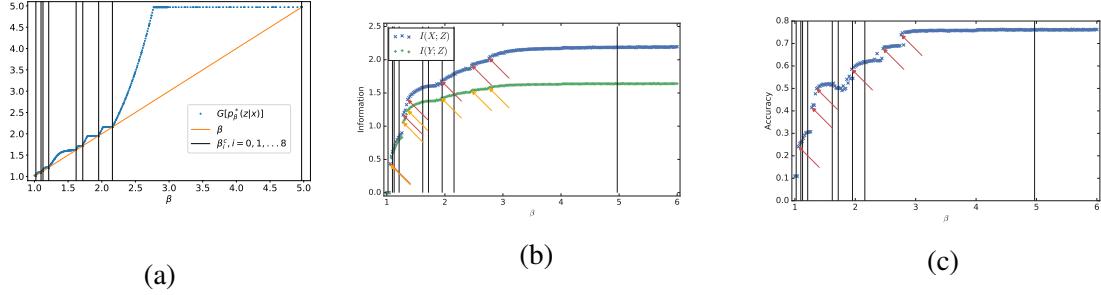


Figure 4-4: (a) Accumulated $G[p_\beta^*(z|x)]$ vs. β by running Alg. 1 with varying starting β (blue dots). Also plotted are predicted phase transition points. (b) $I(X; Z)$ and $I(Y; Z)$ vs. β . The manually-identified phase transition points are labelled with arrows. The vertical black lines are the phase transitions identified by Alg. 2, denoted as $\beta_0^c, \beta_1^c, \dots, \beta_8^c$, from left to right. (c) Accuracy vs. β with the same sets of points identified. The most interesting region is right before $\beta = 2$, where accuracy *decreases* with β . Alg. 2 identifies both sides of that region, as well as points at or near all of the early obvious phase transitions. It also seems to miss later transitions, possibly due to the gap between the variational IB objective and the true IB objective in continuous settings.

condition for IB phase transitions. We further understand the formula via Jensen’s inequality and representational maximum correlation. We reveal the close interplay between the IB objective, the dataset and the learned representation, as each phase transition is learning a nonlinear maximum correlation component in the orthogonal space of the learned representation. We present an algorithm for finding the phase transitions, and show that it gives tight matches with observed phase transitions in categorical datasets, predicts onset of learning new classes and class difficulty in MNIST, and predicts prominent transitions in CIFAR10 experiments. This work is a first theoretical step towards a deeper understanding of the phenomenon of phase transitions in the Information Bottleneck. We believe our approach will be applicable to other “trade-off” objectives, like β -VAE [HMP⁺17] and InfoDropout [AS18a], where the model’s ability to predict is balanced against a measure of complexity.

Algorithm 2 Phase transitions discovery for IB

Require (X, Y) : the dataset
Require f_θ : a neural net with the same encoder architecture as the (variational) IB
Require K : patience
Require δ : precision floor
Require R : maximum ratio between $\beta^{(\text{th})}$ and β .
// First stage: fit $p(y|x)$ using neural net f_θ :
1: $p(y|x) \leftarrow$ fitting (X, Y) using f_θ via maximum likelihood.
2: $p(x) \leftarrow \frac{1}{N}$

// Second stage: coordinate descent using $G[p(z|x)]$ and IB algorithm:
3: $\beta_0^c \leftarrow \beta^{(\text{th})}(1)$
4: $\mathbb{B} \leftarrow \{\beta_0^c\}$ // \mathbb{B} is a set collecting the phase transition points
5: $(\beta^{(\text{new})}, \beta, \text{count}) \leftarrow (\beta_0^c, 1, 0)$
6: **while** $\frac{\beta^{(\text{new})}}{\beta} < R$ **do**:
7: $\beta \leftarrow \beta^{(\text{new})}$
8: $\beta^{(\text{new})} \leftarrow \beta^{(\text{th})}(\beta)$
9: **if** $\beta^{(\text{new})} - \beta < \delta$ **do**:
10: $\text{count} \leftarrow \text{count} + 1$
11: **if** $\text{count} > K$ **do**:
12: $\mathbb{B} \leftarrow \mathbb{B} \cup \{\beta^{(\text{new})}\}$
13: $\beta^{(\text{new})} \leftarrow \beta^{(\text{new})} + \delta$
14: **end if**
15: **else**: $\text{count} \leftarrow 0$
16: **end if**
17: **end while**
18: **return** \mathbb{B}

subroutine $\beta^{(\text{th})}(\beta)$:
s1: Compute $p_\beta^*(z|x)$ using the IB algorithm [TPB00].
s2: $\beta^{(\text{new})} \leftarrow G[p_\beta^*(z|x)]$ using SVD (Eq. 4.8 and 4.11).
s3: **return** $\beta^{(\text{new})}$

Chapter 5

Pareto-optimal data compression for binary classification tasks

The¹² goal of lossy data compression is to reduce the storage cost of a data set X while retaining as much information as possible about something (Y) that you care about. For example, what aspects of an image X contain the most information about whether it depicts a cat? Mathematically, this corresponds to finding a mapping $X \rightarrow Z \equiv f(X)$ that maximizes the mutual information $I(Z, Y)$ while the entropy $H(Z)$ is kept below some fixed threshold. We present a new method for mapping out the Pareto frontier for classification tasks, reflecting the tradeoff between retained entropy and class information. We first show how a random variable X (an image, say) drawn from a class $Y \in \{1, \dots, n\}$ can be distilled into a vector $W = f(X) \in \mathbb{R}^{n-1}$ losslessly, so that $I(W, Y) = I(X, Y)$; for example, for a binary classification task of cats and dogs, each image X is mapped into a single real number W retaining all information that helps distinguish cats from dogs. For the $n = 2$ case of binary classification, we then show how W can be further compressed into a discrete variable $Z = g_\beta(W) \in \{1, \dots, m_\beta\}$ by binning W into m_β bins, in such a way that varying the parameter β sweeps out the full Pareto frontier, solving a generalization of the Discrete Information Bottleneck (DIB) problem. We argue that the most interesting points

¹The paper “Pareto-optimal data compression for binary classification tasks” is Published at *Entropy* 2020, 22(1), 7. Authors: Tegmark, Max and Tailin Wu. arXiv:1908.08961.

²The code is open-sourced at github.com/tailintalent/distillation.

on this frontier are “corners” maximizing $I(Z, Y)$ for a fixed number of bins $m = 2, 3\dots$ which can be conveniently be found without multiobjective optimization. We apply this method to the CIFAR-10, MNIST and Fashion-MNIST datasets, illustrating how it can be interpreted as an information-theoretically optimal image clustering algorithm. We find that these Pareto frontiers are not concave, and that recently reported DIB phase transitions correspond to transitions between these corners, changing the number of clusters.

5.1 Introduction

A core challenge in science, and in life quite generally, is data distillation: keeping only a manageable small fraction of our available data X while retaining as much information as possible about something (Y) that we care about. For example, what aspects of an image contain the most information about whether it depicts a cat ($Y = 1$) rather than a dog ($Y = 2$)? Mathematically, this motivates finding a mapping $X \rightarrow Z \equiv g(X)$ that maximizes the mutual information $I(Z, Y)$ while the entropy $H(Z)$ is kept below some fixed threshold. The tradeoff between $H_* = H(Z)$ (bits stored) and $I_* = I(Z, Y)$ (useful bits) is described by a Pareto frontier, defined as

$$I_*(H_*) \equiv \sup_{\{g: H[g(X)] \leq H_*\}} I[g(X), Y], \quad (5.1)$$

and illustrated in Figure 5-1 (this is for a toy example described below; we compute the Pareto frontier for our cat/dog example in Section 5.3). The shaded region is impossible because $I(Z, Y) \leq I(X, Y)$ and $I(Z, Y) \leq H(Z)$. The colored dots correspond to random likelihood binnings into various numbers of bins, as described in the next section, and the upper envelope of all attainable points define the Pareto frontier. Its “corners”, which are marked by black dots and maximize $I(Z, Y)$ for M bins ($M = 1, 2, \dots$), are seen to lie close to the vertical dashed lines $H(Z) = \log M$, corresponding to all bins having equal size. We plot the H -axis flipped to conform with the tradition that up and to the right are more desirable. The core goal of this paper is to present a method for computing such Pareto frontiers.

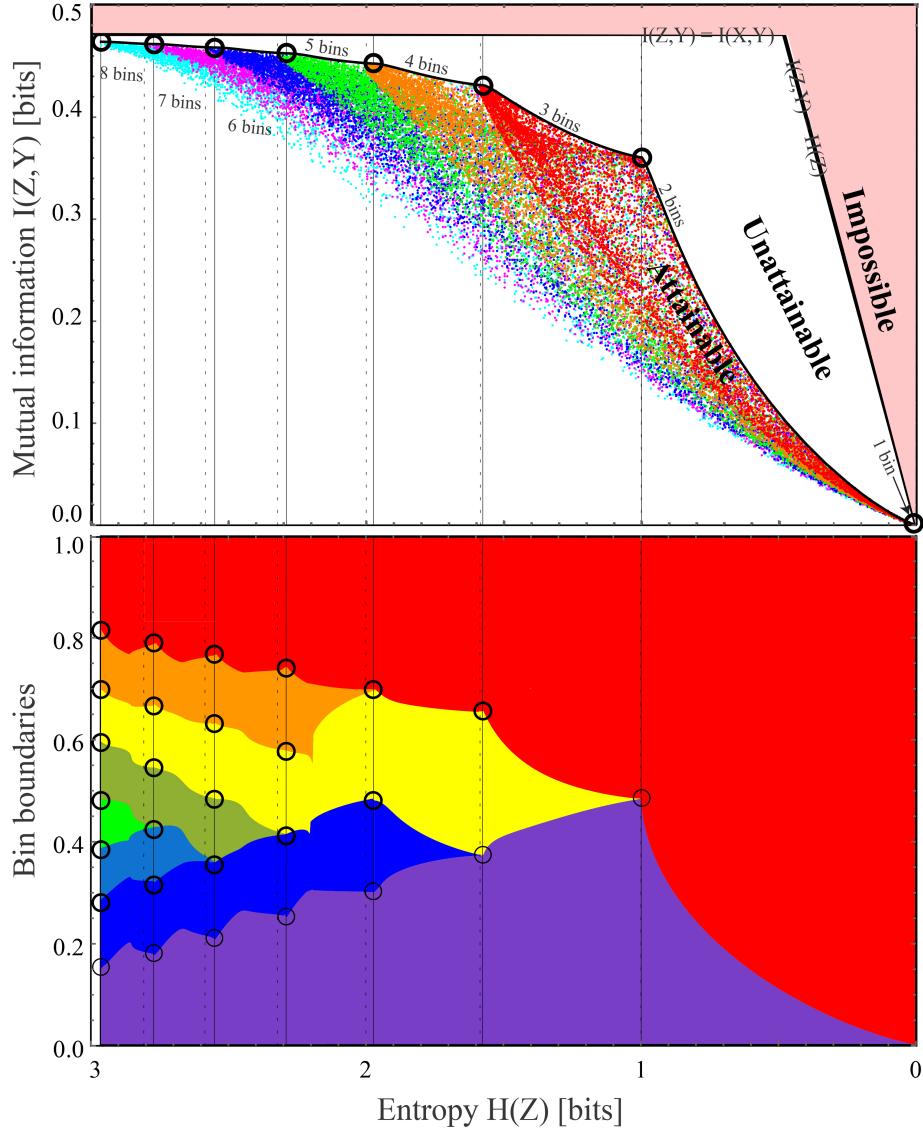


Figure 5-1: The Pareto frontier (top panel) for compressed versions $Z = g(X)$ of our warmup dataset $X \in [0, 1]^2$ with classes $Y \in \{1, 2\}$, showing the maximum attainable class information $I(Z, Y)$ for a given entropy $H(Z)$, mapped using the method described in this paper using the likelihood binning in the bottom panel.

5.1.1 Objectives & relation to prior work

In other words, the goal of this paper is to analyze soft rather than hard classifiers: not to make the most accurate classifier, but rather to compute the Pareto frontier that reveals the most accurate (in an information-theoretic sense) classifier Z given a constraint on its bit content $H(Z)$. This Pareto frontier challenge is part of the broader quest for data distillation: lossy data compression that retains as much as possible of the information that is useful to us. Ideally, the information can be partitioned into a set of independent chunks and sorted

from most to least useful, enabling us to select the number of chunks to retain so as to optimize our tradeoff between utility and data size. Consider two random variables X and Y which may each be vectors or scalars. For simplicity, consider them to be discrete with finite entropy³. For prediction tasks, we might interpret Y as the future state of a dynamical system that we wish to predict from the present state X . For classification tasks, we might interpret Y as a class label that we wish to predict from an image, sound, video or text string X . Let us now consider various forms of ideal data distillation, as summarized in Table 5.1.

Random vectors	What is distilled?	Probability distribution	
		Gaussian	Non-Gaussian
1	Entropy $H(X) = \sum_i H(Z_i)$	PCA $\mathbf{z} = \mathbf{Fx}$	Autoencoder $Z = f(X)$
2	Mutual information $I(X, Y) = \sum_i I(Z_i, Z'_i)$	CCA $\mathbf{z} = \mathbf{Fx}$ $\mathbf{z}' = \mathcal{G}\mathbf{y}$	Latent reps $Z = f(X)$ $Z' = g(Y)$

Table 5.1: Data distillation: the relationship between Principal Component Analysis (PCA), Canonical Correlation Analysis (CCA), nonlinear autoencoders and nonlinear latent representations.

If we distill X as a whole, then we would ideally like to find a function f such that the so-called latent representation $Z = f(X)$ retains the full entropy $H(X) = H(Z) = \sum H(Z_i)$, decomposed into independent⁴ parts with vanishing mutual information: $I(Z_i, Z_j) = \delta_{ij}H(Z_i)$. For the special case where $X = \mathbf{x}$ is a vector with a multivariate Gaussian distribution, the optimal solution is Principal Component Analysis (PCA) [Pea01], which has long been a workhorse of statistical physics and many other disciplines: here f is simply a linear function mapping into the eigenbasis of the covariance matrix of \mathbf{x} . The general case remains unsolved, and it is easy to see that it is hard: if $X = c(Z)$ where c implements some state-of-the-art cryptographic code, then finding $f = c^{-1}$ (to recover the independent pieces of information and discard the useless parts) would generically require breaking the code. Great progress has nonetheless been made for many special cases, using techniques

³The discreteness restriction loses us no generality in practice, since since we can always discretize real numbers by rounding them to some very large number of significant digits.

⁴When implementing any distillation algorithm in practice, there is always a one-parameter tradeoff between compression and information retention which defines a Pareto frontier. A key advantage of the latent variables (or variable pairs) being statistically independent is that this allows the Pareto frontier to be trivially computed, by simply sorting them by decreasing information content and varying the number retained.

such as nonlinear autoencoders [VLBM08] and Generative Adversarial Networks (GANs) [GPAM⁺14].

Now consider the case where we wish to distill X and Y separately, into $Z \equiv f(X)$ and $Z' = g(Y)$, retaining the mutual information between the two parts. Then we ideally have $I(X, Y) = \sum_i I(Z_i, Z'_i)$, $I(Z_i, Z_j) = \delta_{ij} H(Z_i)$, $I(Z'_i, Z'_j) = \delta_{ij} H(Z'_i)$, $I(Z_i, Z'_j) = \delta_{ij} I(Z_i, Z'_j)$. This problem has attracted great interest, especially for time series where $X = \mathbf{u}_i$ and $Y = \mathbf{u}_j$ for some sequence of states \mathbf{u}_k ($k = 0, 1, 2, \dots$) in physics or other fields, where one typically maps the state vectors \mathbf{u}_i into some lower-dimensional vectors $f(\mathbf{u}_i)$, after which the prediction is carried out in this latent space. For the special case where X has a multivariate Gaussian distribution, the optimal solution is Canonical Correlation Analysis (CCA) [Hot36]: here both f and g are linear functions, computed via a singular-value decomposition (SVD) [EY36] of the cross-correlation matrix after prewhitening X and Y . The general case remains unsolved, and is obviously even harder than the above-mentioned 1-vector autoencoding problem. The recent work [OLV18, CLB19] review the state-of-the art as well as presenting Contrastive Predictive Coding and Dynamic Component Analysis, powerful new distillation techniques for time series, following the long tradition of setting $f = g$ even though this is provably not optimal for the Gaussian case as shown in [Teg19].

The goal of this paper is to make progress in the lower right quadrant of Table 5.1. We will first show that if $Y \in \{1, 2\}$ (as in binary classification tasks) and we can successfully train a classifier that correctly predicts the conditional probability distribution $p(Y|X)$, then it can be used to provide an exact solution to the distillation problem, losslessly distilling X into a single real variable $W = f(X)$. We will generalize this to an arbitrary classification problem $Y \in \{1, \dots, n\}$ by losslessly distilling X into a vector $W = f(X) \in \mathbb{R}^{n-1}$, although in this case, the components of the vector W may not be independent. We will then return to the binary classification case and provide a family of binnings that map W into an integer Z , allowing us to scan the full Pareto frontier reflecting the tradeoff between retained entropy and class information, illustrating the end-to-end procedure with the CIFAR-10, MNIST and Fashion-MNIST datasets. This is related to the work of [KY14] which maximizes $I(Z, Y)$

for a fixed number of bins (instead of for a fixed entropy), which corresponds to the “corners” seen in Figure 5-1.

This work is closely related to the Information Bottleneck (IB) method [TPB00], which provides an insightful, principled approach for balancing compression against prediction [TMBS19]. Just as in our work, the IB method aims to find a random variable $Z = f(X)$ that loosely speaking retains as much information as possible about Y and as little other information as possible. The IB method implements this by maximizing the IB-objective

$$\mathcal{L}_{\text{IB}} = I(Z, Y) - \beta I(Z, X) \quad (5.2)$$

where the Lagrange multiplier β tunes the balance between knowing about Y and forgetting about X . [SS17a] considered the alternative Deterministic Information Bottleneck (DIB) objective

$$\mathcal{L}_{\text{DIB}} = I(Z, Y) - \beta H(Z), \quad (5.3)$$

to close the loophole where Z retains random information that is independent of both X and Y (which is possible if f is function that contains random components rather than fully deterministic⁵). However, there is a well-known problem with this objective that occurs when $Z \in \mathbb{R}^n$ is continuous [AG19]: $H(Z)$ is strictly speaking infinite, since it requires an infinite amount of information to store the infinitely many decimals of a generic real number. Although this infinity is normally regularized away by only defining $H(Z)$ up to an additive constant, which is irrelevant when minimizing (5.3), the problem is that we can define a new rescaled random variable

$$Z' = aZ \quad (5.4)$$

for a constant $a \neq 0$ and obtain⁶

$$I(Z', X) = I(Z, X) \quad (5.5)$$

⁵If $Z = f(X)$ for some deterministic function f , which is typically not the case in the popular variational IB-implementation [AFDM16, CMT16, Fis18], then $H(Z|X) = 0$, so $I(Z, X) \equiv H(Z) - H(Z|X) = H(Z)$, which means the two objectives (5.2) and (5.3) are identical.

⁶Throughout this paper, we take log to denote the logarithm in base 2, so that entropy and mutual information are measured in bits.

and

$$H(Z') = H(Z) + n \log|a|. \quad (5.6)$$

This means that by choosing $|a| \ll 1$, we can make $H(Z')$ arbitrarily negative while keeping $I(Z', X)$ unchanged, thus making \mathcal{L}_{DIB} arbitrarily negative. The objective \mathcal{L}_{DIB} is therefore not bounded from below, and trying to minimize it will not produce an interesting result. We will eliminate this Z -rescaling problem by making Z discrete rather than continuous, so that $H(Z)$ is always well-defined and finite. Another challenge with the DIB objective of equation (5.3), which we will also overcome, is that it maximizes a linear combination of the two axes in Figure 5-1, and can therefore only discover concave parts of the Pareto frontier, not convex ones (which are seen to dominate in Figure 5-1).

The rest of this paper is organized as follows: In Section 5.2.1, we will provide an exact solution for the binary classification problem where $Y \in \{1, 2\}$ by losslessly distilling X into a single real variable $Z = f(X)$. We also generalize this to an arbitrary classification problem $Y \in \{1, \dots, n\}$ by losslessly distilling X into a vector $W = f(X) \in \mathbb{R}^{n-1}$, although the components of the vector W may not be independent. In Section 5.2.2, we return to the binary classification case and provide a family of binnings that map Z into an integer, allowing us to scan the full Pareto frontier reflecting the tradeoff between retained entropy and class information. We apply our method to various image datasets in Section 5.3 and discuss our conclusions in Section 5.4.

5.2 Method

Our algorithm for mapping the Pareto frontier transforms our original data set X in a series of steps which will be described in turn below:

$$X \xrightarrow{w} W \mapsto W_{\text{uniform}} \mapsto W_{\text{binned}} \mapsto W_{\text{sorted}} \xrightarrow{B} Z. \quad (5.7)$$

As we will show, the first, second and fourth transformations retain all mutual information with the label Y , and the information loss about Y can be kept arbitrarily small in the

third step. In contrast, the last step treats the information loss as a tuneable parameter that parameterizes the Pareto frontier.

5.2.1 Lossless distillation for classification tasks

Our first step is to compress X (an image, say) into W , a set of $n - 1$ real numbers, in such a way that no class information is lost about $Y \in \{1, \dots, n\}$.

Theorem 8. (Lossless Distillation Theorem): *For an arbitrary random variable X and a categorical random variable $Y \in \{1, \dots, n\}$, we have*

$$P(Y|X) = P(Y|W), \quad (5.8)$$

where $W \equiv w(X) \in \mathbb{R}^{n-1}$ is defined by⁷

$$w_i(X) \equiv P(Y = i|X). \quad (5.9)$$

Proof. Let S denote the domain of X , *i.e.*, $X \in S$, and define the set-valued function

$$s(W) \equiv \{x \in S : w(x) = W\}.$$

These sets $s(W)$ form a partition of S parameterized by W , since they are disjoint and

$$\cup_{W \in \mathbb{R}^{n-1}} s(W) = S. \quad (5.10)$$

For example, if $S = \mathbb{R}^2$ and $n = 2$, then the sets $s(W)$ are simply contour curves of the conditional probability $W \equiv P(Y = 1|X) \in \mathbb{R}$. This partition enables us to uniquely specify X as the pair $\{W, X_W\}$ by first specifying which set $s[f(X)]$ it belongs to (determined by $W = f(X)$), and then specifying the particular element within that set, which we denote

⁷Note that we ignore the n^{th} component since it is redundant: $w_n(X) = 1 - \sum_i^{n-1} w_i(X)$.

$X_W \in S(W)$. This implies that

$$P(Y|X) = P(Y|W, X_W) = P(Y|W), \quad (5.11)$$

completing the proof. The last equal sign follows from the fact that the conditional probability $P(Y|X)$ is independent of X_W , since it is by definition constant throughout the set $s(W)$. \blacksquare

The following corollary implies that W is an optimal distillation of the information X has about Y , in the sense that it constitutes a lossless compression of said information: $I(W, Y) = I(X, Y)$ as shown, and the total information content (entropy) in W cannot exceed that of X since it is a deterministic function thereof.

Corollary 8.1. *With the same notation as above, we have*

$$I(X, Y) = I(W, Y). \quad (5.12)$$

Proof. For any two random variables, we have the identity $I(U, V) = H(V) - H(V|U)$, where $I(U, V)$ is their mutual information and $H(V|U)$ denotes conditional entropy. We thus obtain

$$\begin{aligned} I(X, Y) &= H(Y) - H(Y|X) = H(Y) + \langle \log P(Y|X) \rangle_{X,Y} \\ &= H(Y) + \langle \log P(Y|W) \rangle_{W,X_W,Y} \\ &= H(Y) + \langle \log P(Y|W) \rangle_{W,Y} \\ &= H(Y) - H(Y|W) = I(W, Y), \end{aligned} \quad (5.13)$$

which completes the proof. We obtain the second line by using $P(Y|X) = P(Y|W)$ from Theorem 1 and specifying X by W and X_W , and the third line since $P(Y|W)$ is independent of X_W , as above. \blacksquare

In most situations of practical interest, the conditional probability distribution $P(Y|X)$ is not precisely known, but can be approximated by training a neural-network-based classifier

that outputs the probability distribution for Y given any input X . We present such examples in Section 5.3. The better the classifier, the smaller the information loss $I(X, Y) - I(W, Y)$ will be, approaching zero in the limit of an optimal classifier.

5.2.2 Pareto-optimal compression for binary classification tasks

Let us now focus on the special case where $n = 2$, *i.e.*, binary classification tasks. For example, X may correspond to images of equal numbers of felines and canines to be classified despite challenges with variable lighting, occlusion, *etc.* as in Figure 5-2, and $Y \in \{1, 2\}$ may correspond to the labels “cat” and “dog”. In this case, Y contains $H(Y) = 1$ bit of information of which $I(X, Y) \leq 1$ bit is contained in X . Theorem 8 shows that for this case, all of this information about whether an image contains a cat or a dog can be compressed into a single number W which is not a bit like Y , but a real number between zero and one.

The goal of this section is find a class of functions g that perform Pareto-optimal lossy compression of W , mapping it into an integer $Z \equiv g(W)$ that maximizes $I(Z, Y)$ for a fixed entropy $H(Z)$.⁸ The only input we need for our work in this section is the joint probability distribution $f_i(w) = P(Y=i, W=w)$, whose marginal distributions are the discrete probability distribution for P_i^Y for Y and the probability distribution f for W , which we will henceforth assume to be continuous:

$$f(w) \equiv \sum_{i=1}^2 f_i(w), \quad (5.14)$$

$$P_i^Y \equiv P(Y=i) = \int_0^1 f_i(w) dw. \quad (5.15)$$

Uniformization of W

For convenience and without loss of generality, we will henceforth assume that $f(w) = 1$, *i.e.*, that W has a uniform distribution on the unit interval $[0, 1]$. We can do this because

⁸Throughout this paper, we will use the term “Pareto-optimal” or “optimal” in this sense, *i.e.*, maximizing $I(X, Y)$ for a fixed $H(Z)$.

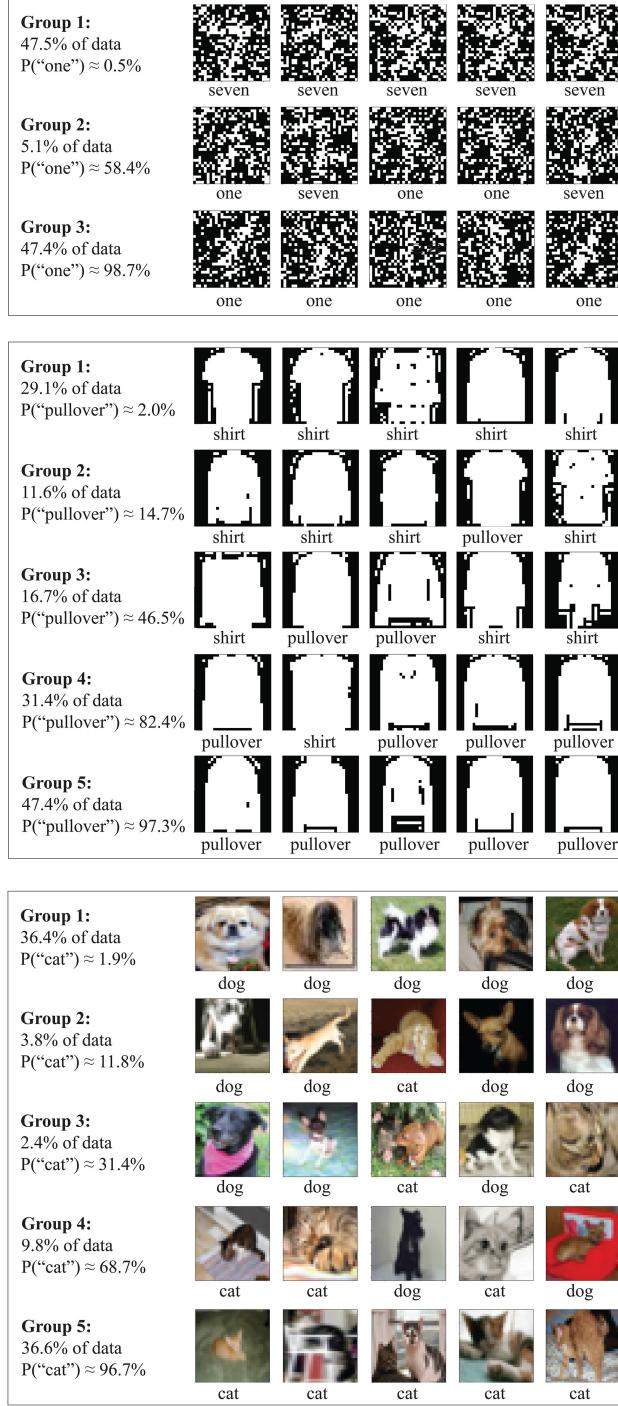


Figure 5-2: Sample data from Section 5.3. Images from MNIST (top), Fashion-MNIST (middle) and CIFAR-10 are mapped into integers (group labels) $Z = f(X)$ retaining maximum mutual information with the class variable Y (ones/sevens, shirts/pullovers and cats/dogs, respectively) for 3, 5 and 5 groups, respectively. These mappings f correspond to Pareto frontier “corners”.

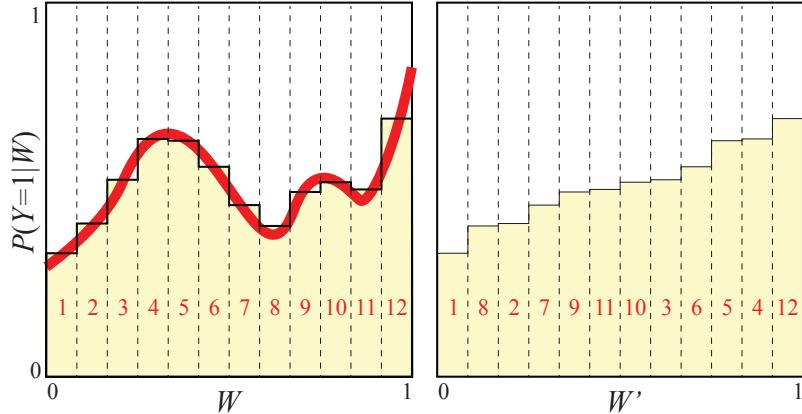


Figure 5-3: Essentially all information about Y is retained if W is binned into sufficiently narrow bins. Sorting the bins (left) to make the conditional probability monotonically increasing (right) changes neither this information nor the entropy.

if W were not uniformly distributed, we could make it so by using the standard statistical technique of applying its cumulative probability distribution function to it:

$$W \mapsto W' \equiv F(W), \quad F(w) \equiv \int_0^w f(w') dw', \quad (5.16)$$

retaining all information — $I(W', Y) = I(W, Y)$ — since this procedure is invertible almost everywhere.

Binning W

Given a set of bin boundaries $b_1 < b_2 < \dots < b_{n-1}$ grouped into a vector \mathbf{b} , we define the integer-value *contiguous binning function*

$$B(x, \mathbf{b}) \equiv \begin{cases} 1 & \text{if } x < b_1 \\ k & \text{if } b_{k-1} < x \leq b_k \\ n & \text{if } x \geq b_{N-1} \end{cases} \quad (5.17)$$

$B(x, \mathbf{b})$ can thus be interpreted as the ID of the bin into which x falls. Note that B is a monotonically increasing piecewise constant function of x that is shaped like an N -level staircase with $n - 1$ steps at b_1, \dots, b_{N-1} .

Let us now bin W into N equispaced bins, by mapping it into an integer $W' \in \{1, \dots, N\}$ (the bin ID) defined by

$$W' \equiv W_{\text{binned}} \equiv B(W, \mathbf{b}_N), \quad (5.18)$$

where \mathbf{b} is the vector with elements $b_j = j/N$, $j = 1, \dots, N - 1$. As illustrated visually in Figure 5-3 and mathematically in Appendix A.4.1, binning $W \mapsto W'$ corresponds to creating a new random variable for which the conditional distribution $p_1(w) = P(Y=1|W=w)$ is replaced by a piecewise constant function $\bar{p}_1(w)$, replacing the values in each bin by their average. The binned variable W' thus retains only information about which bin W falls into, discarding all information about the precise location within that bin. In the $N \rightarrow \infty$ limit of infinitesimal bins, $\bar{p}_1(w) \rightarrow p_1(w)$, and we expect the above-mentioned discarded information to become negligible. This intuition is formalized by 14 in Appendix A.4.1, which under mild smoothness assumptions ensuring that $p_1(w)$ is not pathological shows that

$$I(W', Y) \rightarrow I(W, Y) \quad \text{as} \quad N \rightarrow \infty, \quad (5.19)$$

i.e., that we can make the binned data W' retain essentially all the class information from W as long as we use enough bins.

In practice, such as for the numerical experiments that we will present in Section 5.3, training data is never infinite and the conditional probability function $p_1(w)$ is never known to perfect accuracy. This means that the pedantic distinction between $I(W', Y) = I(W, Y)$ and $I(W', Y) \approx I(W, Y)$ for very large N is completely irrelevant in practice. In the rest of this paper, we will therefore work with the unbinned (W) and binned (W') data somewhat interchangeably below for convenience, occasionally dropping the apostrophe ' from W' when no confusion is caused.

Making the conditional probability monotonic

For convenience and without loss of generality, we can assume that the conditional probability distribution $\bar{p}_1(w)$ is a monotonically increasing function. We can do this because if this were not the case, we could make it so by sorting the bins by increasing conditional

probability, as illustrated in Figure 5-3, because both the entropy $H(W')$ and the mutual information $I(W', Y)$ are left invariant by this renumbering/relabeling of the bins. The “cat” probability $P(Y=1)$ (the total shaded area in Figure 5-3) is of course also left unchanged by both this sorting and by the above-mentioned binning.

Proof that Pareto frontier is spanned by contiguous binnings

We are now finally ready to tackle the core goal of this paper: mapping the Pareto frontier (H_*, I_*) of optimal data compression $X \mapsto Z$ that reflects the tradeoff between $H(Z)$ and $I(Z, Y)$. While fine-grained binning has no effect on the entropy $H(Y)$ and negligible effect on $I(W, Y)$, it dramatically reduces the entropy of our data. Whereas $H(W) = \infty$ since W is continuous⁹, $H(W') = \log N$ is finite, approaching infinity only in the limit of infinitely many infinitesimal bins. Taken together, these scalings of I and H imply that the leftmost part of the Pareto frontier $I_*(H_*)$, defined by equation (5.1) and illustrated in Figure 5-1, asymptotes to a horizontal line of height $I_* = I(X, Y)$ as $H_* \rightarrow \infty$.

To reach the interesting parts of the Pareto frontier further to the right, we must destroy some information about Y . We do this by defining

$$Z = g(W'), \quad (5.20)$$

where the function g groups the tiny bins indexed by $W' \in \{1, \dots, N\}$ into fewer ones indexed by $Z \in \{1, \dots, M\}$, $M < N$. There are vast numbers of such possible groupings, since each group corresponds to one of the $2^N - 2$ nontrivial subsets of the tiny bins. Fortunately, as we will now prove, we need only consider the $\mathcal{O}(N^M)$ contiguous groupings, since non-contiguous ones are inferior and cannot lie on the Pareto frontier. Indeed, we will see that for the examples in Section 5.3, $M \lesssim 5$ suffices to capture the most interesting information.

Theorem 9. (Contiguous Binning Theorem): *If W has a uniform distribution and the*

⁹While this infinity, which reflects the infinite number of bits required to describe a single generic real number, is customarily eliminated by defining entropy only up to an overall additive constant, we will not follow that custom here, for the reason explained in the introduction.

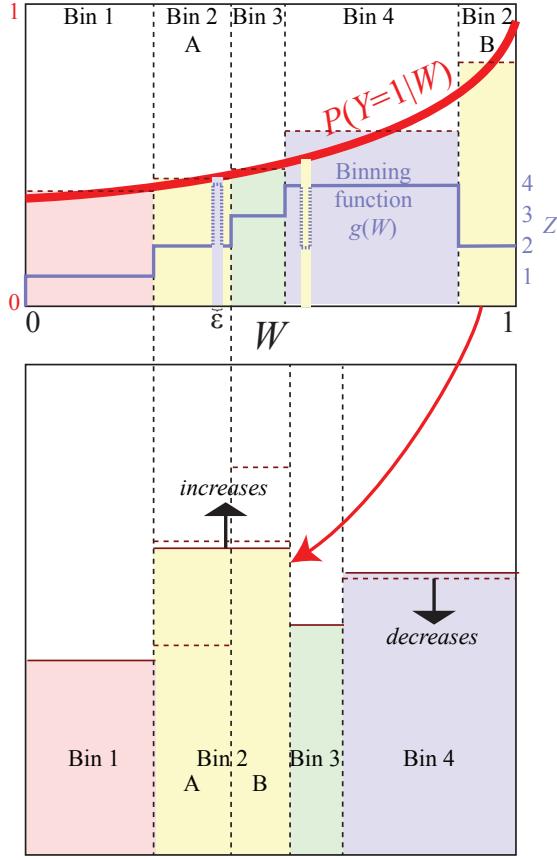


Figure 5-4: The reason that the Pareto frontier can never be reached using non-contiguous bins is that a swapping parts of them against parts of an intermediate bin can increase $I(Z, X)$ while keeping $H(Z)$ constant. In this example, the binning function g assigns two separate W -intervals (top panel) to the same bin (bin 2) as seen in the bottom panel. The shaded rectangles have widths P_i , heights p_i and areas $P_{i1} = P_i p_1$. In the upper panel, the conditional probabilities p_i are monotonically increasing because they are averages of the monotonically increasing curve $p_1(w)$.

conditional probability distribution $P(W|Y=1)$ is monotonically increasing, then all points (H_, I_*) on the Pareto frontier correspond to binning W into contiguous intervals, i.e., if*

$$I(H_*) \equiv \sup_{\{g: H[g(W)] \leq H_*\}} I[g(W), Y], \quad (5.21)$$

then there exists a set of bin boundaries $b_1 < \dots < b_{n-1}$ such that the binned variable $Z \equiv B(W, \mathbf{b}) \in \{1, \dots, M\}$ satisfies $H(Z) = H_$ and $I(Z, Y) = I_*$.*

Proof. We prove this by contradiction: we will assume that there is a point (H_*, I_*) on the

Pareto frontier to which we can come arbitrarily close with $(H(Z), I(Z, Y))$ for $Z \equiv g(X)$ for a compression function $g : \mathbb{R} \mapsto \{1, \dots, M\}$ that is not a contiguous binning function, and obtain a contradiction by using g to construct another compression function $g'(W)$ lying above the Pareto frontier, with $H[g'(W)] = H_*$ and $I[g'(W), Y] > I_*$. The joint probability distribution P_{ij} for the Z and Y is given by the Lebesgue integral

$$P_{ij} \equiv P(Z=i, Y=j) = \int f_j d\mu_i, \quad (5.22)$$

where $f_j(w)$ is the joint probability distribution for W and Y introduced earlier and μ_j is the set $\mu \equiv \{w \in [0, 1] : g(w) = j\}$, i.e., the set of w -values that are grouped together into the j^{th} large bin. We define the marginal and conditional probabilities

$$P_i \equiv P(Z=i) = P_{i1} + P_{i2}, \quad p_i \equiv P(Y=1|Z=i) = \frac{P_{i1}}{P_i}. \quad (5.23)$$

Figure 5-4 illustrates the case where the binning function g corresponds to $M = 4$ large bins, the second of which consists of two non-contiguous regions that are grouped together; the shaded rectangles in the bottom panel have width P_i , height p_i and area $P_{ij} = P_i p_i$.

According to Theorem 15 in the Appendix, we obtain the contradiction required to complete our proof (an alternative compression $Z' \equiv g'(W)$ above the Pareto frontier with $H(Z') = H_*$ and $I(Z', Y) > I_*$) if there are two different conditional probabilities $p_k \neq p_l$, and we can change g into g' so that the joint distribution P'_{ij} of Z' and Y changes in the following way:

1. Only P_{kj} and P_{lj} change,
2. both marginal distributions remain the same,
3. the new conditional probabilities p'_k and p'_l are further apart.

Figure 5-4 shows how this can be accomplished for non-contiguous binning: let k be a bin with non-contiguous support set μ_k (bin 2 in the illustrated example), let l be a bin whose support μ_l (bin 4 in the example) contains a positive measure subset $\mu_l^{\text{mid}} \subset \mu_l$ within two

parts μ_k^{left} and μ_k^{right} of μ_k , and define a new binning function $g'(w)$ that differs from $g(w)$ only by swapping a set $\mu^\epsilon \subset \mu_l^{\text{mid}}$ against a subset of either μ_k^{left} or μ_k^{right} of measure ϵ (in the illustrated example, the binning function change implementing this subset is shown with dotted lines). This swap leaves the total measure of both bins (and hence the marginal distribution P_i) unchanged, and also leaves $P(Y=1)$ unchanged. If $p_k < p_l$, we perform this swap between μ_l^{mid} and μ_k^{right} (as in the figure), and if $p_k > p_l$, we instead perform this swap between μ_l^{mid} and μ_k^{left} , in both cases guaranteeing that p_l and p_k move further apart (since $p(w)$ is monotonically increasing). This completes our proof by contradiction except for the case where $p_k = p_l$; in this case, we swap to entirely eliminate the discontiguity, and repeat our swapping procedure between other bins until we increase the entropy (again obtaining a contradiction) or end up with a fully contiguous binning (if needed, $g(w)'$ can be changed to eliminate any measure-zero subsets that ruin contiguity, since they leave the Lebesgue integral in equation (5.22) unchanged.) ■

5.2.3 Mapping the frontier

Theorem 9 implies that we can in practice find the Pareto frontier for any random variable X by searching the space of contiguous binnings of $W = w(X)$ after uniformization, binning and sorting. In practice, we can first try the 2-bin case by scanning the bin boundary $0 < b_1 < 1$, then trying the 3-bin case by trying bin boundaries $0 < b_1 < b_2 < 1$, then trying the 4-bin case, *etc.*, as illustrated in Figure 5-1. Each of these cases corresponds to a standard multi-objective optimization problem aiming to maximize the two objectives $I(Z, Y)$ and $H(Z)$. We perform this optimization numerically with the AWS algorithm of [KdW05] as described in the next section.

Although the uniformization, binning and sorting procedures are helpful in practice as well as for simplifying proofs, they are not necessary in practice. Since what we really care about is grouping into integrals containing similar conditional probabilities $p_1(w)$, not similar w -values, it is easy to see that binning horizontally after sorting is equivalent to binning vertically before sorting. In other words, we can eliminate the binning and sorting

steps if we replace “horizontal” binning $g(W) = B(W, \mathbf{b})$ by “vertical” binning

$$g(W) = B[p_1(W), \mathbf{b}], \quad (5.24)$$

where p_1 denotes the conditional probability as before.

5.3 Results

We will now test our algorithm for Pareto frontier mapping using some well-known datasets: the CIFAR-10 image database [KNH14], the MNIST database of hand-written digits [LCB10] and the Fashion-MNIST database of garment images [XRV17]. Before doing this, however, let us build intuition for how it works by testing on a much simpler toy model that is analytically solvable, where the accuracy of all approximations can be exactly determined.

5.3.1 Analytic warmup example

Let the random variables $X = (x_1, x_2) \in [0, 1]^2$ and $Y \in \{1, 2\}$ be defined by the bivariate probability distribution

$$f(X, Y) = \begin{cases} 2x_1x_2 & \text{if } Y = 1, \\ 2(1 - x_1)(1 - x_2) & \text{if } Y = 2, \end{cases} \quad (5.25)$$

which corresponds to x_1 and x_2 being two independent and identically distributed random variables with triangle distribution $f(x_i) = x_i$ if $Y = 1$, but flipped $x_i \mapsto 1 - x_i$ if $Y = 2$. This gives $H(Y) = 1$ bit and mutual information

$$I(X, Y) = 1 - \frac{\pi^2 - 4}{16 \ln 2} \approx 0.4707 \text{ bits.} \quad (5.26)$$

The compressed random variable $W = w(X) \in \mathbb{R}$ defined by equation (5.9) is thus

$$W = P(Y=1|X) = \frac{x_1x_2}{x_1x_2 + (1 - x_1)(1 - x_2)}. \quad (5.27)$$

After defining $Z \equiv B(W, \mathbf{b})$ for a vector \mathbf{b} of bin boundaries, a straightforward calculation shows that the joint probability distribution of Y and the binned variable Z is given by

$$P_{ij} \equiv P(Z=i, Y=j) = F_j(b_{i+1}) - F_j(b_i), \quad (5.28)$$

where the cumulative distribution function $F_j(w) \equiv P(W < w, Y=j)$ is given by

$$\begin{aligned} F_1(w) &= \frac{w^2 [(2w-1)(5-4w) + 2(1-w^2)\log(w^{-1}-1)]}{2(2w-1)^4}, \\ F_2(w) &= \frac{1}{2} - F_1(1-w). \end{aligned} \quad (5.29)$$

Computing $I(W, Y)$ using this probability distribution recovers exactly the same mutual information $I \approx 0.4707$ bits as in equation (5.26), as we proved in Theorem 8.

5.3.2 The Pareto frontier

Given any binning vector \mathbf{b} , we can plot a corresponding point $(H[Z], I[Z, Y])$ in Figure 5-1 by computing $I(Z, Y) = H(Z) + H(Y) - H(Z, Y)$,

$H(Z, Y) = -\sum P_{ij}\log P_{ij}$, etc., where P_{ij} is given by equation (5.28).

The figure shows 6,000 random binnings each for $M = 3, \dots, 8$ bins; as we have proven, the upper envelope of points corresponding to all possible (contiguous) binnings defines the Pareto frontier. The Pareto frontier begins with the black dot at $(0, 0)$ (the lower right corner), since $M = 1$ bin obviously destroys all information. The $M = 2$ bin case corresponds to a 1-dimensional closed curve parametrized by the single parameter b_1 that specifies the boundary between the two bins: it runs from $(0, 0)$ when $b_1 = 1$, moves to the left until $H(Z) = 1$ when $b_1 = 0.5$, and returns to $(0, 0)$ when $b_1 = 1$. The $b_1 < 0.5$ and $b_1 > 0.5$ branches are indistinguishable in Figure 5-1 because of the symmetry of our warmup problem, but in generic cases, a closed loop can be seen where only the upper part defines the Pareto frontier.

More generally, we see that the set of all binnings into $M > 2$ bins maps the vector \mathbf{b} of $M - 1$ bin boundaries into a contiguous region in Figure 5-1. The inferior white region

region below can also be reached if we use non-contiguous binnings.

The Pareto Frontier is seen to resemble the top of a circus tent, with convex segments separated by “corners” where the derivative vanishes, corresponding to a change in the number of bins. We can understand the origin of these corners by considering what happens when adding a new bin of infinitesimal size ϵ . As long as $p_i(w)$ is continuous, this changes all probabilities P_{ij} by amounts $\delta P_{ij} = \mathcal{O}(\epsilon)$, and the probabilities corresponding to the new bin (which used to vanish) will now be $\mathcal{O}(\epsilon)$. The function $\epsilon \log \epsilon$ has infinite derivative at $\epsilon = 0$, blowing up as $\mathcal{O}(\log \epsilon)$, which implies that the entropy increase $\delta H(Z) = \mathcal{O}(-\log \epsilon)$. In contrast, a straightforward calculation shows that all $\log \epsilon$ -terms cancel when computing the mutual information, which changes only by $\delta I(Z, Y) = \mathcal{O}(\epsilon)$. As we birth a new bin and move leftward from one of the black dots in Figure 5-1, the initial slope of the Pareto frontier is thus

$$\lim_{\epsilon \rightarrow 0} \frac{\delta I(Z, Y)}{\delta H(Z)} = 0. \quad (5.30)$$

In other words, the Pareto frontier starts out *horizontally* to the left of each of its corners in Figure 5-1. Indeed, the corners are “soft” in the sense that the derivative of the Pareto Frontier is continuous and vanishes at the corners: for a given number of bins, $I(X, Z)$ by definition takes its global maximum at the corresponding corner, so the derivative $\partial I(Z, Y)/\partial H(Z)$ vanishes also as we approach the corner from the right.¹⁰

Our theorems imply that in the $M \rightarrow \infty$ limit of infinitely many bins, successive corners become gradually less pronounced (with ever smaller derivative discontinuities), because the left asymptote of the Pareto frontier simply approaches the horizontal line $I_* = I(Y, Z)$.

Approximating $w(X)$

For our toy example, we knew the conditional probability distribution $P(Y|X)$ and could therefore compute $W = w(X) = P(Y=1|X)$ exactly. For practical examples where this is not the case, we can instead train a neural network to implement a function $\hat{w}(X)$ that

¹⁰The first corner (the transition from 2 to 3 bins) can nonetheless look fairly sharp because the 2-bin curve turns around rather abruptly, and the right derivative does not vanish in the limit where a symmetry causes the upper and lower parts of the 2-bin loop to coincide.

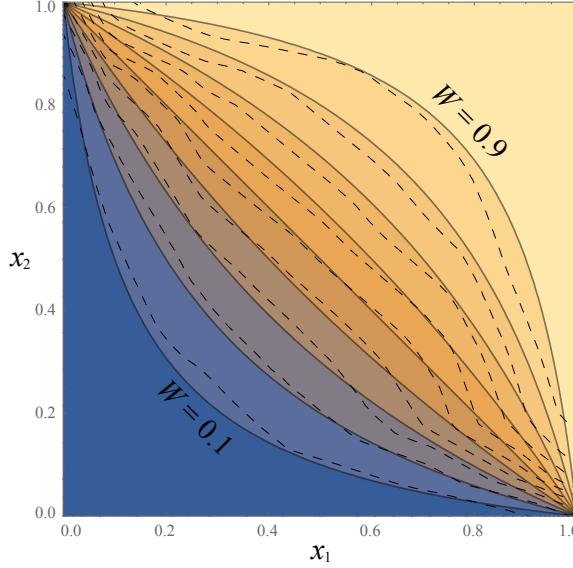


Figure 5-5: Contour plot of the function $W(x_1, x_2)$ computed both exactly using equation 5.27 (solid curves) and approximately using a neural network (dashed curves).

approximates $P(Y=1|X)$. For our toy example, we train a fully connected feedforward neural network to predict Y from X using cross-entropy loss; it has 2 hidden layers, each with 256 neurons with ReLU activation, and a final linear layer with softmax activation, whose first neuron defines $\hat{w}(X)$. As illustrated in Figure 5-5, the network prediction for the conditional probability $\hat{w}(X) \equiv P(Y=1)$ is fairly accurate, but slightly over-confident, tending to err on the side of predicting more extreme probabilities (further from 0.5). The average KL-divergence between the predicted and actual conditional probability distribution $P(Y|X)$ is about 0.004, which causes negligible loss of information about Y .

Approximating $f_1(W)$

For practical examples where the conditional joint probability distribution $P(W, Y)$ cannot be computed analytically, we need to estimate it from the observed distribution of W -values output by the neural network. For our examples, we do this by fitting each probability distribution by a beta-distribution times the exponential of a polynomial of degree d :

$$f(w, \mathbf{a}) \equiv \exp \left[\sum_{k=0}^d a_k w^k \right] x^{a_{d+1}} (1-x)^{a_{d+2}}, \quad (5.31)$$

Experiment	Y	a_0	a_1	a_2	a_3	a_4	a_5	a_6
Analytic	1	0.0668	-4.7685	16.8993	-25.0849	13.758	0.5797	-0.2700
	2	0.4841	-5.0106	5.7863	-1.5697	-1.7180	-0.3313	-0.0030
Fashion-MNIST	Pullover	0.2878	-12.9596	44.9217	-68.0105	37.3126	0.3547	-0.2838
	Shirt	1.0821	-23.8350	81.6655	-112.2720	53.9602	-0.4068	0.4552
CIFAR-10	Cat	0.9230	0.2165	0.0859	6.0013	-1.0037	0.8499	
	Dog	0.8970	0.6795	0.0511	0.6838	-1.0138	0.9061	
MNIST	One	3.1188	-65.224	231.4	-320.054	150.779	1.1226	-0.6856
	Seven	-1.0325	-47.5411	189.895	-269.28	127.363	-0.8219	0.1284

Table 5.2: Fits to the conditional probability distributions $P(W|Y)$ for our experiments, in terms of the parameters a_i defined by equation (5.31).

where the coefficient a_0 is fixed by the normalization requirement $\int_0^1 f(w, \mathbf{a}) dw = 1$. We use this simple parametrization because it can fit any smooth distribution arbitrarily well for sufficiently large d , and provides accurate fits for the probability distributions in our examples using quite modest d ; for example, $d = 3$ gives $d_{KL}[f_1(w), f(w, \mathbf{a})] \approx 0.002$ for

$$\begin{aligned} \mathbf{a} &\equiv \underset{\mathbf{a}'}{\operatorname{argmin}} d_{KL}[f_1(w), f(w, \mathbf{a}')] \\ &= (-1.010, 2.319, -5.579, 4.887, 0.308, -0.307), \end{aligned} \quad (5.32)$$

which causes rather negligible loss of information about Y . For our examples below where we do not know the exact distribution $f_1(w)$ and merely have samples W_i drawn from it, one for each element of the data set, we instead perform the fitting by the standard technique of minimizing the cross entropy loss, *i.e.*,

$$\mathbf{a} \equiv \underset{\mathbf{a}'}{\operatorname{argmin}} - \sum_{k=1}^n \log f(W_k, \mathbf{a}'). \quad (5.33)$$

Table 5.2 lists the fitting coefficients used, and Figure 5-6 illustrates the fitting accuracy.

5.3.3 MNIST, Fashion-MNIST and CIFAR-10

The MNIST database consists of 28x28 pixel greyscale images of handwritten digits: 60,000 training images and 10,000 testing images [LCB10]. We use the digits 1 and 7, since they are the two that are most frequently confused, relabeled as $Y = 1$ (ones) and $Y = 2$ (sevens).

To increase difficulty, we inject 30% of pixel noise, i.e., randomly flip each pixel with 30% probability (see examples in Figure 5-2). For easy comparison with the other cases, we use the same number of samples for each class.

The Fashion-MNIST database has the exact same format (60,000 + 10,000 28x28 pixel greyscale images), depicting not digits but 10 classes of clothing [XRV17]. Here we again use the two most easily confused classes: pullovers ($Y = 1$) and shirts ($Y = 2$); see Figure 5-2 for examples.

The architecture of the neural network classifier we train on the above two datasets is adapted from here¹¹: two convolutional layers (kernel size 5, stride 1, ReLU activation) with 20 and 50 features, respectively, each of which is followed by max-pooling with kernel size 2. This is followed by a fully connected layer with 500 ReLU neurons and finally a softmax layer that produces the predicted probabilities for the two classes. After training, we apply the trained model to the test set to obtain $W_i = P(Y|X_i)$ for each dataset.

CIFAR-10 [KH09] is one of the most widely used datasets for machine learning research, and contains 60,000 32×32 color images in 10 different classes. We use only the cat ($Y = 1$) and dog ($Y = 2$) classes, which are the two that are empirically hardest to discriminate; see Figure 5-2 for examples. We use a ResNet18 architecture¹² [HZRS16a]. We train with a learning rate of 0.01 for the first 150 epochs, 0.001 for the next 100, and 0.0001 for the final 100 epochs; we keep all other settings the same as in the original repository.

Figure 5-6 shows observed cumulative distribution functions $F_i(w)$ (solid curves) for the $W_i = P(Y = 1|X_i)$ generated by the neural network classifiers, together with our above-mentioned analytic fits (dashed curves).¹³ Figure 5-7 shows the corresponding conditional probability curves $P(Y = 1|W)$ after remapping W to have a uniform distribution as described above. Figure 5-6 shows that the original W -distributions are strongly peaked around $W \approx 0$ and $W \approx 1$, so this remapping stretches the W -axis so as to shift probability toward more central values.

¹¹We use the neural network architecture from github.com/pytorch/examples/blob/master/mnist/main.py; the only difference in architecture is that our output number of neurons is 2 rather than 10.

¹²The architecture is adapted from github.com/kuangliu/pytorch-cifar, for which we use its ResNet18 model; the only difference in architecture is that we use 2 rather than 10 output neurons.

¹³In the case of CIFAR-10, the observed distribution $f(w)$ was so extremely peaked near the endpoints that

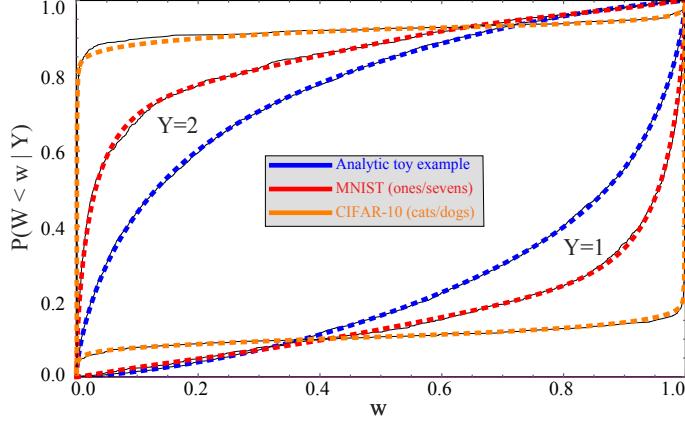


Figure 5-6: Cumulative distributions $F_i(w) \equiv P(W < w | Y=i)$ are shown for the analytic (blue/dark grey), Fashion-MNIST (red/grey) and CIFAR-10 (orange/light grey) examples. Solid curves show the observed cumulative histograms of W from the neural network, and dashed curves show the fits defined by equation (5.31) and Table 5.2.

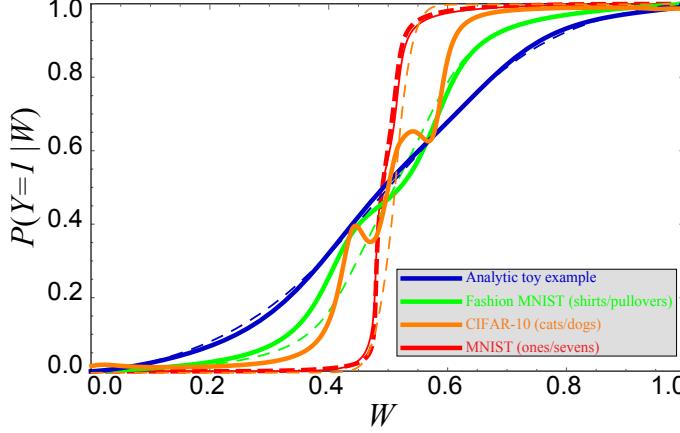


Figure 5-7: The solid curves show the actual conditional probability $P(Y=1|W)$ for CIFAR-10 (where the labels $Y=1$ and 2 correspond to “cat” and “dog”) and MNIST with 20% label noise (where the labels $Y=1$ and 2 correspond to “1” and “7”), respectively. The color-matched dashed curves show the conditional probabilities predicted by the neural network; the reason that they are not diagonal lines $P(Y=1|W) = W$ is that W has been reparametrized to have a uniform distribution. If the neural network classifiers were optimal, then solid and dashed curves would coincide.

we replaced equation (5.31) by the more accurate fit

$$f(w) \equiv F'(w), \quad (5.34)$$

$$F(w) \equiv \begin{cases} \mathbf{a}_0^A F_*[w, \mathbf{a}^A] & \text{if } w < 1/2, \\ 1 - (1 - \mathbf{a}_0^A) F_*[1 - w, \mathbf{a}^B] & \text{otherwise,} \end{cases} \quad (5.35)$$

$$F_*(x) \equiv G\left[\frac{(2x)^{a_1}}{2}\right], \quad (5.36)$$

$$G(x) \equiv \left[\left(\frac{x}{a_2}\right)^{a_3 a_4} + (a_5 + a_6 x)^{a_4}\right]^{1/a_4}, \quad (5.37)$$

$$a_6 \equiv 2 \left[(1 - (2a_2)^{-a_3 a_4})^{1/a_4} - a_5 \right], \quad (5.38)$$

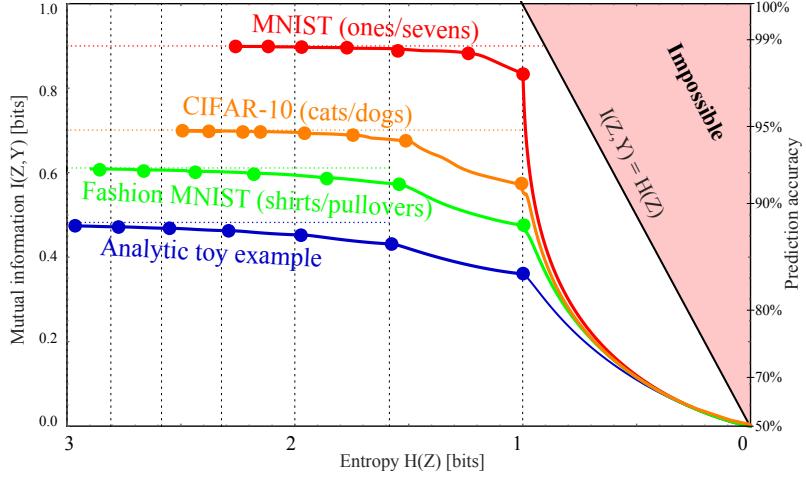


Figure 5-8: The Pareto frontier for compressed versions $Z = g(X)$ of our four datasets X , showing the maximum attainable class information $I(Z, Y)$ for a given entropy $H(Z)$. The “corners” (dots) correspond to the maximum $I(Z, Y)$ attainable when binning the likelihood W into a given number of bins (2, 3, ..., 8 from right to left). The horizontal dotted lines show the maximum available information $I(X, Y)$ for each case, reflecting that there is simply less to learn in some examples than in others.

The final result of our calculations is shown in Figure 5-8: the Pareto frontiers for our four datasets, computed using our method.

5.3.4 Interpretation of our results

To build intuition for our results, let us consider our CIFAR-10 example of images X depicting cats ($Y=1$) and dogs ($Y=2$) as in Figure 5-2 and ask what aspects $Z = g(X)$ of an image X capture the most information about the species Y . Above, we estimated that $I(X, Y) \approx 0.69692$ bits, so what Z captures the largest fraction of this information for a fixed entropy? Given a good neural network classifier, a natural guess might be the single bit Z containing its best guess, say “it’s probably a cat”. This corresponds to defining $Z = 1$ if $P(Y=1|X) > 0.5$, $Z = 2$ otherwise, and gives the joint distribution $P_{ij} \equiv P(Y=i, Z=j)$

$$\mathbf{P} = \begin{pmatrix} 0.454555 & 0.045445 \\ 0.042725 & 0.457275 \end{pmatrix}$$

where the parameters vectors \mathbf{a}^A and \mathbf{a}^B are given in Table 5.2 for both cats and dogs. For the cat case, this fit gives not $f(w)$ but $f(1 - w)$. Note that $F_*(0) = 0$, $F_*(1/2) = 1$.

corresponding to $(Z, Y) \approx 0.56971$ bits. But our results show that we can improve things in two separate ways.

First of all, if we only want to store one bit Z , then we can do better, corresponding to the first “corner” in Figure 5-8: moving the likelihood cutoff from 0.5 to 0.51, *i.e.*, redefining $Z = 1$ if $P(Y|X) > 0.51$, increases the mutual information to $I(Z, Y) \approx 0.56974$ bits.

More importantly, we are still falling far short of the 0.69692 bits of information we had without data compression, capturing only 88% of the available species information. Our Theorem 8 showed that we can retain *all* this information if we instead define Z as the cat probability itself: $Z \equiv W \equiv P(Y|X)$. For example, a given image might be compressed not into “*It’s probably a cat*” but into “*I’m 94.2477796% sure it’s a cat*”. However, it is clearly impractical to report the infinitely many decimals required to retain all the species information, which would make $H(Z)$ infinite. Our results can be loosely speaking interpreted as the optimal way to round Z , so that the information $H(Z)$ required to store it becomes finite. We found that simply rounding to a fixed number of decimals is suboptimal; for example, if we pick 2 decimals and say “*I’m 94.25% sure it’s a cat*”, then we have effectively binned the probability W into 10,000 bins of equal size, even though we can often do much better with bins of unequal size, as illustrated in the bottom panel of Figure 5-1. Moreover, when the probability W is approximated by a neural network, we found that what should be optimally binned is not W but the conditional probability $P(Y=1|W)$ illustrated in Figure 5-7 (“vertical binning”).

It is convenient to interpret our Pareto-optimal data compression $X \mapsto Z$ as *clustering*, *i.e.*, as a method of grouping our images or other data X_i into clusters based on what information they contain about Y . For example, Figure 5-2 illustrates CIFAR-10 images clustered by their degree of “cattiness” into 5 groups $Z = 1, \dots, 5$ that might be nicknamed “1.9% cat”, “11.8% cat”, “31.4% cat”, “68.7% cat” and “96.7% cat”. This gives the joint distribution $P_{ij} \equiv P(Y=i, Z=j)$ where

$$\mathbf{P} = \begin{pmatrix} 0.350685 & 0.053337 & 0.054679 & 0.034542 & 0.006756 \\ 0.007794 & 0.006618 & 0.032516 & 0.069236 & 0.383836 \end{pmatrix}$$

and gives $I(Z, Y) \approx 0.6882$, thus increasing the fraction of species information retained from 82% to 99%.

This is a striking result: we can group the images into merely five groups and discard all information about all images except which group they are in, yet retain 99% of the information we cared about. Such grouping may be helpful in many contexts. For example, given a large sample of labeled medical images of potential tumors, they can be used to define say five optimal clusters, after which future images can be classified into five degrees of cancer risk that collectively retain virtually all the malignancy information in the original images.

Given that the Pareto Frontier is continuous and corresponds to an infinite family of possible clusterings, which one is most useful in practice? Just as in more general multi-objective optimization problems, the most interesting points on the frontier are arguably its “corners”, indicated by dots in Figure 5-8, where we do notably well on both criteria. This point was also made in the important paper [SS19] in the context of the DIB-frontier discussed below. We see that the parts of the frontier between corners tend to be convex and thus rather unappealing, since any weighted average of $-H(Z)$ and $I(Z, Y)$ will be maximized at a corner. Our results show that these corners can conveniently be computed without numerically tedious multiobjective optimization, by simply maximizing the mutual information $I(Z, Y)$ for $m = 2, 3, 4, \dots$ bins. The first corner, at $H(Z) = 1\text{bit}$, corresponds to the learnability phase transition for DIB, *i.e.*, the largest β for which DIB is able to learn a non-trivial representation. In contrast to the IB learnability phase transition [WFCT19a, WFCT19b] where $I(Z, Y)$ increases continuously from 0, here the $I(Y; Z)$ has a jump from 0 to a positive value, due to the non-concave nature of the Pareto frontier.

Moreover, all the examples in Figure 5-8 are seen to get quite close to the $m \rightarrow \infty$ asymptote $I(Z, Y) \rightarrow I(X, Y)$ for $m \gtrsim 5$, so the most interesting points on the Pareto frontier are simply the first handful of corners. For these examples, we also see that the greater the mutual information is, the fewer bins are needed to capture most of it.

An alternative way of interpreting the Pareto plane in Figure 5-8 is as a tradeoff between two

evils:

$$\textbf{Information bloat: } H(Z|Y) \equiv H(Z) - I(Z, Y) \geq 0,$$

$$\textbf{Information loss: } \Delta I \equiv I(X, Y) - I(Z, Y) \geq 0.$$

What we are calling the *information bloat* has also been called “causal waste” [TGM⁺18]. It is simply the conditional entropy of Z given Y , and represents the excess bits we need to store in order to retain the desired information about Y . Geometrically, it is the horizontal distance to the impossible region to the right in Figure 5-8, and we see that for MNIST, it takes local minima at the corners for both 1 and 2 bins. The *information loss* is simply the information discarded by our lossy compression of X . Geometrically, it is the vertical distance to the impossible region at the top of Figure 5-1 (and, in Figure 5-8, it is the vertical distance to the corresponding dotted horizontal line). As we move from corner to corner adding more bins, we typically reduce the information loss at the cost of increased information bloat. For the examples in Figure 5-8, we see that going beyond a handful of bins essentially just adds bloat without significantly reducing the information loss.

5.3.5 Real-world issues

We just discussed how lossy compression is a tradeoff between information bloat and information loss. Let us now elaborate on the latter, for the real-world situation where $W \equiv P(Y=1|X)$ is approximated by a neural network.

If the neural network learns to become perfect, then the function w that it implements will be such that $W \equiv w(X)$ satisfies $P(Y = 1|W) = W$, which corresponds to the dashed curves in Figure 5-7 being identical to the solid curves. Although we see that this is close to being the case for the analytic and MNIST examples, the neural networks are further from optimal for Fashion-MNIST and CIFAR-10. The figure illustrates that the general trend is for these neural networks to overfit and therefore be overconfident, predicting probabilities that are too extreme.

This fact that $P(Y=1|W) \neq W$ probably indicates that our Fashion-MNIST and CIFAR-10

classifiers $W = w(X)$ destroy information about X , but it does not prove this, because if we had a perfect lossless classifier $W \equiv w(X)$ satisfying $P(Y=1|W) = W$, then we could define an overconfident lossless classifier by an invertible (and hence information-preserving) reparameterization such as $W' \equiv W^2$ that violates the condition $P(Y=1|W') = W'$.

So how much information does X contain about Y ? One way to lower-bound $I(X; Y)$ uses the classification accuracy: if we have a classification problem where $P(Y=1) = 1/2$ and compress X into a single classification bit Z (corresponding to a binning of W into two bins), then we can write the joint probability distribution for Y and the guessed class Z as

$$P = \begin{pmatrix} \frac{1}{2} - \epsilon_1 & \epsilon_1 \\ \epsilon_2 & \frac{1}{2} - \epsilon_2 \end{pmatrix}.$$

For a fixed total error rate $\epsilon \equiv \epsilon_1 + \epsilon_2$, Fano's Inequality implies that the mutual information takes a minimum

$$I(Z, Y) = 1 + \epsilon \log \epsilon + (1 - \epsilon) \log(1 - \epsilon) \quad (5.39)$$

when $\epsilon_1 = \epsilon_2 = \epsilon/2$, so if we can train a classifier that gives an error rate ϵ , then the right-hand-side of equation (5.39) places a lower bound on the mutual information $I(X, Y)$. The prediction accuracy $1 - \epsilon$ is shown for reference on the right side of Figure 5-8. Note that getting close to one bit of mutual information requires extremely high accuracy; for example, 99% prediction accuracy corresponds to only 0.92 bits of mutual information.

We can obtain a stronger estimated lower bound on $I(X, Y)$ from the cross-entropy loss function \mathcal{L} used to train our classifiers:

$$\langle \mathcal{L} \rangle = -\langle \log P(Y=Y_i|X=X_i) \rangle = H(Y|X) + d_{\text{KL}}, \quad (5.40)$$

where $d_{\text{KL}} \geq 0$ denotes the average KL-divergence between true and predicted conditional probability distributions, and $\langle \cdot \rangle$ denotes ensemble averaging over data points, which implies

that

$$\begin{aligned} I(X, Y) &= H(Y) - H(Y|X) = H(Y) - \langle \mathcal{L} \rangle - d_{\text{KL}} \\ &\geq H(Y) - \langle \mathcal{L} \rangle. \end{aligned} \quad (5.41)$$

If $P(Y=1|W) \neq W$ as we discussed above, then d_{KL} and hence the loss can be further reduced by recalibrating W as we have done, which increases the information bound from equation (5.41) up to the value computed directly from the observed joint distribution $P(W, Y)$.

Unfortunately, without knowing the true probability $p(Y|X)$, there is no rigorous and practically useful *upper* bound on the mutual information other than the trivial inequality $I(X, Y) < H(Y) = 1$ bit, as the following simple counterexample shows: suppose our images X are encrypted with some encryption algorithm that is extremely time-consuming to crack, rendering the images for all practical purposes indistinguishable from random noise. Then any reasonable neural network will produce a useless classifier giving $I(W, Y) \approx 0$ even though the true mutual information $I(X, Y)$ could be as large as one bit. In other words, we generally cannot know the true information loss caused by compressing $X \mapsto W$, so the best we can do in practice is to pick a corner reasonably close to the upper asymptote in Figure 5-8.

5.3.6 Performance compared with Blahut-Arimoto method

The most commonly used technique to date for finding the Pareto frontier is the Blahut-Arimoto (BA) method [Bla72, Ari72] applied to the DIB objective of equation (5.3) as described in [SS17a]. Figure 5-9 shows the BA method implemented as in [SS19], applied to our above-mentioned analytic toy example, after binning using 2,000 equispaced W -bins and $Z \in \{1, \dots, 8\}$, scanning the β -parameter from equation (5.3) from 10^{-10} to 1 in 20,000 logarithmically equispaced steps. Our method is seen to improve on the BA method in two ways. First, our method finds the entire continuous frontier, whereas the BA method finds only discrete disconnected points. This is because the BA-method tries to maximize the

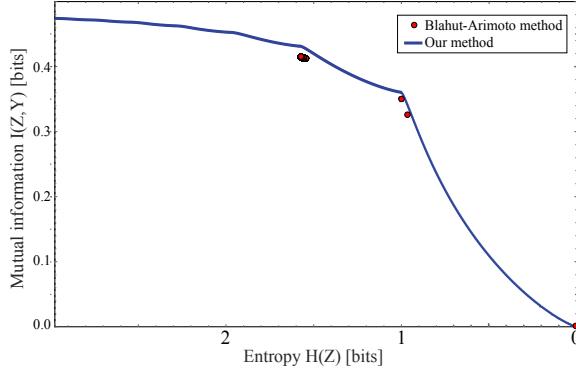


Figure 5-9: The Pareto frontier our analytic example is computed exactly with our method (solid curve) and approximately with the Blahut-Arimoto method (dots).

the DIB-objective from equation (5.3) and thus cannot discover points where the Pareto frontier is convex as discussed above. Second, our method finds the exact frontier, whereas the BA-method finds only approximations, which are seen to all lie below the true frontier.

5.4 Conclusions

We have presented a method for mapping out the Pareto frontier for classification tasks (as in Figure 5-8), reflecting the tradeoff between retained entropy and class information. We first showed how a random variable X (an image, say) drawn from a class $Y \in \{1, \dots, n\}$ can be distilled into a vector $W = f(X) \in \mathbb{R}^{n-1}$ losslessly, so that $I(W, Y) = I(X, Y)$. For the $n = 2$ case of binary classification, we then showed how the Pareto frontier is swept out by a one-parameter family of binnings of W into a discrete variable $Z = g_\beta(W) \in \{1, \dots, m_\beta\}$ that corresponds to binning W into $m_\beta = 2, 3, \dots$ bins, such that $I(Z, Y)$ is maximized for each fixed entropy $H(Z)$. Our method efficiently finds the exact Pareto frontier, significantly outperforming the Blahut-Arimoto (BA) method [Bla72, Ari72].

5.4.1 Relation to Information Bottleneck

As mentioned in the introduction, the Discrete Information Bottleneck (DIB) method [SS17a] maximizes a linear combination $I(Z, Y) - \beta H(Z)$ of the two axes in Figure 5-8.

We have presented a method solving a generalization of the DIB problem. The generalization lies in switching the objective from equation (5.3) to equation (5.1), which has the advantage of discovering the full Pareto frontier in Figure 5-8 instead of merely the corners and concave parts (as mentioned, the DIB objective cannot discover convex parts of the frontier). The solution lies in our proof that the frontier is spanned by binnings of the likelihood into 2, 3, 4, ... bins, which enables it to be computed more efficiently than with the iterative/variational method of [SS17a].

The popular original Information Bottleneck (IB) method [TPB00] generalizes DIB by allowing the compression function $g(X)$ to be non-deterministic, thus adding noise that is independent of X . Starting with a Pareto-optimal $Z \equiv g(X)$ and adding such noise will simply shift us straight to the left in Figure 5-8, away from the frontier (which is by definition monotonically decreasing) and into the Pareto-suboptimal region in the $I(Y; Z)$ vs. $H(Z)$ plane. As shown in [SS17a], IB-compressions tend to altogether avoid the rightmost part of Figure 5-8, with an entropy $H(Z)$ that never drops below some fixed value independent of β .

5.4.2 Relation to phase transitions in DIB learning

Recent work has revealed interesting phase transitions that occur during information bottleneck learning [CGTW05, SS17a, WFCT19a, WFCT19b], as well as phase transitions in other objectives, e.g. β -VAE [RV18], infoDropout [AS18a]. Specifically, when the β -parameter that controls the tradeoff between information retention and model simplicity is continuously adjusted, the resulting point in the IB-plane can sometimes “get stuck” or make discontinuous jumps. For the DIB case, our results provide an intuitive understanding of these phase transitions in terms of the geometry of the Pareto frontier.

Let us consider Figure 5-1 as an example. The DIB maximization of $I(Z, Y) - \beta H(Z)$ geometrically corresponds to finding a tangent line of the Pareto frontier of slope $-\beta$.

If the Pareto frontier $I_*(H)$ were everywhere continuous and concave, so that $I''_*(H) < 0$, then its slope would range from some steepest value $-\beta_*$ at the right endpoint $H = 0$ and

continuously flatten out as we move leftward, asymptotically approaching zero slope as $H \rightarrow \infty$. The learnability phase transition studied in [WFCT19a, WFCT19b] would then occur when $\beta = \beta_*$: for $\beta < \beta_*$, the DIB method learns nothing, *e.g.*, discovers as optimal the point $(H, I) = (0, 0)$ where Z retains no information whatsoever about Y . As $\beta \leq \beta_*$ is continuously reduced, the DIB-discovered point would then continuously move up and to the left along the Pareto frontier.

This was for the case of an everywhere concave frontier, but Figures 5-1 and 5-8 show that actual Pareto frontiers need *not* be concave — indeed, none of the frontiers that we have computed are concave. Instead, they are seen to consist of long convex segments joint together by short concave pieces near the “corners”. This means that as β is continuously increased, the DIB solution exhibits first-order phase transitions, making discontinuous jumps from corner to corner at certain critical β -values; these phase transitions correspond to increasing the number of clusters into which the data X is grouped.

5.4.3 Outlook

Our results suggest a number of opportunities for further work, ranging from information theory to machine learning, neuroscience and physics.

As to information theory, it will be interesting to try to generalize our method from binary classification into classification into more than two classes. Also, one can ask if there is a way of pushing the general information distillation problem all the way to bits. It is easy to show that a discrete random variable $Z \in \{1, \dots, m\}$ can always be encoded as $m - 1$ independent random bits (Bernoulli variables) $B_1, \dots, B_{m-1} \in \{0, 1\}$, defined by¹⁴

$$P(B_k=1) = P(Z=k+1)/P(Z \leq k+1), \quad (5.42)$$

although this generically requires some information bloat. So in the spirit of the introduction,

¹⁴The mapping z from bit strings \mathbf{B} to integers $Z \equiv z(\mathbf{B})$ is defined so that $z(\mathbf{B})$ is the position of the last bit that equals one when \mathbf{B} is preceded by a one. For example, for $m = 4$, the mapping from length-3 bit strings $\mathbf{B} \in \{0, 1\}^3$ to integers $Z \in \{1, \dots, 4\}$ is $z(001) = z(011) = z(101) = z(111) = 4$, $z(010) = z(110) = 3$, $z(100) = 2$, $z(000) = 1$.

is there some useful way of generalizing PCA, autoencoders, CCA and/or the method we have presented so that the quantities Z_i and Z'_i in Table 5.1 are not real numbers but bits?

As to neural networks, it is interesting to explore novel classifier architectures that reduce the overfitting and resulting overconfidence revealed by Figure 5-7, as this might significantly increase the amount of information we can distill into our compressed data. It is important not to complacently declare victory just because classification accuracy is high; as mentioned, even 99% binary classification accuracy can waste 8% of the information.

As to neuroscience, our discovery of optimal “corner” binnings begs the question of whether evolution may have implemented such categorization in brains. For example, if some binary variable Y that can be inferred from visual imagery is evolutionarily important for a given species (say, whether potential food items are edible), might our method help predict how many distinct colors m their brains have evolved to classify hues into? In this example, X might be a triplet of real numbers corresponding to light intensity recorded by three types of retinal photoreceptors, and the integer Z might end up corresponding to some definitions of yellow, orange, *etc.* A similar question can be asked for other cases where brains define finite numbers of categories, for example categories defined by distinct words.

As to physics, it has been known even since the introduction of Maxwell’s Demon that a physical system can use information about its environment to extract work from it. If we view an evolved life form as an intelligent agent seeking to perform such work extraction, then it faces a tradeoff between retaining too little relevant information (consequently extracting less work) and retaining too much (wasting energy on information processing and storage). Susanne Still recently proved the remarkable physics result [Sti17] that the lossy data compression optimizing such work extraction efficiency is precisely that prescribed by the above-mentioned Information Bottleneck method [TPB00]. As she puts it, an intelligent data representation strategy emerges from the optimization of a fundamental physical limit to information processing. This derivation made minimal and reasonable seeming assumptions about the physical system, but did not include an energy cost for information encoding. We conjecture that this can be done such that an extra Shannon coding term proportional to $H(Z)$ gets added to the loss function, which means that when this term dominates, the

generalized Still criterion would instead prefer the Deterministic Information Bottleneck or one of our Pareto-optimal data compressions.

Although noise-adding IB-style data compression may turn out to be commonplace in many biological settings, it is striking that the types of data compression we typically associate with human perception intelligence appears more deterministic, in the spirit of DIB and our work. For example, when we compress visual input into “this is a probably a cat”, we do not typically add noise by deliberately flipping our memory to “this is probably a dog”. Similarly, the popular jpeg image compression algorithm dramatically reduces image sizes while retaining essentially all information that we humans find relevant, and does so deterministically, without adding noise.

It is striking that simple information-theoretical principles such as IB, DIB and Pareto-optimality appear relevant across the spectrum of known intelligence, ranging from extremely simple physical systems as in Still’s work all the way up to high-level human perception and cognition. This motivates further work on the exciting quest for a deeper understanding of Pareto-optimal data compression and its relation to neuroscience and physics.

Chapter 6

Learning causal relations from observations

Identifying¹² the underlying directional/causal relations from observational time series with nonlinear interactions and complex relational structures is key to a wide range of applications, yet remains a hard problem. In this chapter, we introduce a novel minimum predictive information regularization method to infer directional relations from time series, allowing deep learning models to discover nonlinear relations. Our method substantially outperforms other methods for learning nonlinear relations in synthetic datasets, and discovers the directional relations in a video game environment and a heart-rate vs. breath-rate dataset.

6.1 Introduction and Related Work

Imagine a dataset with tens or hundreds of observational time series. There may exist interesting directional relations between the time series which we want to uncover, but their relation graph may be complicated, and the relation may be nonlinear as we do not know its functional form. How can we discover the underlying relations of those challenging

¹²The paper “Nonlinear Causal Discovery with Minimum Predictive Information Regularization” was presented at ICML 2019 Time Series workshop, and awarded the *Best Poster Award*. Authors: Tailin Wu, Thomas Breuel, Michael Skuhersky and Jan Kautzin. arXiv:2001.01885.

²The code for the methods and experiments is open-sourced at github.com/tailintalent/causal.

scenarios in an efficient way, or at least identify candidate relations that are worth further investigation by a researcher? Problems of this type are omnipresent and important in a variety of scientific endeavors and applications, e.g., gene regulatory networks [LALR09], neuroscience [NCB08, SBB15], economics [Gra69, SW89] and finance [HJ94, GHY00].

To address this question, the field of causal learning has proposed a large class of methods to discover or quantify causal relations. These methods have certain limitations in regards to capability of handling nonlinearity, and/or scalability and efficiency to large numbers of time series. Pearl [Pea02, Pea09, P⁺09] defines causality in terms of intervention and structural dependence, under the structural equation models (SEM). However, in our problem, where only observational time series is available, Pearl’s definition may not be applicable. In his seminal work, Granger [Gra69, Gra80] defines causality via prediction: if the prediction of the future Y via a linear model can be improved by including the information of X , then X causes Y in the Granger sense. The original Granger causality is limited to linear causal models. Although later works also extend Granger causality to kernel methods [AMS04, MPS08b, MPS08a, SQL12], they may still be insufficient to model and discover the nonlinear causal relations in real data. On the other hand, the causal measures of transfer entropy [Sch00] and causal influence [JBGW⁺13] are in theory able to handle any nonlinearity. However, both measures require density estimation of the joint distribution for the full N time series (N is the number of time series), which is difficult and data-hungry when N is large. Constraint-based methods [SGS⁺00, HD13, Pea02, SGS⁺00] require repetitive conditional independence tests, where the number of tests will grow large when N is large and the underlying causal graph is dense. Score-based methods search for the structure that yields the optimal score w.r.t. the data, generally using greedy search methods, for example GES [Chi02], rankGES [NHM⁺18] and GIES [HB12]. This in general requires $\Theta(N^2)$ steps, and the number of neighboring states may grow very large at each step. Another closely related field is sparse learning/feature selection methods. Some important classes are Lasso [Tib96] and elastic net [ZH05], which are effective but subject to the limitations of linear models. For nonlinear models, although L1 and group L1 regularization [MVDGB08, SCHU17, TCF⁺18] can induce sparsity in the model parameters, they are model and input dependent.

To handle the nonlinear relations in time series, a promising tool is neural nets. Not only are neural nets universal function approximators [Hor91], a deep neural net also provides exponentially large expressive power [RT18], making it particularly suitable for modeling the unknown nonlinear relations in time series. Recently there has been an increasing amount of work on learning the dynamic models of interacting systems [BPL⁺16, CUTT16, GVW⁺16, WZW⁺17a, Hos17, vSCGS18]. However, their main focus is to make better predictions, using implicit interaction models (e.g. using fully connected graph networks). In this paper, we are mainly interested in discovering the underlying directional relations in an explicit form, utilizing the expressive power of neural nets.

To discover nonlinear directional relations from potentially large number of time series in an efficient way, the contribution of our work is as follows:

- We introduce a novel relational learning with Minimum Predictive Information Regularization (MPIR) method for exploratory discovery of nonlinear directional relations from observational time series. It is based on minimizing a mutual information-regularized risk with learnable input noise of a prediction model, which allows function approximators such as neural nets to learn nonlinear relations, combining the benefits of the Granger causality paradigm with deep learning models. At the minimization of the objective, the minimum predictive information term quantifies the directional predictive strength between each pair of time series given other time series. For discovering the directional relations among N time series, it only has to learn N models, and does not require density estimation for the joint N time series.
- We prove that the minimum predictive information is able to differentiate dependence or independence between pairs of time series, which allows for statistical test. Moreover, we prove that the minimum predictive information is invariant to the scaling of input and reparameterization of the model. We further provide intuition that under certain conditions, our method is likely to discover direct relations instead of indirect associations.
- We demonstrate on nonlinear synthetic datasets that our method outperforms other methods in discovering true causal relations with larger N , and discovers the directional relations in video game environment and real-world heart-rate vs. breath-rate datasets.

6.2 Method

6.2.1 Problem setup

We consider N time series $x^{(1)}, x^{(2)}, \dots, x^{(N)}$, where each time series $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \dots, x_t^{(i)}, \dots)$ and each $x_t^{(i)} \in \mathbf{R}^M$ is an M -dimensional vector. Denote $X_{t-1}^{(i)} = (x_{t-K}^{(i)}, x_{t-K+1}^{(i)}, \dots, x_{t-1}^{(i)})$ with maximum time horizon of K , and $\mathbf{X}_{t-1} = \{X_{t-1}^{(i)}\}, i = 1, 2, \dots, N$. We also denote $\mathbf{X}_{t-1}^{(\hat{j})} = \mathbf{X}_{t-1} \setminus X_{t-1}^{(j)}$ (\mathbf{X}_{t-1} excluding $X_{t-1}^{(j)}$). We assume that $x^{(1)}, x^{(2)}, \dots, x^{(N)}$ are generated by stationary response functions h_i that are unknown to the learner:

$$\begin{cases} x_t^{(1)} := h_1(\mathbf{X}_{t-1}, u_1) \\ x_t^{(2)} := h_2(\mathbf{X}_{t-1}, u_2) \\ \dots \\ x_t^{(N)} := h_N(\mathbf{X}_{t-1}, u_N) \end{cases} \quad (6.1)$$

for $t = K + 1, K + 2, \dots$. Here $u_i \in \mathbf{R}^M, i = 1, 2, \dots, N$ are noise variables that are mutually independent, are independent of any $X_{t-1}^{(i)}, x_t^{(i)}, i \in \{1, 2, \dots, N\}$. For any $i, j \in \{1, 2, \dots, N\}$, we assume that the variables $(\mathbf{X}_{t-1}^{(\hat{j})}, X_{t-1}^{(j)}, x_t^{(i)})$ have probability density function $P(\mathbf{X}_{t-1}^{(\hat{j})}, X_{t-1}^{(j)}, x_t^{(i)})$.

Our method is inspired by Granger causality [Gra69, Gra80], which defines causality via predictions, making it especially suitable for relational inference of observational time series. Adapting to our notation:

Granger causality [Gra80]: *Assuming causal sufficiency [PJS17], we say $X_{t-1}^{(j)}, j \neq i$ does not Granger-cause $x_t^{(i)}$, if $P(x_t^{(i)} | X_{t-1}^{(j)}, \mathbf{X}_{t-1}^{(\hat{j})}) = P(x_t^{(i)} | \mathbf{X}_{t-1}^{(\hat{j})})$. Otherwise, we say $X_{t-1}^{(j)}$ Granger-causes $x_t^{(i)}$.*

In practice, we say that time series j Granger-causes time series i , if it can be shown via significance tests that the null hypothesis of $P(x_t^{(i)} | X_{t-1}^{(j)}, \mathbf{X}_{t-1}^{(\hat{j})}) = P(x_t^{(i)} | \mathbf{X}_{t-1}^{(\hat{j})})$ is rejected, i.e. $X_{t-1}^{(j)}$ provides statistically significant information for predicting $x_t^{(i)}$.

In his original work, Granger [Gra69] investigates causality with linear function predictors.

Later works have extended it to kernel methods [AMS04, MPS08b, MPS08a, SQL12], which essentially estimate linear Granger causality on the feature space of the kernel. To learn potentially highly nonlinear response functions, it may be desirable to use expressive and universal function approximators [Hor91] such as neural nets. Neural nets are much more flexible than linear models, and do not require kernel selection as in kernel methods.

6.2.2 Our method

Based on the definition of Granger causality, a naïve way to combine it with neural net is: for each $j \rightarrow i$, train two neural nets, one predicting $x_t^{(i)}$ based on $\mathcal{X}_{t-1}^{(j)}$, another predicting $x_t^{(i)}$ based on the full $\mathcal{X}_{t-1} = (\mathcal{X}_{t-1}^{(j)}, X_{t-1}^{(j)})$, and test whether former MSE is significantly larger than the latter. This method suffers from two major drawbacks: (1) instability: different training of the neural net may end up in different local minima, so that the two MSEs have large variance, which is observed in our initial explorations; (2) inefficiency: to discover the relations among N time series, it has to train at least N^2 models (for each $x_t^{(i)}$, train $N - 1$ models with one $X_{t-1}^{(j)}$ removed, and Q models ($Q \geq 1$) with full \mathcal{X}_{t-1} for accumulating statistics). On the other hand, these two drawbacks exactly inspire our method. Instead of predicting $x_t^{(i)}$ with one $X_{t-1}^{(j)}$ missing at a time, what if we let each $X_{t-1}^{(j)}$ have *learnable* corruption, and encourage each $X_{t-1}^{(j)}$ to provide as little information to $x_t^{(i)}$ as possible while maintaining good prediction? In this way, we have a *single* shared model that can span the full product space of [total corruption, no corruption] $^{\otimes N}$ for N input time series, which is more stable and efficient than the removing one $X_{t-1}^{(j)}$ at a time and training N models. To achieve this, we add independent noise with learnable amplitudes to each input $X_{t-1}^{(j)}$, and measure the corruption by the mutual information between the input and the corrupted input. We then define the following risk:

$$R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\eta}] = \mathbb{E}_{\mathbf{X}_{t-1}, x_t^{(i)}, \boldsymbol{\epsilon}} \left[\left(x_t^{(i)} - f_\theta(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) \right)^2 \right] + \lambda \cdot \sum_{j=1}^N I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) \quad (6.2)$$

where $\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})} := \mathbf{X}_{t-1} + \boldsymbol{\eta} \odot \boldsymbol{\epsilon}$ (or element-wise, $\tilde{X}_{t-1}^{(j)(\eta_j)} := X_{t-1}^{(j)} + \eta_j \cdot \epsilon_j$, $j = 1, 2, \dots, N$) is the noise-corrupted inputs with *learnable* noise amplitudes $\eta_j \in \mathbf{R}^{KM}$, and $\epsilon_j \sim N(\mathbf{0}, \mathbf{I})$. $\lambda > 0$ is a positive hyperparameter for the mutual information $I(\cdot, \cdot)$. Intuitively, the minimization of the second term $I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)})$ requires the noise amplitude η_j to go up. The minimization of the first term requires the noise amplitude η_j to go down, and the larger causal strength from $X_{t-1}^{(j)}$ to $x_t^{(i)}$, the larger this force. The minimization of the two terms strikes a balance, at which point the $I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)})$ measures the *minimum* number of bits of information the time series j need to provide to the learner, without compromising the prediction.

At the minimization of $R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\eta}]$, we define $W_{ji} = I\left(\tilde{X}_{t-1}^{(j)(\eta_j^*)}; X_{t-1}^{(j)}\right)$, which we term *minimum predictive information*, where $(f_{\theta^*}, \boldsymbol{\eta}^*) = \operatorname{argmin}_{(f_\theta, \boldsymbol{\eta})} R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\eta}]$. Essentially, W_{ji} measures the *predictive strength* of time series j for predicting time series i , *conditioned* on all the other observed time series. We have that W_{ji} satisfies the following properties:

- (1) If $x^{(j)} \perp\!\!\!\perp x^{(i)}$, then $W_{ji} = 0$.
- (2) W_{ji} is invariant to affine transformation of each individual $X_{t-1}^{(k)}$, $k = 1, 2, \dots, N$.
- (3) W_{ji} is invariant to reparameterization of θ in f_θ (the mapping remains the same).

The proofs are provided in Appendix A.5.2. Property 1 shows that W_{ji} is able to differentiate time series that are dependent or independent with the target time series i . Empirically, to perform statistical tests, we can let the null hypothesis be $x^{(j)} \perp\!\!\!\perp x^{(i)}$. Before training, we append to \mathcal{X}_{t-1} some fake time series $v_{t-1}^{(s)}$, $s = 1, 2, \dots, S$ (e.g. by randomly permuting $X_{t-1}^{(j)}$) so that $v_{t-1}^{(s)} \perp\!\!\!\perp x_t^{(i)}$. After optimizing w.r.t. to the augmented dataset, the values of W_{si} between $v_{t-1}^{(s)}$ and $x_t^{(i)}$ form a distribution for which we know that the null hypothesis is true. Then if certain W_{ji} is greater than the $1 - \alpha$ quantile (e.g. $\alpha = 0.05$) of the distribution, we can reject the null hypothesis of independence. Properties 2 and 3 show the benefit of our method which essentially regularizes the *input information*, compared with L1 and group L1 [MVDGB08, SCHU17, TCF⁺18] which regularize the *model* and thus do not satisfy these two properties.

Algorithm 3 Relational Learning with Minimum Predictive Information Regularization

Require $x_t^{(i)}, \mathbf{X}_{t-1}$, for $i \in \{1, 2, \dots, N\}$, $t \in \mathbf{T} = \{K+1, K+2, \dots\}$.
Require η_0 : a small value for initialization of $\boldsymbol{\eta}$.
Require λ : coefficient for the mutual information term.
Require S : number of fake time series.
Require α : significance level.

1: Randomly select S indices i_1, i_2, \dots, i_S from $\{1, 2, \dots, N\}$
2: $v_{t-1}^{(s)} \leftarrow \text{Permute-examples}_t(X_{t-1}^{(i_s)})$ **for** $s = 1, 2, \dots, S$ // *Permuting on the example dimension*
3: $\mathcal{X}_{t-1}^{(\text{aug})} \leftarrow [\mathcal{X}_{t-1}, \mathbf{v}_{t-1}]$, where $\mathbf{v}_{t-1} = [v_{t-1}^{(1)}, \dots, v_{t-1}^{(S)}]$ and $[\cdot, \dots, \cdot]$ denotes concatenation along the dimension of N (thus $\mathcal{X}_{t-1}^{(\text{aug})}$ consists of $N + S$ time series)
4: **for** i in $\{1, 2, \dots, N\}$ **do**:
5: Initialize function approximator f_θ .
6: Initialize $\boldsymbol{\eta} = (\eta_1, \eta_2, \dots, \eta_N) = (\eta_0 \mathbf{1}, \eta_0 \mathbf{1}, \dots, \eta_0 \mathbf{1})$, where each element $\eta_0 \mathbf{1}$ is a KM -dimensional vector, same dimension as $X_{t-1}^{(j)}$.
7: $(f_{\theta^*}, \boldsymbol{\eta}^*) \leftarrow \text{Minimize}_{(f_\theta, \boldsymbol{\eta})} \hat{R}_{\mathcal{X}^{(\text{aug})}, x^{(i)}, \epsilon}[f_\theta, \boldsymbol{\eta}]$ (Eq. 6.3) with e.g. gradient descent.
8: $W_{ji} \leftarrow I(\tilde{X}_{t-1}^{(j)(\eta_j^*)}; X_{t-1}^{(j)})$, for $j = 1, 2, \dots, N, N+1, \dots, N+S$.
9: **end for**
10: Accumulate the values of W_{si} between all $v_{t-1}^{(s)}$, $s = 1, 2, \dots, S$ and $x_t^{(i)}$, $i = 1, 2, \dots, N$, and obtain the $1 - \alpha$ quantile as the threshold.
11: Zero the W_{ji} elements ($j, i = 1, 2, \dots, N$) whose value are below the threshold.
12: **return** W // *Return the main $N \times N$ matrix*

Moreover, in Appendix A.5.2 we further provide intuition that under certain conditions, W_{ji} is likely to favor the time series that *directly* causes time series i , compared with the time series that relate to i via the direct causal connections. Note that our method is not *guaranteed* to identify *direct causal* relations (in Granger [Gra80] or Pearl [Pea02] sense), which is a very hard problem given the potential large number of time series and nonlinearity present. However, our method provides an effective data exploratory tool to identify time series that are *predictive* of one another, *conditioned* on all the other observed time series, whose identified directional relations can be investigated further by a researcher. As stated above, under certain conditions, our method does favor the direct causal relations. And in the experiment section, we will compare the estimated W_{ji} with true causal relations if available.

Empirically, we minimize the following empirical risk:

$$\hat{R}_{\mathbf{X}, x^{(i)}, \epsilon}[f_\theta, \boldsymbol{\eta}] = \frac{1}{|\mathbf{T}|} \sum_{t \in \mathbf{T}} \left(x_t^{(i)} - f_\theta(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) \right)^2 + \lambda \sum_{j=1}^N I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) \quad (6.3)$$

In general, it may be inefficient to estimate the mutual information $I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)})$ with large dimension of $X_{t-1}^{(j)}$ such that the expression is also differentiable w.r.t. η_j . Utilizing the property of Gaussian channels, in Appendix A.5.3 we prove that $I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) \leq \frac{1}{2} \sum_{l=1}^{KM} \log \left(1 + \frac{\text{Var}(X_{t-1,l}^{(j)})}{\eta_{j,l}^2} \right)$, where l denotes the l^{th} element of a vector, and $\text{Var}(X_{t-1,l}^{(j)})$ is the variance of $X_{t-1,l}^{(j)}$ across t . Therefore, in practice to improve efficiency, we can optimize an *upper bound* of the risk:

$$\hat{R}_{\mathbf{X}, x^{(i)}, \epsilon}^{\text{upper}}[f_\theta, \boldsymbol{\eta}] = \frac{1}{|\mathbf{T}|} \sum_{t \in \mathbf{T}} \left(x_t^{(i)} - f_\theta(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) \right)^2 + \frac{\lambda}{2} \sum_{j=1}^N \sum_{l=1}^{KM} \log \left(1 + \frac{\text{Var}(X_{t-1,l}^{(j)})}{\eta_{j,l}^2} \right) \quad (6.4)$$

When the dimension of $X_{t-1}^{(j)}$ is large, a differentiable estimate of the mutual information (e.g. MINE [BRB⁺18]) can be applied. We provide Algorithm 3 to empirically estimate W_{ji} , which we term relational learning with Minimum Predictive Information Regularization (MPIR). The steps 1-3 construct fake input time series $v_{t-1}^{(s)}$, $s = 1, 2, \dots, S$ (which we know the null hypothesis of $v_{t-1}^{(s)} \perp\!\!\!\perp x_t^{(i)}$ is true) to append to \mathcal{X}_{t-1} . Steps 4-9 optimize the objective w.r.t. the augmented dataset, and obtain a $(N + S) \times N$ matrix W_{ji} . Steps 10-11 performs significance test and only preserve the W_{ji} values in the main $N \times N$ matrix that are statistically significant. Finally the main matrix is returned.

To select an appropriate hyperparameter λ , we can additionally append to the target $x_t^{(i)}$ a few time series w_t constructed from \mathcal{X}_{t-1} . We then select λ such that the estimated causal strength between \mathcal{X}_{t-1} and w_t (for which we know the causal relations) is at least 4σ away from the estimated causal strength between v_{t-1} and w_t (for which we know that they are independent). See Appendix A.5.1 for details.

6.3 Experiments

To demonstrate that our proposed method is able to discover interesting underlying directional (possibly causal) relations, we test it on both synthetic and real datasets. We first use synthetic datasets, where we know the underlying causal structure and compare with other methods. We then test whether our algorithm can infer directional relations among trajectories of objects from watching an agent playing video games. Finally, we apply our algorithm to a real-world heart-rate vs. breath-rate dataset and a rat EEG dataset to test its effectiveness. We use the $\hat{R}_{\mathbf{X},x^{(i)},\epsilon}^{\text{upper}}[f_\theta, \boldsymbol{\eta}]$ (Eq. 6.4) for optimization for all experiments.

6.3.1 Synthetic experiment with log-normal causal strengths

In this experiment, we evaluate our method together with other methods using a nonlinear synthetic dataset generated to have a known causal structure (hidden to the methods being compared). We study performance with varying number N of time series, with N up to 30. To generate the data, we let each $x_t^{(i)}$ have dimension $M = 1$, and also set the maximum time horizon $K = 3$, so each $X_{t-1}^{(j)}$ is a $K \times M = 3 \times 1$ matrix. We use the following realization of the response function h_i in Eq. (6.1):

$$x_t^{(i)} = h_i(\mathbf{X}_{t-1}, u_t) = \mathbf{H}_1 \left(\sum_{j=1}^N \left[A_{ji} \odot \mathbf{H}_2(B_j \odot X_{t-1}^{(j)}) \right] \right) + u_t, i = 1, 2, \dots, N \quad (6.5)$$

where $u_t \sim N(\mathbf{0}, \mathbf{I}) \in \mathbf{R}^M$, \odot denotes element-wise multiplication, and \mathbf{H}_1 and \mathbf{H}_2 are two nonlinear functions to make the response functions nonlinear. In this experiment, we use $\mathbf{H}_1(x) = \text{softplus}(x) = \log(1 + e^x)$, and $\mathbf{H}_2(x) = \tanh(x)$. B_j is a $K \times M$ random matrix, whose element is sampled from $U[-1, 1]$. A_{ji} is a $K \times M$ matrix, with 0.5 probability of being a zero matrix and 0.5 probability of being a nonzero random matrix, characterizing the underlying causal strength from j to i . Crucially, to reflect that the causal strength may span different orders of magnitude, if A_{ji} is sampled to be a nonzero matrix, then the amplitude of each of its element is sampled from a log-normal distribution with $\mu = 1, \sigma = 0$, their sign sampling from $U\{-1, 1\}$. Denote $\mathbb{1}(A)$ as the 0-1 indicator matrix of causality

$(\mathbb{1}(A)_{ji} = 1 \text{ if } |A_{ji}| > 0; 0 \text{ otherwise})$. The goal of each algorithm being evaluated is to produce an $N \times N$ score matrix \tilde{A} , where each entry \tilde{A}_{ji} characterizes the directional strength from j to i . Then the flattened \tilde{A} is evaluated against the flattened $\mathbb{1}(A)$ (excluding diagonal elements of the matrices) via different metrics. Fig. SA.4 in Appendix A.5.5 shows example snapshots of the time series.

In general, for a large N , the number of possible causal graphs grows double exponentially: there are 2^{N^2} possible matrix of $\mathbb{1}(A)$. To give an estimate, for $N = 3, 4, 5, 8, 10, 20, 30$, there are $512, 6.6 \times 10^4, 3.3 \times 10^7, 1.8 \times 10^{19}, 1.2 \times 10^{30}, 2.6 \times 10^{120}, 8.5 \times 10^{270}$ number of possible graphs, respectively. Therefore, estimating the underlying causal graph is in general a non-trivial task when N is large. We compare our algorithm with previous methods including transfer entropy [Sch00], causal influence [JBGW⁺13], linear Granger causality [Gra69, DCB06], kernel Granger causality [MPS08b, MPS08a], and three baselines: (1) mutual information $\tilde{A}_{ji} = I(X_{t-1}^{(j)}; x_t^{(i)})$ (which gives $\tilde{A}_{ji} = \tilde{A}_{ij}$), (2) a sparse feature selection method, elastic net [ZH05], and (3) a random matrix, each element of which is drawn from a standard Gaussian distribution. For each N , we sample 10 datasets with different A_{ji} and B_j matrices, and compare each method's average performance over 10 datasets together with their standard deviation. The implementation details for each method and each experiment are provided in Appendix A.5.4 and A.5.5, respectively. Since many of the methods do not provide a threshold or significance test, we use the standard metrics of area under the precision-recall curve (AUC-PR) [DG06a] (Table 6.1 below) and area under the ROC curve (AUC-ROC) (Table SA.1 in Appendix A.5.6) to compare their performance.

We see that for smaller N ($N \leq 4$), methods with smaller expressivity (linear Granger, kernel Granger) performs slightly better. However, as N becomes larger, our method outperforms other methods with increasing margin, demonstrating our method's capability to infer complex relational structures from interacting time series. Particularly, although two linear methods, linear Granger and elastic net, have relatively strong performance with $N \leq 5$, they quickly degrade with larger N due to more nonlinearity present in the data. With the help of kernels, kernel Granger degrades slower, but can not compete in larger N with our method which allows expressive neural nets to model complex nonlinear

Table 6.1: Mean and standard deviation of AUC-PR (%) vs. N , over 10 random sampling of datasets. Bold font marks the top method for each N .

	N	3	4	5	8	10	15	20	30
method									
MPIR (ours)		$97.5_{\pm 5.3}$	$98.4_{\pm 2.5}$	$97.6_{\pm 2.7}$	$96.1_{\pm 2.4}$	$93.5_{\pm 3.7}$	$91.3_{\pm 3.0}$	$85.9_{\pm 2.4}$	$76.3_{\pm 1.5}$
Mutual Information		$90.5_{\pm 13.7}$	$93.3_{\pm 3.8}$	$90.0_{\pm 4.3}$	$82.4_{\pm 5.1}$	$76.9_{\pm 9.3}$	$76.8_{\pm 4.8}$	$71.9_{\pm 3.8}$	$70.6_{\pm 3.1}$
Transfer Entropy		$93.5_{\pm 7.7}$	$97.3_{\pm 3.3}$	$91.6_{\pm 8.2}$	$83.7_{\pm 7.2}$	$76.2_{\pm 5.7}$	$67.1_{\pm 4.2}$	$61.2_{\pm 4.3}$	$55.7_{\pm 2.5}$
Linear Granger		$99.4_{\pm 1.8}$	$97.8_{\pm 2.5}$	$92.0_{\pm 8.3}$	$83.1_{\pm 8.8}$	$79.4_{\pm 9.2}$	$71.0_{\pm 10.0}$	$63.7_{\pm 8.8}$	$52.4_{\pm 1.7}$
Kernel Granger		$99.3_{\pm 2.3}$	$99.3_{\pm 1.5}$	$96.5_{\pm 4.8}$	$92.5_{\pm 3.4}$	$90.0_{\pm 3.3}$	$86.0_{\pm 2.4}$	$81.0_{\pm 4.0}$	$73.1_{\pm 1.8}$
Elastic Net		$99.1_{\pm 2.9}$	$98.5_{\pm 2.0}$	$95.7_{\pm 4.2}$	$88.9_{\pm 6.2}$	$83.6_{\pm 4.6}$	$79.1_{\pm 3.0}$	$75.3_{\pm 3.6}$	$69.1_{\pm 5.8}$
Causal Influence		$67.5_{\pm 26.7}$	$60.2_{\pm 24.1}$	$59.3_{\pm 15.3}$	$44.1_{\pm 8.9}$	$42.7_{\pm 7.8}$	$47.0_{\pm 3.1}$	$44.5_{\pm 4.1}$	$44.6_{\pm 2.1}$
Gaussian random		$60.0_{\pm 14.7}$	$57.9_{\pm 12.9}$	$51.6_{\pm 8.0}$	$44.5_{\pm 5.6}$	$41.3_{\pm 6.2}$	$44.6_{\pm 4.0}$	$44.0_{\pm 2.4}$	$44.3_{\pm 2.4}$

interactions. For the Causal Influence method, although it has very good mathematical properties, it may be impractical in practice, as is also shown in the table. This is due to that it is defined as the KL-divergence between $(\mathbf{X}_{t-1}, x_{t-1}^{(i)})$ and its counterpart (whose causal arrows to and from time series j are cut), each of which is an $(NK + 1)M$ -dimensional vector, which can quickly go to high dimensions, where density estimation required to calculate KL-divergence is in general data-hungry and difficult. In comparison, our method that estimates predictive strength via minimizing prediction errors is comparatively easier in high dimensions.

6.3.2 Experiments with video games

To see how our method can discover the directional (possibly causal) relations in real video games, and potentially improve reinforcement learning (RL) or imitation learning (IL), we apply our method to the relational inference between the trajectories of different objects from a trained CNN RL-agent playing Atari Breakout games ([BNVB13], implementation details see Appendix A.5.8). Fig. 6-1 shows the inferred W_{ji} matrix for our method and compared methods, respectively. The true underlying causal chain is marked in dark color in Fig. 6-1e, with light color marking the competing causal relations that are indistinguishable from data (e.g. decrease of bricks and increase of reward happen at the same time step, so

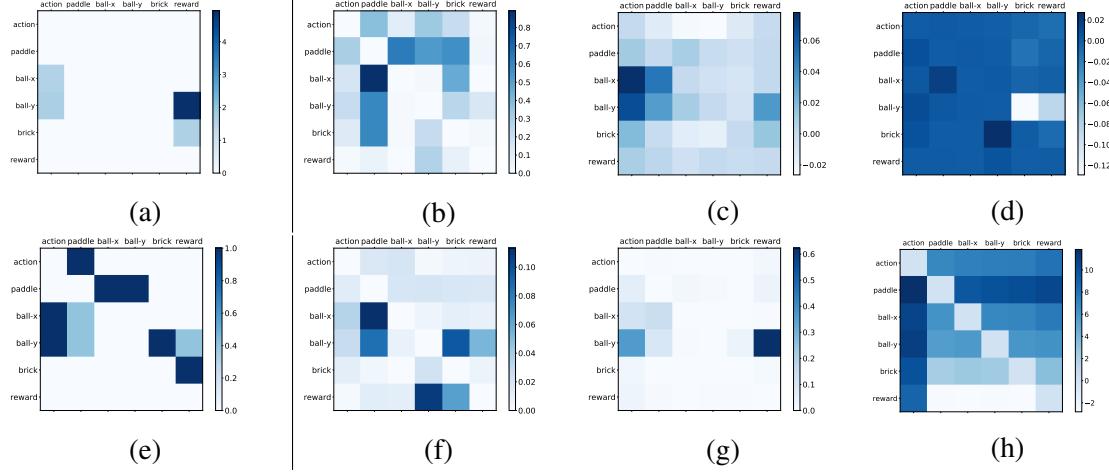


Figure 6-1: (a) Predictive strength W_{ji} inferred by our method in Section 6.3.2. The (j, i) element denotes the inferred causal strength from j to i . (e) True underlying causal relations are marked dark, with light color marking competing causal relations that are indistinguishable from data. Other subfigures are: directional strength inferred by (b) mutual information (c) transfer entropy (d) linear Granger (f) kernel Granger (g) elastic net (h) causal influence.

we cannot distinguish $\text{ball-y} \rightarrow \text{brick}$ and $\text{ball-y} \rightarrow \text{reward}$). Compared with other methods, we see that our method is able to discover comparatively most of the causal relations without finding false positives. Specifically, it correctly discovers a prominent causal direction from the ball’s y position to the reward, as well as $\text{brick} \rightarrow \text{reward}$, $\text{ball-x} \rightarrow \text{action}$, $\text{ball-y} \rightarrow \text{action}$. The latter two show that the ball’s x and y positions also have influences on the trained agent’s action: in order that the ball does not fall to the bottom, the agent has to position itself at the right position depending on the x and y positions of the ball.

In comparison, mutual information (Fig. 6-1b) gives a symmetric matrix that does not differentiate the two possible directions, and also misses the arrows $\text{ball-y} \rightarrow \text{brick} \rightarrow \text{reward}$. For transfer entropy (Fig. 6-1c), although it correctly discovers a number of causal arrows, it also gives relatively high scores for some incorrect arrows: $\text{brick} \rightarrow \text{action}$, $\text{ball-y} \rightarrow \text{ball-x}$. For kernel Granger (Fig. 6-1f), although it correctly discovers four causal relations, it also incorrectly finds $\text{reward} \rightarrow \text{ball-y}$ and $\text{reward} \rightarrow \text{brick}$. For elastic net (Fig. 6-1g), it correctly discovers two prominent causal relations: $\text{ball-y} \rightarrow \text{action}$ and $\text{ball-y} \rightarrow \text{reward}$, but misses a few others. Linear Granger (Fig. 6-1d) and causal influence (Fig. 6-1h) fail to discover useful causal arrows.

6.3.3 Experiment with heart-rate vs. breath-rate and rat brain EEG datasets

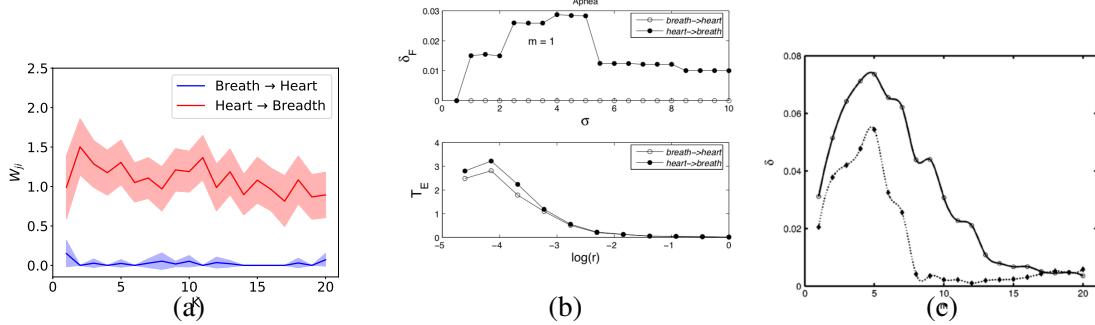


Figure 6-2: (a) Predictive strength W_{ji} inferred by our method with the heart-rate vs. breath-rate dataset, averaged over 50 initializations of f_θ . The shaded areas are the 95% confidence interval. (b) Upper: the filtered causality index vs. varying width of Gaussian kernel σ [MPS08b]; lower: transfer entropy vs. r , the length scale [Sch00]; (c) The causality index for breath→heart (lower) and heart→breath (upper) in [AMS04], where m is the maximum time lag (equivalent to our K).

Now we test our algorithm with real-world datasets. As a common dataset studied in previous causal works, we use the time-series of the breath rate and instantaneous heart rate of a sleeping patient suffering from sleep apnea (samples 2350-3550 of data set B from Santa Fe Institute time series contest held in 1991, available in [Phy]). We apply our method to infer the directional relations between the breath rate and heart rate, with different maximum time horizon K . The result is shown in Fig. 6-2. We see that the predictive strength W_{ji} from heart to breath is significantly higher than the reverse direction that is basically 0, consistent with the results from previous causal inference methods [Sch00, AMS04, MPS08b] as also shown in Fig. 6-2(b)(c). Notably, the W_{ji} from heart to breath estimated by our method remains at roughly the same level for different K s, in contrast to the decaying causality index w.r.t. increasing history length in ([AMS04], Fig. 6-2 (c)), showing a merit of our method in estimating directional strength across different time-horizons, aided by the flexibility of neural nets in extracting the right information to predict the future. The implementation details are provided in Appendix A.5.9. In addition, in Appendix A.5.10 we test our algorithm on a rat EEG dataset, and obtain consistent result with previous works.

6.4 Discussion and conclusion

In this paper, we have introduced a novel relational learning with Minimum Predictive Information Regularization (MPIR) method for exploratory discovery of nonlinear directional relations from observational time series. It allows functional approximators like neural nets to learn complex directional relations from time series data. We prove its three theoretical properties, and provide intuition that it favors variables that directly cause the variable of interest. We demonstrate in synthetic datasets, a video game environment and heart-rate vs. breath-rate datasets, that our method has better capability to handle nonlinearity, and can scale to large numbers of time series. We believe our work endows practitioners with a useful tool for deciphering the directional relations in complex systems, and are excited to see it in broader applications.

Chapter 7

Meta-learning autoencoders for few-shot prediction

Compared¹ to humans, machine learning models generally require significantly more training examples and fail to extrapolate from experience to solve previously unseen physical prediction tasks. To help close this performance gap, we augment single-task neural networks with a meta-recognition model which learns a succinct model code via its autoencoder structure, using just a few informative examples. The model code is then employed by a meta-generative model to construct parameters for the task-specific model. We demonstrate that for previously unseen tasks, without additional training, this *Meta-Learning Autoencoder* (MeLA) framework can build models that closely match the true underlying models, with losses significantly lower than fine-tuned baseline networks, and performance that compares favorably with state-of-the-art meta-learning algorithms. MeLA also adds the ability to identify influential training examples and predict which additional unseen data will be most valuable to improve model prediction.

¹The paper “Meta-learning autoencoders for few-shot prediction” is under review. Authors: Tailin Wu, John Peurifoy, Isaac L. Chuang, Max Tegmark. arXiv:1807.09912.

7.1 Introduction

Physical reasoning with few examples is an essential part of human-like intelligence. Humans are not only able to develop physical models that can describe the dynamics of objects in a single environment, we can generalize to a continuum of models in unseen environments with few observations (known as few-shot learning). For example, after seeing several moving objects accelerated by different forces in different environments, humans are able to develop a meta-model that can generalize to a continuum of unseen accelerated objects. Upon arriving at a new environment and seeing an object moving for only a short time, s/he can quickly propose a new model for the moving object, including a good estimate of its acceleration. Incorporating this ability of generalization beyond the initial environments (training data) for quick recognition from few examples remains an important challenge in machine learning.

Great progress has been made in recent years towards developing machine learning models for physical systems. For example, disentangling recognition and dynamics models[FKPW17], visual de-animation[WLK⁺17], modeling physical interactions[CUTT16, BPL⁺16, WZW⁺17b]. However, most works aim to learn models that perform well in a single environment, without considering generalization to unseen environments where the dynamics or the environment constraint is novel. Hence when encountering new environments with different dynamics or environment constraints, the model has to be relearned. Moreover, the training of the models usually requires a large number of examples, posing a performance gap compared with humans.

In this work, we tackle the above problems by proposing a novel class of neural network architecture for few-shot/meta-learning of physical models, which learns to generalize across environments so that the learning of the dynamics in an unseen environment only requires a few examples. Although much progress has been made in recent years in few-shot/meta-learning, a large number of works are specifically designed for classification [VBL⁺16, KZS15, SSZ17, ES16], and the regression benchmark is only a simple trigonometric sine regression problem. Our work fills this gap, by introducing a *Meta-Learning Autoencoders* (MeLA) architecture, designed for few-shot physical prediction/regression.

At its core, MeLA consists of a learnable meta-recognition model that can for each (unseen) task distill a few input-output examples into a model code vector parameterizing the task’s functional relationship, and a learnable meta-generative model that maps this model code into the weight and bias parameters of a neural network implementing this function. This architecture forces the meta-recognition model to discover and encode the important variations of the functional mappings for different tasks, and the meta-generative model to decode the model codes to corresponding task-specific models with a common model-generating network.

This brings the key innovation of MeLA: for a class of tasks, MeLA does not attempt to learn a *single* good initialization for multiple tasks[FAL17], or learn an update function[Sch87, BBCG92, ADG⁺16], or learn an update function together with a *single* good initialization[LZCL17, RL16]. Instead, it learns to map the few examples from different datasets into *different* models, which not only allows for more diverse model parameters tailored for each individual tasks, but also obviates the need for fine-tuning. Moreover, by encoding each function as a vector in a single low-dimensional latent space, MeLA is able to generalize beyond the training datasets, by both interpolating between and extrapolating beyond learned models into a continuum of models. We will demonstrate that the meta-learning autoencoder has the following 3 important capabilities:

1. **Augmented model recognition:** MeLA strategically builds on a pre-existing, single-task trained network for physical prediction, augmenting it with a second network used for meta-recognition. It achieves lower loss in unseen environments at zero and few gradient steps, compared with both the original architecture upon which it is based and state-of-the-art meta-learning algorithms.
2. **Influence identification:** MeLA can identify which examples are most useful for determining the model (for example, a rectangle’s vertices have far greater influence in determining its position and size than its inferior points).
3. **Interactive learning:** MeLA can actively request new samples which maximize its ability to learn models.

7.2 Methods

7.2.1 Meta-learning problem setup

We are interested in modeling a set of vector-valued functions \mathbf{h}_α (which we will refer to as *models*), that each map an m -dimensional input vector \mathbf{x} into an n -dimensional output vector \mathbf{y} . Let's first consider the case for a single dataset. Given many input-output pairs $\mathbf{y}_i = \mathbf{h}_\alpha(\mathbf{x}_i)$ linked by the same function \mathbf{h}_α , we group the corresponding vectors into matrices \mathbf{X} and \mathbf{Y} whose i^{th} rows are the vectors \mathbf{x}_i and \mathbf{y}_i . Since we focus on physical prediction/regression, the target $\mathbf{y} \in \mathbf{R}^n$ is continuous. This class of problems includes a wide range of scenarios, *e.g.*, modeling time series data, learning physics and dynamics, and frame-to-frame prediction of videos.

The meta-learning problem we tackle is as follows. Suppose that we are given an ensemble of datasets $\{\mathbf{D}^\alpha\} = \{(\mathbf{X}^\alpha, \mathbf{Y}^\alpha)\}$, $\alpha = 1, 2, \dots$, each of which is generated by a corresponding function \mathbf{h}_α . In the single-task scenario, we want to train a model f that predicts all output vectors for the corresponding input vectors from a single dataset, to minimize some loss function ℓ that quantifies the prediction errors. The meta-learning goal is, after training on an ensemble of training datasets $\mathbf{D}^1, \dots, \mathbf{D}^a$, to be able to quickly learn from few examples from held-out datasets \mathbf{D}^{a+1}, \dots , and obtain a low loss on them.

7.2.2 Meta-learning autoencoder architecture

The architecture of our Meta-Learning Autoencoder (MeLA) is illustrated in Figure 7-1. It is defined by three vector-valued functions \mathbf{f}_θ , \mathbf{g}_γ and \mathbf{m}_μ that are defined by feedforward neural networks parametrized by vectors θ , γ and μ , respectively. In contrast to prior methods for learning to quickly adapt to different datasets [FAL17] or using memory-augmented setup [SBB⁺16, VBL⁺16], the MeLA takes full advantage of the prior that the datasets are generated by a hidden model class, where the functions \mathbf{h} lie in a relatively low-dimensional submanifold of the space of all functions. Based on this prior, we use a meta-recognition model \mathbf{m}_μ that maps a whole dataset $\mathbf{D} = (\mathbf{X}, \mathbf{Y})$ to a model code vector

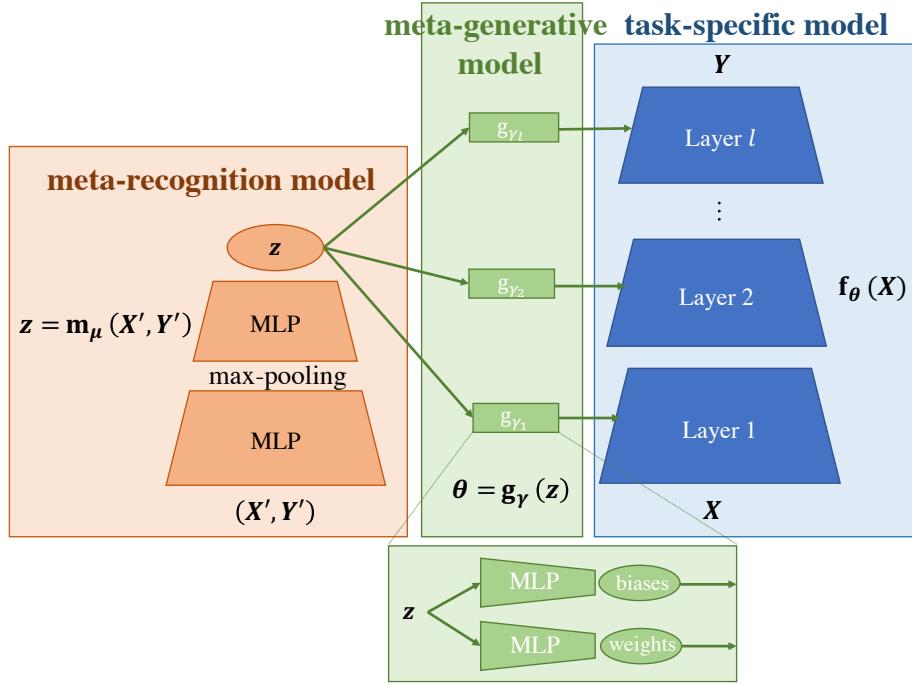


Figure 7-1: Architecture of our Meta-Learning Autoencoder (MeLA). MeLA augments a pre-existing neural network architecture f_θ (right) with a meta-recognition model (left) that generates the model code z based on a few examples X' , Y' , and a meta-generative model (middle) that generates the parameters θ of model f_θ based on the model code. f_θ , g_γ and \mathbf{m}_μ are implemented as multilayer perceptron (MLP).

z , and a meta-generative model g_γ that maps z to the parameters vector θ of the network implementing the function f_θ . In other words, $\theta = g_\gamma(z)$, and for a specific dataset (X, Y) , f_θ can be instantiated by

$$f_\theta = f_{g_\gamma(z)} = f_{g_\gamma(m_\mu(X, Y))}. \quad (7.1)$$

This architecture is designed so that it can easily transform a neural network that is originally intended to learn from a single task into an architecture that can perform meta/few-shot learning on a number of tasks, combining the knowledge of individual task architectures with MeLA's meta-learning power. If the original single-task model is f_θ , then without changing the architecture of f_θ , we can simply attach a meta-recognition model \mathbf{m}_μ and a meta-generative model g_γ that generates the parameters of f_θ , and train on an ensemble of tasks.

Network architecture examples Although the MeLA architecture described above can be implemented with any choices whatsoever for the three feedforward neural networks that define the functions f_θ , g_γ and m_μ , let us consider simple specific implementations to build intuition and get ready for the numerical experiments.

Suppose we implement the main model f_θ as a network with two hidden layers with s_1 and s_2 neurons, respectively. Its input size is s_0 and its output size is s_{out} . The meta-recognition model m_μ takes as input \mathbf{X} and \mathbf{Y} concatenated horizontally into a single $N \times (s_0 + s_{\text{out}})$ matrix, where N is the number of training examples at hand. The feedforward neural network implementing m_μ has two parts: the first is a series of layers that collectively transform the $N \times (s_0 + s_{\text{out}})$ input matrix into an $N \times s_{\text{pool}}$ matrix, where s_{pool} is the number of output neurons in this first block (we typically use 200 to 400 below). Then a max-pooling operation is applied over the N examples, transforming this $N \times s_{\text{pool}}$ matrix into a single vector of length s_{pool} . The meta-recognition model m_μ is thus defined independently of the number of training examples N . As will be explained in the “Influence identification” subsection below, max-pooling is key to MeLA, forcing the meta-recognition model to learn to capture key characteristics in a few representative examples. The second block of the m_μ network is a multilayer feedforward neural network, which takes as input the max-pooled vector, and transforms it into a s_{code} -dimensional model code \mathbf{z} that parametrizes the functional relationship between \mathbf{x} and \mathbf{y} .

The meta-generative model g_γ takes as input the model code \mathbf{z} , and for each layer in the main model f_θ , it has two separate neural networks that map \mathbf{z} to all the weight and bias parameters of that layer. We typically implement each of these subnetworks of g_γ using 2-3 hidden layers with 60 neurons each. The number of parameters for the new model $f_{g_\gamma(z)}$ is linear w.r.t. the number of parameters for the original model f_θ , independent of the number of tasks.

7.2.3 MeLA’s meta-training and evaluation

The extension from the training on a single-task f_θ to MeLA is straightforward. Suppose that the loss function for the single-task is $\ell(\hat{\mathbf{y}}, \mathbf{y})$, with expected risk $R_{\ell, \mathbf{D}_k}(f_\theta) \equiv$

Algorithm 4 Meta-Training for MeLA

Require datasets $\{\mathbf{D}^\alpha\} = \{(\mathbf{X}^\alpha, \mathbf{Y}^\alpha)\}, \alpha = 1, 2, \dots, a$
Require n : number of meta-iterations
Require β : learning rate hyperparameter

- 1: Initialize random parameters for $\mathbf{m}_\mu, \mathbf{g}_\gamma$.
- 2: $i \leftarrow 0$
- 3: **while** $i < n$:
- 4: $\{\mathbf{D}^{\alpha'}\} \leftarrow \text{permute}(\{\mathbf{D}^\alpha\})$ //Randomly permute the order of datasets.
- 5: **for** \mathbf{D}_j in $\{\mathbf{D}^{\alpha'}\}$ **do**
- 6: Split $\mathbf{D}_j = (\mathbf{X}_j, \mathbf{Y}_j)$ into training examples $(\mathbf{X}_j^{\text{train}}, \mathbf{Y}_j^{\text{train}})$ and testing examples $(\mathbf{X}_j^{\text{test}}, \mathbf{Y}_j^{\text{test}})$
- 7: $\mathbf{z} \leftarrow \mathbf{m}_\mu(\mathbf{X}_j^{\text{train}}, \mathbf{Y}_j^{\text{train}})$
- 8: $\boldsymbol{\theta} \leftarrow \mathbf{g}_\gamma(\mathbf{z})$
- 9: Update $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} - \beta \nabla_{\boldsymbol{\mu}} \ell[\mathbf{f}_{\mathbf{g}_\gamma(\mathbf{z})}(\mathbf{X}_j^{\text{test}}), \mathbf{Y}_j^{\text{test}}]$
 $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} - \beta \nabla_{\boldsymbol{\gamma}} \ell[\mathbf{f}_{\mathbf{g}_\gamma(\mathbf{z})}(\mathbf{X}_j^{\text{test}}), \mathbf{Y}_j^{\text{test}}]$
- 10: **end for**
- 11: $i \leftarrow i + 1$
- 12: **end while**

$\mathbb{E}_{(\mathbf{X}, \mathbf{Y}) \sim \mathbf{D}_k} [\ell(\mathbf{f}_\theta(\mathbf{X}), \mathbf{Y})]$. Then the meta-expected risk for MeLA is

$$R_{\ell, p(\mathbf{D})}(\mathbf{m}_\mu, \mathbf{g}_\gamma) = \mathbb{E}_{\mathbf{D}_k \sim p(\mathbf{D})} [\mathbb{E}_{(\mathbf{X}, \mathbf{Y}) \sim \mathbf{D}_k} [\ell(\mathbf{f}_{\mathbf{g}_\gamma(\mathbf{z})}(\mathbf{X}), \mathbf{Y})]] \quad (7.2)$$

where $p(\mathbf{D})$ is the distribution for datasets $\{\mathbf{D}_k\}$ generated by the hidden model class \mathbf{h} . The goal of meta-training is to learn the parameters for the meta-recognition model \mathbf{m}_μ and meta-generative model \mathbf{g}_γ such that $R_{\ell, p(\mathbf{D})}(\mathbf{m}_\mu, \mathbf{g}_\gamma)$ is minimized:

$$(\boldsymbol{\mu}, \boldsymbol{\gamma}) = \arg \min (\boldsymbol{\mu}, \boldsymbol{\gamma}) R_{\ell, p(D)}(\mathbf{m}_\mu, \mathbf{g}_\gamma) \quad (7.3)$$

Algorithm 4 illustrates the step-by-step meta-training process for MeLA implementing an empirical meta-risk minimization for Eq. (7.3). In each iteration, the training dataset ensemble is randomly permuted, from which each dataset is selected once for inner-loop task-specific training. Inside the task-specific training, the training examples for each dataset are used for calculating the model code $\mathbf{z} = \mathbf{m}_\mu(\mathbf{X}^{\text{train}}, \mathbf{Y}^{\text{train}})$, after which the model parameter vector $\boldsymbol{\theta} = \mathbf{g}_\gamma(\mathbf{z})$ and the testing examples are used to calculate the task-specific testing loss $\ell(\mathbf{f}_{\mathbf{g}_\gamma(\mathbf{z})}(\mathbf{X}^{\text{test}}), \mathbf{Y}^{\text{test}})$, from which the gradients w.r.t. $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$ are computed and

used for one-step of gradient descent for the meta-recognition model and meta-generative model. Note that here the task-specific testing loss in the training datasets serves as the training loss in the meta-training.

During the evaluation of MeLA, we use the held-out datasets unseen during the meta-training. For each held-out dataset, we split it into training and testing examples. The training examples is fed to MeLA and a task-specific model is generated without any gradient descent. Then we evaluate the task-specific model on the testing examples in the held-out datasets. We also evaluate whether the task-specific model can further improve with a few more steps of gradient descent.

7.2.4 Influence identification

The max-pooling over examples in the meta-recognition model \mathbf{m}_μ is key to MeLA, and also provides a natural way to identify the influence of each example on the model \mathbf{f}_θ . Typically, some examples are more useful than others in determining the model. For example, suppose that we try to learn a function \mathbf{f}_θ defined on \mathbf{R}^2 that equals 1 inside a polygon and 0 outside, with different polygons corresponding to different models parameterized by θ . Then data points near the polygon vertices carry far more information about θ than do points in the deep interior, and the max-pooling over the dimension of examples forces the meta-recognition model to recognize those influential points, and based on them perform computation that returns a model code that determines the whole polygon. Recall that max-pooling compresses $N \times s_{\text{pool}}$ numbers into merely s_{pool} , which means that for each column of the $N \times s_{\text{pool}}$ matrix, only one of the N examples takes the maximum value and hence contributes to this feature. We therefore define the *influence* of an example as

$$\text{Influence} = \frac{\text{Number of columns where the example is maximal}}{s_{\text{pool}}} \quad (7.4)$$

The influence of each example can be interpreted as a percentage, since it lies in $[0, 1]$, and the influences sum to 1 for all the examples in the dataset fed to the meta-recognition model.

7.2.5 Interactive learning

In some situations, measurements are hard or costly to obtain. It is then helpful if we can do better than merely acquiring random examples, and instead determine in advance at which data points \mathbf{x}_i to collect measurements \mathbf{y}_i to glean as much information as possible about the correct function f . Specifically, suppose that we want to predict $\hat{\mathbf{y}}^* = \mathbf{f}_\theta(\mathbf{x}^*)$ as accurately as possible at a given input point \mathbf{x}^* where we have no training data. If before making our prediction, we have the option to measure \mathbf{y} at one of several candidate points $\mathbf{x}'_1, \mathbf{x}'_2, \dots, \mathbf{x}'_n$, then which point shall we choose?

The MeLA architecture provides a natural way to answer this question. We can first use \mathbf{f}_θ to calculate the current predictions for $\mathbf{y}'_1, \mathbf{y}'_2, \dots, \mathbf{y}'_n$ at $\mathbf{x}'_1, \mathbf{x}'_2, \dots, \mathbf{x}'_n$ based on current model generated by $\theta = \mathbf{g}_\gamma(\mathbf{z})$ and $\mathbf{z} = \mathbf{m}_\mu(\mathbf{X}', \mathbf{Y}')$, where \mathbf{X}' and \mathbf{Y}' are the examples that are already given. Then we can fix the meta-parameters μ and γ , and calculate the *sensitivity matrix* of \mathbf{y}^* w.r.t. each current prediction \mathbf{y}'_i :

$$\frac{\partial \mathbf{y}^*}{\partial \mathbf{y}'_i} = \mathbf{J} \frac{\partial \mathbf{z}}{\partial \mathbf{y}'_i}, \quad \text{where} \quad \mathbf{J} \equiv \frac{\partial \mathbf{f}_{\mathbf{g}_\gamma(\mathbf{z})}(\mathbf{x}^*)}{\partial \mathbf{z}}. \quad (7.5)$$

We can select the candidate point whose sensitivity matrix has the largest determinant, *i.e.*, the point for which the measured data carries the most information about the answer \mathbf{y}^* that we want:

$$\mathbf{y}'_i = \arg \max y'_i \left| \frac{\partial \mathbf{y}^*}{\partial \mathbf{y}'_i} \right| \quad (7.6)$$

If we model our uncertainty about \mathbf{y}^* as a multivariate Gaussian distribution, then this criterion maximizes the entropy reduction, *i.e.*, the number of bits of information learned about \mathbf{y}^* from the new measurement. Note that with γ fixed and for a given \mathbf{x}^* , the Jacobian matrix \mathbf{J} is independent of the different candidate inquiry inputs $\mathbf{x}'_1, \mathbf{x}'_2, \dots, \mathbf{x}'_n$. This means that we can simply select the candidate point that has the largest “projection” of $\left| \mathbf{J} \frac{\partial \mathbf{z}}{\partial \mathbf{y}'_i} \right|$ onto \mathbf{J} , requiring in total only one forward and one backward pass for all the candidate examples to obtain the gradient. This factorization emerges naturally from MeLA’s architecture.

7.3 Related work

MeLA addresses few-shot learning of physical dynamics models. Past works of learning physical dynamics models generally consider learning in a single environment, where the training and testing tasks have the same dynamics or environment constraint[FKPW17, WLK⁺17, CUTT16, BPL⁺16, WZW⁺17b]. Our MeLA architecture is complementary in that it can boost the architecture designed for single-task learning into one that can quickly adapt to new tasks with few examples.

MeLA addresses few-shot/meta-learning [TP12, Sch87, NM92], whose goal is to quickly adapt to new tasks with one-shot or few-shot examples. A recent innovative meta-learning method MAML[FAL17] optimizes the parameters of the model so that it is easy to fine-tune to individual tasks in a few gradient steps. Another class of methods focuses on learning a learning rule or update functions[Sch87, BBCG92, ADG⁺16], or learning an update function from a single good initialization[LZCL17, RL16]. Compared to these methods that only learn a *single* good initialization point or how to update from a single initialization point, our method learns recognition and generative models that can quickly determine the model code for the model, and directly propose the appropriate neural network parameters tailored for each task without the need of fine-tuning.

Another interesting class of few-shot learning methods uses memory-augmented networks. [VBL⁺16] proposes matching nets for one-shot classification, which generates the probability distribution for the test example based on the support set using attention mechanisms, essentially learning a “similarity” metric between the test example and the support set. [SBB⁺16] utilizes a neural Turing machines for few-shot learning, and [DSC⁺16, WKNT⁺16] learn fast reinforcement learning agents with recurrent policies using memory-augmented nets. In contrast to memory-augmented approaches, our model learns to distill features from representative examples and produces a model code, based on which it directly generates the parameters of the main model. This eliminates the need to store the examples for the support set, and allows a continuous generation of models, which is especially suitable for generating a continuum of regression models. Other few-shot learning techniques include using Siamese structures [KZS15] and evolutionary methods [MLC16].

Autoencoders are typically used for representation learning in a single dataset, and have only recently been applied to multiple datasets. The recent neural statistician work [ES16] applies the variational autoencoder approach to the encoding and generation of datasets. Compared to their work, our MeLA differs in the following aspects. Firstly, the problem is different. While in neural statistician, each example in the dataset is an instance of a class, in MeLA, we are dealing with datasets whose examples are (\mathbf{x}, \mathbf{y}) pairs, where we don't know *a priori* where the input \mathbf{x} will be in testing time. Therefore, direct autoencoding of datasets is not enough for prediction, especially for regression tasks. Therefore, instead of using autoencoding to generate the *dataset*, our MeLA uses autoencoding to generate the *model* that can generate the dataset given test inputs \mathcal{X} , which is a more compact way to express the relationship between \mathcal{X} and \mathcal{Y} .

7.4 Experiments

Here we examine the core value of MeLA: can it transform a model that is originally intended for single-task learning into one that can quickly adapt to new tasks with few examples without training, and continue to improve with a few gradient steps? The baseline we compare with is a single network pretrained to fit to all tasks, which during testing is fine-tuned to each individual task through further training. MeLA has the same main network architecture \mathbf{f}_θ as this baseline network, supplemented by the meta-recognition and meta-generative models trained via Algorithm 4. We also compare with the state-of-the-art meta-learning algorithm MAML[FAL17], with the same network architecture \mathbf{f}_θ . In addition, we explore the two other MeLA capabilities: influence identification and interactive learning.

For all experiments, the true model \mathbf{h} and its parameters are hidden from all algorithms, except for an *oracle* model which "cheats" by getting access to the true model parameters for each example, thus providing an upper bound on performance. The performance of each algorithm is then evaluated on previously unseen test datasets. For all experiments, the Adam optimizer[KB14] with default parameters is used for training and fine-tuning during

evaluation.¹

7.4.1 Simple regression problem

We first demonstrate the 3 capabilities of MeLA via the same simple regression benchmark previously studied with MAML [FAL17], where the hidden function class is $h(x) = c_1 \sin(x + c_2)$, and the parameters $c_1 \sim U[0.1, 5.0]$, $c_2 \sim U[0, \pi]$ are randomly generated for each dataset. For each dataset, 10 input points x_i are sampled from $U[-5, 5]$ as training examples and another 10 are sampled as testing examples. 100 such datasets are presented for the algorithms during training. The baseline model is a 3-layer network where each hidden layer has 40 neurons with leakyReLU activation.

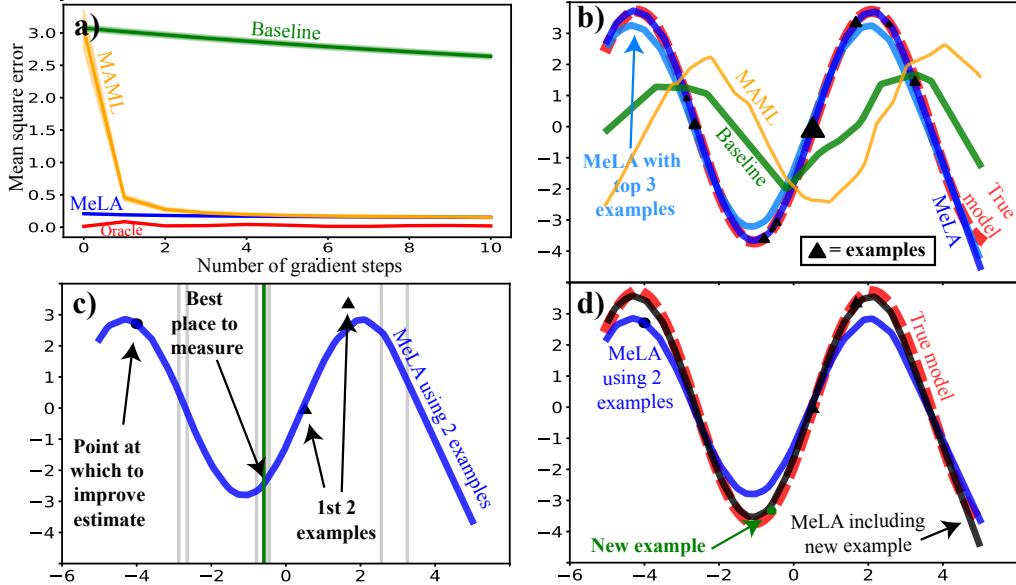


Figure 7-2: (a) MSE vs. number of gradient steps (with learning rate = 0.001, Adam optimizer) on 20,000 randomly sampled testing datasets, for MeLA, MAML, baseline (pretrained) and oracle. MeLA starts at MSE of 0.208 and gets down to 0.129 after 10 steps, while MAML starts at 3.05 and gets down to 0.208 after 5 steps. (b) Predictions after 0 gradient steps for an example test dataset (MAML is after 1 gradient step). The markers' size is proportional to the influence identified by MeLA. Also plotted is MeLA's prediction given only the top 3 influential examples. (c) To get a better prediction at $x^* = -4$ using only two examples at hand, MeLA requests the example at $x = -0.593$ from 8 candidate positions (vertical lines). (d) Improved estimate at $x^* = -4$ after obtaining the requested example.

¹The code for MeLA, the dataset and experiments will be open-sourced upon publication of the paper.

The results are shown in Fig. 7-2. Panel a) plots the mean squared error vs. number of gradient steps on unseen randomly generated testing datasets, showing that MeLA outclasses the baseline model at all stages. It also shows that MeLA asymptotes to the same performance as MAML but learns much faster, starting with a low loss that MAML needs 5 gradient steps to surpass. Panel b) compares predictions with 0 gradient steps. MeLA not only proposes a model that accurately matches the true model, but also identifies each examples' influence on the model generation, and obtains good prediction if only the top 3 influential examples are given. Panels c) and d) show MeLA's capability of actively requesting informative examples by predicting which additional example will help improve the prediction the most.

7.4.2 Ball bouncing with state representation

Next, we test MeLA's capability in simple but challenging physical environments, where it is desirable that an algorithm quickly adapts to each new environment with few observations of states or frames. This is a much harder task than the previous simple "Sin" regression benchmark in few-shot/meta-learning, and is another contribution to the community. Each environment, implemented as a custom Gym environment[BCP⁺16], consists of a room with 4 walls, whose frictionless floor is a random 4-sided convex polygon inside the 2-dimensional unit square $[0, 1] \times [0, 1]$ (Fig. 7-3(a)), and a ball of radius 0.075 that bounces elastically off of these walls and otherwise moves with constant velocity. Because the different room geometries give the ball conflicting bouncing dynamics in different environments, a model trained well in one environment may not necessarily perform well in another, providing an ideal test bed for few-shot/meta-learning. During training, all models take as input 3 consecutive time steps of ball's state (x - and y - coordinates), recorded every time it has moved a distance 0.1. The oracle model is also given as input the coordinates of the floor's 4 corners.

Fig. 7-3 (b) plots the mean Euclidean distance of the models' predictions vs. rollout distance traveled. We can see that MeLA outperforms pretrained and MAML for both 0 and 5 gradient steps. Moreover, what MeLA identifies as influential examples (Fig. 7-4) lies near

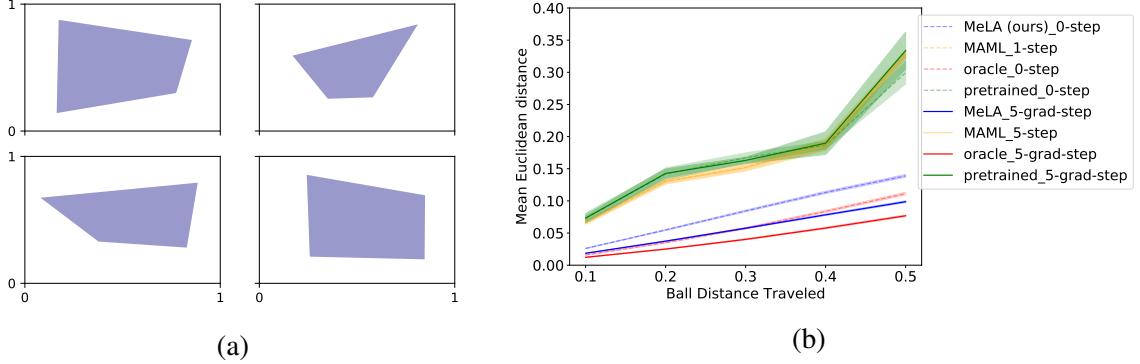


Figure 7-3: (a) Examples of the polygon "bouncy-house" environments. (b) Mean Euclidean distance between target and prediction vs. rollout distance traveled on 1000 randomly generated testing environments.

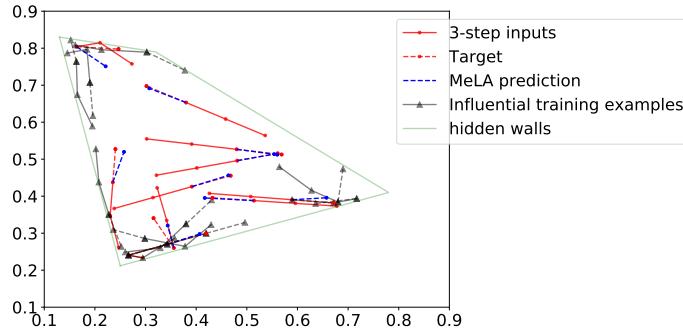


Figure 7-4: Ball bouncing prediction by MeLA for an example testing dataset. Also plotted are the top 10 most influential training trajectories identified by MeLA, which are all near the vertices.

the vertices of the polygon, showing that MeLA essentially learns to capture the convex hull of all the trajectories when proposing the model.

7.4.3 Video prediction

To test MeLA’s ability to integrate into other end-to-end architectures that deal with high-dimensional inputs, we present it with an ensemble of video prediction tasks, each of which has a ball bouncing inside randomly generated polygon walls. The environment setup is the same as in section 7.4.2, except that the inputs are 3 consecutive frames of 39×39 pixel snapshots, and the target is a 39×39 snapshot of the next time step. For all the models, a convolutional autoencoder is used for autoencoding the frames, and the models differ only

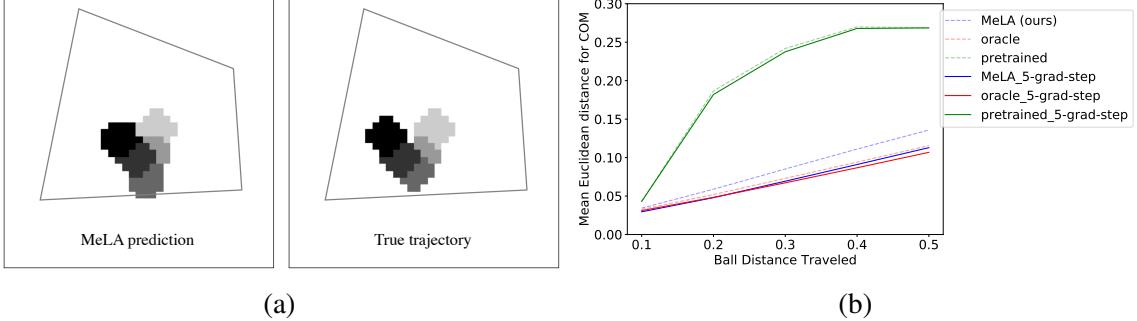


Figure 7-5: (a) Example MeLA prediction vs. true trajectory for 5 rollout steps, with an unseen testing environment without gradient steps. (b) Mean Euclidean distance between the center of mass (COM) of true trajectory and prediction vs. rollout distance traveled for MeLA, pretrained and oracle on 100 randomly generated testing environments.

in the latent dynamics model that predicts the future latent variable based on the 3 steps of latent variables encoded by the autoencoder. For the pretrained model, a single 4-layer network with 40 neurons in each hidden layer is used for the latent dynamics model, training on all tasks. MAML and MeLA also have/generate the same architecture for the latent dynamics model. For the oracle model, the coordinates of the vertices are concatenated with the latent variables as inputs.

Fig. 7-5b plots the mean Euclidean distance of the center of mass of the models' predictions vs. rollout distance. We see that MeLA again greatly reduces the prediction error compared to the baseline model which has to use a single model to predict the trajectory in all environments. MeLA's accuracy is seen to be near that of the oracle, demonstrating that MeLA is learning to quickly recognize and model each environment and propose reasonable models.

7.5 Conclusions

In this paper, we have introduced MeLA, an algorithm for rapid recognition and determination of physical models in few-shot/meta-learning. We demonstrate that MeLA can transform a model intended for single-task learning into one that can quickly adapt with a few examples to a new task. Further, we show that MeLA allows the model to improve with a few gradient steps, for fast few-shot learning. We demonstrate how MeLA learns more

accurately and with fewer examples than both the original model it is based on, and the state-of-the-art meta-learning algorithm MAML. We also demonstrate two additional capabilities of MeLA: its ability to identify influential examples, and how MeLA can interactively request informative examples to optimize learning.

A core enabler of human’s skill to handle novel tasks is our ability to quickly recognize and propose models in new environments, based on previously learned physical models. We believe that by incorporating this ability, machine learning models will become more adaptive and capable for new environments and unsolved problems.

Chapter 8

Rank Pruning for robust learning with noisy labels

$\tilde{P}\tilde{N}$ learning¹ is the problem of binary classification when training examples may be mislabeled (flipped) uniformly with noise rate ρ_1 for positive examples and ρ_0 for negative examples. We propose Rank Pruning (RP) to solve $\tilde{P}\tilde{N}$ learning and the open problem of estimating the noise rates, i.e. the fraction of wrong positive and negative labels. Unlike prior solutions, RP is time-efficient and general, requiring $\mathcal{O}(T)$ for any unrestricted choice of probabilistic classifier with T fitting time. We prove RP has consistent noise estimation and equivalent expected risk as learning with uncorrupted labels in ideal conditions, and derive closed-form solutions when conditions are non-ideal. RP achieves state-of-the-art noise estimation and F1, error, and AUC-PR for both MNIST and CIFAR datasets, regardless of the amount of noise and performs similarly impressively when a large portion of training examples are noise drawn from a third distribution. To highlight, RP with a CNN classifier can predict if an MNIST digit is a *one* or *not* with only 0.25% error, and 0.46% error across all digits, even when 50% of positive examples are mislabeled and 50% of observed positive labels are mislabeled negative examples.

¹The paper “Learning with Confident Examples: Rank Pruning for Robust Classification with Noisy Labels” is published at Conference on Uncertainty in Artificial Intelligence (UAI 2017). Authors: Curtis G. Northcutt*, Tailin Wu*, Isaac L. Chuang, where * denotes equal contributions. arXiv: 1705.01936.

8.1 Introduction

Consider a student with no knowledge of animals tasked with learning to classify whether a picture contains a dog. A teacher shows the student example pictures of lone four-legged animals, stating whether the image contains a dog or not. Unfortunately, the teacher may often make mistakes, asymmetrically, with a significantly large false positive rate, $\rho_1 \in [0, 1]$, and significantly large false negative rate, $\rho_0 \in [0, 1]$. The teacher may also include “white noise” images with a uniformly random label. This information is unknown to the student, who only knows of the images and corrupted labels, but suspects that the teacher may make mistakes. Can the student (1) estimate the mistake rates, ρ_1 and ρ_0 , (2) learn to classify pictures with dogs accurately, and (3) do so efficiently (e.g. less than an hour for 50 images)? This allegory clarifies the challenges of $\tilde{P}\tilde{N}$ learning for any classifier trained with corrupted labels, perhaps with intermixed noise examples. We elect the notation $\tilde{P}\tilde{N}$ to emphasize that both the positive and negative sets may contain mislabeled examples, reserving P and N for uncorrupted sets.

This example illustrates a fundamental reliance of supervised learning on training labels [MCM86]. Traditional learning performance degrades monotonically with label noise [AKA91, NOPF10], necessitating semi-supervised approaches [BLS10]. Examples of noisy datasets are medical [RI96], human-labeled [PCI10], and sensor [LML⁺10] datasets. The problem of uncovering the same classifications as if the data was not mislabeled is our fundamental goal.

Towards this goal, we introduce Rank Pruning², an algorithm for $\tilde{P}\tilde{N}$ learning composed of two sequential parts: (1) estimation of the asymmetric noise rates ρ_1 and ρ_0 and (2) removal of mislabeled examples prior to training. The fundamental mantra of Rank Pruning is *learning with confident examples*, i.e. examples with a predicted probability of being positive *near* 1 when the label is positive or 0 when the label is negative. If we imagine non-confident examples as a noise class, separate from the confident positive and negative classes, then their removal should unveil a subset of the uncorrupted data.

An ancillary mantra of Rank Pruning is *removal by rank* which elegantly exploits ranking

² Rank Pruning is open-source and available at <https://github.com/cgnorthcutt/rankpruning>

without sorting. Instead of pruning non-confident examples by predicted probability, we estimate the number of mislabeled examples in each class. We then remove the k^{th} -most or k^{th} -least examples, *ranked* by predicted probability, via the BFPRT algorithm [BFP⁺73] in $\mathcal{O}(n)$ time, where n is the number of training examples. *Removal by rank* mitigates sensitivity to probability estimation and exploits the reduced complexity of learning to rank over probability estimation [MJV⁺12]. Together, *learning with confident examples* and *removal by rank* enable robustness, i.e. invariance to erroneous input deviation.

Beyond prediction, confident examples help estimate ρ_1 and ρ_0 . Typical approaches require averaging predicted probabilities on a holdout set [LT16c, EN08] tying noise estimation to the accuracy of the predicted probabilities, which in practice may be confounded by added noise or poor model selection. Instead, we estimate ρ_1 and ρ_0 as a fraction of the predicted counts of confident examples in each class, encouraging robustness for variation in probability estimation.

8.1.1 Related Work

Rank Pruning bridges framework, nomenclature, and application across PU and $\tilde{P}\tilde{N}$ learning. In this section, we consider the contributions of Rank Pruning in both.

PU Learning

Positive-unlabeled (PU) learning is a binary classification task in which a subset of positive training examples are labeled, and the rest are unlabeled. For example, co-training [BM98, NG00] with labeled and unlabeled examples can be framed as a PU learning problem by assigning all unlabeled examples the label ‘0’. PU learning methods often assume corrupted negative labels for the unlabeled examples U such that PU learning is $\tilde{P}\tilde{N}$ learning with no mislabeled examples in P , hence their naming conventions.

Early approaches to PU learning modified the loss functions via weighted logistic regression [LL03] and biased SVM [LDL⁺03] to penalize more when positive examples are predicted incorrectly. Bagging SVM [MV14] and RESVM [CSSM15] extended biased SVM to instead

Table 8.1: Variable definitions and descriptions for $\tilde{P}\tilde{N}$ learning and PU learning. Related work contains a prominent author using each variable. ρ_1 is also referred to as *contamination* in PU learning literature.

VARIABLE	CONDITIONAL	DESCRIPTION	DOMAIN	RELATED WORK
ρ_0	$P(s = 1 y = 0)$	FRACTION OF N EXAMPLES MISLABELED AS POSITIVE	$\tilde{P}\tilde{N}$	LIU
ρ_1	$P(s = 0 y = 1)$	FRACTION OF P EXAMPLES MISLABELED AS NEGATIVE	$\tilde{P}\tilde{N}$, PU	LIU, CLAESSEN
π_0	$P(y = 1 s = 0)$	FRACTION OF MISLABELED EXAMPLES IN \tilde{N}	$\tilde{P}\tilde{N}$	SCOTT
π_1	$P(y = 0 s = 1)$	FRACTION OF MISLABELED EXAMPLES IN \tilde{P}	$\tilde{P}\tilde{N}$	SCOTT
$c = 1 - \rho_1$	$P(s = 1 y = 1)$	FRACTION OF CORRECTLY LABELED P IF $P(y = 1 s = 1) = 1$	PU	ELKAN

use an ensemble of classifiers trained by resampling U (and P for RESVM) to improve robustness [Bre96]. RESVM claims state-of-the-art for *PU* learning, but is impractically inefficient for large datasets because it requires optimization of five parameters and suffers from the pitfalls of SVM model selection [CV99]. [EN08] introduce a formative time-efficient probabilistic approach (denoted *Elk08*) for *PU* learning that directly estimates $1 - \rho_1$ by averaging predicted probabilities of a holdout set and dividing all predicted probabilities by $1 - \rho_1$. On the SwissProt database, *Elk08* was 621 times faster than biased SVM, which only requires two parameter optimization. However, *Elk08* noise rate estimation is sensitive to inexact probability estimation and both RESVM and *Elk08* assume $P = \tilde{P}$ and do not generalize to $\tilde{P}\tilde{N}$ learning. Rank Pruning leverages *Elk08* to initialize ρ_1 , but then re-estimates ρ_1 using confident examples for both robustness (RESVM) and efficiency (*Elk08*).

Table 8.2: Summary of state-of-the-art and selected general solutions to $\tilde{P}\tilde{N}$ and *PU* learning.

RELATED WORK	NOISE ESTIM.	$\tilde{P}\tilde{N}$	PU	ANY PROB. CLASSIFIER	PROB ESTIM. ROBUSTNESS	TIME EFFICIENT	THEORY SUPPORT	ADDED NOISE
[EN08]	✓	✓	✓			✓	✓	
[CSSM15]			✓		✓			
[SBH13]	✓			✓	✓		✓	
[NDRT13B]	✓	✓	✓	✓	✓	✓	✓	
[LT16C]	✓	✓	✓	✓		✓	✓	
RANK PRUNING	✓	✓	✓	✓	✓	✓	✓	✓

$\tilde{P}\tilde{N}$ Learning

Theoretical approaches for $\tilde{P}\tilde{N}$ learning often have two steps: (1) estimate the noise rates, ρ_1, ρ_0 , and (2) use ρ_1, ρ_0 for prediction. To our knowledge, Rank Pruning is the only time-efficient solution for the open problem [LT16c, YMJ⁺12] of noise estimation.

We first consider relevant work in noise rate estimation. [SBH13] established a lower bound method for estimating the *inversed* noise rates π_1 and π_0 (defined in Table 8.1). However, the method can be intractable due to unbounded convergence and assumes that the positive and negative distributions are mutually irreducible. Under additional assumptions, [Sco15] proposed a time-efficient method for noise rate estimation, but [LT16c] reported poor performance. [LT16c] used the minimum predicted probabilities as the noise rates, which often yields futile estimates of $\min = 0$. [NDRT13b] provide no method for estimation and view the noise rates as parameters optimized with cross-validation, inducing a sacrificial accuracy, efficiency trade-off. In comparison, Rank Pruning noise rate estimation is time-efficient, consistent in ideal conditions, and robust to imperfect probability estimation.

[NDRT13b] developed two methods for prediction in the $\tilde{P}\tilde{N}$ setting which modify the loss function. The first method constructs an unbiased estimator of the loss function for the true distribution from the noisy distribution, but the estimator may be non-convex even if the original loss function is convex. If the classifier's loss function cannot be modified directly, this method requires splitting each example in two with class-conditional weights and ensuring split examples are in the same batch during optimization. For these reasons, we instead compare Rank Pruning with their second method (*Nat13*), which constructs a label-dependent loss function such that for 0-1 loss, the minimizers of *Nat13*'s risk and the risk for the true distribution are equivalent.

[LT16c] generalized *Elk08* to the $\tilde{P}\tilde{N}$ learning setting by modifying the loss function with per-example importance reweighting (*Liu16*), but reweighting terms are derived from predicted probabilities which may be sensitive to inexact estimation. To mitigate sensitivity, [LT16c] examine the use of density ratio estimation [SSK12]. Instead, Rank Pruning mitigates sensitivity by learning from confident examples selected by rank order, not predicted probability. For fairness of comparison across methods, we compare Rank Pruning with

their probability-based approach.

Assuming perfect estimation of ρ_1 and ρ_0 , we, [NDRT13b], and [LT16c] all prove that the expected risk for the modified loss function is equivalent to the expected risk for the perfectly labeled dataset. However, both [NDRT13b] and [LT16c] effectively "flip" example labels in the construction of their loss function, providing no benefit for added random noise. In comparison, Rank Pruning will also remove added random noise because noise drawn from a third distribution is unlikely to appear confidently positive or negative. Table 8.2 summarizes our comparison of $\tilde{P}\tilde{N}$ and PU learning methods.

Procedural efforts have improved robustness to mislabeling in the context of machine vision [XXY⁺15b], neural networks [RLA⁺15], and face recognition [AAMP05]. Though promising, these methods are restricted in theoretical justification and generality, motivating the need for Rank Pruning.

8.1.2 Contributions

In this paper, we describe the Rank Pruning algorithm for binary classification with imperfectly labeled training data. In particular, we:

- Develop a robust, time-efficient, general solution for both $\tilde{P}\tilde{N}$ learning, i.e. binary classification with noisy labels, and estimation of the fraction of mislabeling in both the positive and negative training sets.
- Introduce the *learning with confident examples* mantra as a new way to think about robust classification and estimation with mislabeled training data.
- Prove that under assumptions, Rank Pruning achieves perfect noise estimation and equivalent expected risk as learning with correct labels. We provide closed-form solutions when those assumptions are relaxed.
- Demonstrate that Rank Pruning performance generalizes across the number of training examples, feature dimension, fraction of mislabeling, and fraction of added noise examples drawn from a third distribution.

- Improve the state-of-the-art of $\tilde{P}\tilde{N}$ learning across F1 score, AUC-PR, and Error. In many cases, Rank Pruning achieves nearly the same F1 score as learning with correct labels when 50% of positive examples are mislabeled and 50% of observed positive labels are mislabeled negative examples.

8.2 Framing the $\tilde{P}\tilde{N}$ Learning Problem

In this section, we formalize the foundational definitions, assumptions, and goals of the $\tilde{P}\tilde{N}$ learning problem illustrated by the student-teacher motivational example.

Given n observed training examples $x \in \mathcal{R}^D$ with associated observed corrupted labels $s \in \{0, 1\}$ and unobserved true labels $y \in \{0, 1\}$, we seek a binary classifier f that estimates the mapping $x \rightarrow y$. Unfortunately, if we fit the classifier using observed (x, s) pairs, we estimate the mapping $x \rightarrow s$ and obtain $g(x) = P(\hat{s} = 1|x)$.

We define the observed noisy positive and negative sets as $\tilde{P} = \{x|s = 1\}$, $\tilde{N} = \{x|s = 0\}$ and the unobserved true positive and negative sets as $P = \{x|y = 1\}$, $N = \{x|y = 0\}$. Define the hidden training data as $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, drawn i.i.d. from some true distribution \mathcal{D} . We assume that a class-conditional Classification Noise Process (CNP) [AL88] maps y true labels to s observed labels such that each label in P is flipped independently with probability ρ_1 and each label in N is flipped independently with probability ρ_0 ($s \leftarrow CNP(y, \rho_1, \rho_0)$). The resulting observed, corrupted dataset is $D_\rho = \{(x_1, s_1), (x_2, s_2), \dots, (x_n, s_n)\}$. Therefore, $(s \perp\!\!\!\perp x)|y$ and $P(s = s|y = y, x) = P(s = s|y = y)$. In recent work, CNP is referred to as the random noise classification (RCN) noise model [LT16c, NDRT13b].

The noise rate $\rho_1 = P(s = 0|y = 1)$ is the fraction of P examples mislabeled as negative and the noise rate $\rho_0 = P(s = 1|y = 0)$ is the fraction of N examples mislabeled as positive. Note that $\rho_1 + \rho_0 < 1$ is a necessary condition, otherwise more examples would be mislabeled than labeled correctly. Thus, $\rho_0 < 1 - \rho_1$. We elect a subscript of “0” to refer to the negative set and a subscript of “1” to refer to the positive set. Additionally, let $p_{s1} = P(s = 1)$ be the fraction of corrupted labels that are positive and $p_{y1} = P(y = 1)$

be the fraction of true labels that are positive. It follows that the inversed noise rates are $\pi_1 = P(y = 0|s = 1) = \frac{\rho_0(1-p_{y1})}{p_{s1}}$ and $\pi_0 = P(y = 1|s = 0) = \frac{\rho_1 p_{y1}}{(1-p_{s1})}$. Combining these relations, given any pair in $\{(\rho_0, \rho_1), (\rho_1, \pi_1), (\rho_0, \pi_0), (\pi_0, \pi_1)\}$, the remaining two and p_{y1} are known.

We consider five levels of assumptions for P , N , and g :

Perfect Condition: g is a “perfect” probability estimator iff $g(x) = g^*(x)$ where $g^*(x) = P(s = 1|x)$. Equivalently, let $g(x) = P(s = 1|x) + \Delta g(x)$. Then $g(x)$ is “perfect” when $\Delta g(x) = 0$ and “imperfect” when $\Delta g(x) \neq 0$. g may be imperfect due to the method of estimation or due to added uniformly randomly labeled examples drawn from a third noise distribution.

Non-overlapping Condition: P and N have “non-overlapping support” if $P(y = 1|x) = \mathbb{1}[[y = 1]]$, where the indicator function $\mathbb{1}[[a]]$ is 1 if the a is true, else 0.

Ideal Condition³: g is “ideal” when both perfect and non-overlapping conditions hold and $(s \perp\!\!\!\perp x)|y$ such that

$$\begin{aligned} g(x) &= g^*(x) = P(s = 1|x) \\ &= P(s = 1|y = 1, x) \cdot P(y = 1|x) + P(s = 1|y = 0, x) \cdot P(y = 0|x) \\ &= (1 - \rho_1) \cdot \mathbb{1}[[y = 1]] + \rho_0 \cdot \mathbb{1}[[y = 0]] \end{aligned} \quad (8.1)$$

Range Separability Condition g range separates P and N iff $\forall x_1 \in P$ and $\forall x_2 \in N$, we have $g(x_1) > g(x_2)$.

Unasssuming Condition: g is “unassuming” when perfect and/or non-overlapping conditions may not be true.

Their relationship is: **Unasssuming** \supset **Range Separability** \supset **Ideal** = **Perfect** \cap **Non-overlapping**.

We can now state the two goals of Rank Pruning for $\tilde{P}\tilde{N}$ learning. **Goal 1** is to perfectly estimate $\hat{\rho}_1 \triangleq \rho_1$ and $\hat{\rho}_0 \triangleq \rho_0$ when g is ideal. When g is not ideal, to our knowledge perfect estimation of ρ_1 and ρ_0 is impossible and at best **Goal 1** is to provide exact expressions for $\hat{\rho}_1$ and $\hat{\rho}_0$ w.r.t. ρ_1 and ρ_0 . **Goal 2** is to use $\hat{\rho}_1$ and $\hat{\rho}_0$ to uncover the classifications of f from

³ Eq. (8.1) is first derived in [EN08].

g . Both tasks must be accomplished given only observed (x, s) pairs. y, ρ_1, ρ_0, π_1 , and π_0 are hidden.

8.3 Rank Pruning

We develop the Rank Pruning algorithm to address our two goals. In Section 8.3.1, we propose a method for noise rate estimation and prove consistency when g is ideal. An estimator is “consistent” if it achieves perfect estimation in the expectation of infinite examples. In Section 8.3.2, we derive exact expressions for $\hat{\rho}_1$ and $\hat{\rho}_0$ when g is unassuming. In Section 8.3.3, we provide the entire algorithm, and in Section 8.3.5, prove that Rank Pruning has equivalent expected risk as learning with uncorrupted labels for both ideal g and non-ideal g with weaker assumptions. Throughout, we assume $n \rightarrow \infty$ so that P and N are the hidden distributions, each with infinite examples. This is a necessary condition for Theorems. 10, 11 and Lemmas 9.1, 10.1.

8.3.1 Deriving Noise Rate Estimators $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$

We propose the *confident counts* estimators $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$ to estimate ρ_1 and ρ_0 as a fraction of the predicted counts of confident examples in each class, encouraging robustness for variation in probability estimation. To estimate $\rho_1 = P(s = 0|y = 1)$, we count the number of examples with label $s = 0$ that we are “confident” have label $y = 1$ and divide it by the total number of examples that we are “confident” have label $y = 1$. More formally,

$$\hat{\rho}_1^{conf} := \frac{|\tilde{N}_{y=1}|}{|\tilde{N}_{y=1}| + |\tilde{P}_{y=1}|}, \quad \hat{\rho}_0^{conf} := \frac{|\tilde{P}_{y=0}|}{|\tilde{P}_{y=0}| + |\tilde{N}_{y=0}|} \quad (8.2)$$

such that

$$\begin{cases} \tilde{P}_{y=1} = \{x \in \tilde{P} \mid g(x) \geq LB_{y=1}\} \\ \tilde{N}_{y=1} = \{x \in \tilde{N} \mid g(x) \geq LB_{y=1}\} \\ \tilde{P}_{y=0} = \{x \in \tilde{P} \mid g(x) \leq UB_{y=0}\} \\ \tilde{N}_{y=0} = \{x \in \tilde{N} \mid g(x) \leq UB_{y=0}\} \end{cases} \quad (8.3)$$

where g is fit to the corrupted training set D_ρ to obtain $g(x) = P(\hat{s} = 1|x)$. The threshold $LB_{y=1}$ is the predicted probability in $g(x)$ above which we guess that an example x has hidden label $y = 1$, and similarly for upper bound $UB_{y=0}$. $LB_{y=1}$ and $UB_{y=0}$ partition \tilde{P} and \tilde{N} into four sets representing a *best guess* of a *subset* of examples having labels (1) $s = 1, y = 1$, (2) $s = 1, y = 0$, (3) $s = 0, y = 1$, (4) $s = 0, y = 0$. The threshold values are defined as

$$\begin{cases} LB_{y=1} := P(\hat{s} = 1 \mid s = 1) = E_{x \in \tilde{P}}[g(x)] \\ UB_{y=0} := P(\hat{s} = 1 \mid s = 0) = E_{x \in \tilde{N}}[g(x)] \end{cases}$$

where \hat{s} is the predicted label from a classifier fit to the observed data. $|\tilde{P}_{y=1}|$ counts examples with label $s = 1$ that are *most* likely to be correctly labeled ($y = 1$) because $LB_{y=1} = P(\hat{s} = 1|s = 1)$. The three other terms in Eq. (8.3) follow similar reasoning. Importantly, the four terms do not sum to n , i.e. $|N| + |P|$, but $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$ are valid estimates because mislabeling noise is assumed to be uniformly random. The choice of threshold values relies on the following two important equations:

$$\begin{aligned} LB_{y=1} &= E_{x \in \tilde{P}}[g(x)] = E_{x \in \tilde{P}}[P(s = 1|x)] \\ &= E_{x \in \tilde{P}}[P(s = 1|x, y = 1)P(y = 1|x) + P(s = 1|x, y = 0)P(y = 0|x)] \\ &= E_{x \in \tilde{P}}[P(s = 1|y = 1)P(y = 1|x) + P(s = 1|y = 0)P(y = 0|x)] \\ &= (1 - \rho_1)(1 - \pi_1) + \rho_0\pi_1 \end{aligned} \quad (8.4)$$

Similarly, we have

$$UB_{y=0} = (1 - \rho_1)\pi_0 + \rho_0(1 - \pi_0) \quad (8.5)$$

To our knowledge, although simple, this is the first time that the relationship in Eq. (8.4) (8.5) has been published, linking the work of [EN08], [LT16c], [SBH13] and [NDRT13b]. From Eq. (8.4) (8.5), we observe that $LB_{y=1}$ and $UB_{y=0}$ are linear interpolations of $1 - \rho_1$ and ρ_0 and since $\rho_0 < 1 - \rho_1$, we have that $\rho_0 < LB_{y=1} \leq 1 - \rho_1$ and $\rho_0 \leq UB_{y=0} < 1 - \rho_1$. When g is ideal we have that $g(x) = (1 - \rho_1)$, if $x \in P$ and $g(x) = \rho_0$, if $x \in N$. Thus when g is ideal, the thresholds $LB_{y=1}$ and $UB_{y=0}$ in Eq. (8.3) will perfectly separate P and N examples within each of \tilde{P} and \tilde{N} . Lemma 9.1 immediately follows.

Lemma 9.1. *When g is ideal,*

$$\begin{aligned}\tilde{P}_{y=1} &= \{x \in P \mid s = 1\}, & \tilde{N}_{y=1} &= \{x \in P \mid s = 0\}, \\ \tilde{P}_{y=0} &= \{x \in N \mid s = 1\}, & \tilde{N}_{y=0} &= \{x \in N \mid s = 0\}\end{aligned}\tag{8.6}$$

Thus, when g is ideal, the thresholds in Eq. (8.3) partition the training set such that $\tilde{P}_{y=1}$ and $\tilde{N}_{y=0}$ contain the correctly labeled examples and $\tilde{P}_{y=0}$ and $\tilde{N}_{y=1}$ contain the mislabeled examples. Theorem 10 follows (for brevity, proofs of all theorems/lemmas are in Appendix A.7.1-A.7.1).

Theorem 10. *When g is ideal,*

$$\hat{\rho}_1^{conf} = \rho_1, \quad \hat{\rho}_0^{conf} = \rho_0\tag{8.7}$$

Thus, when g is ideal, the *confident counts* estimators $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$ are consistent estimators for ρ_1 and ρ_0 and we set $\hat{\rho}_1 := \hat{\rho}_1^{conf}$, $\hat{\rho}_0 := \hat{\rho}_0^{conf}$. These steps comprise Rank Pruning noise rate estimation (see Alg. 5). There are two practical observations. First, for any g with T fitting time, computing $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$ is $\mathcal{O}(T)$. Second, $\hat{\rho}_1$ and $\hat{\rho}_0$ should be estimated out-of-sample to avoid over-fitting, resulting in sample variations. In our experiments, we use 3-fold cross-validation, requiring at most $2T = \mathcal{O}(T)$.

8.3.2 Noise Estimation: Unasssuming Case

Theorem 10 states that $\hat{\rho}_i^{conf} = \rho_i$, $\forall i \in \{0, 1\}$ when g is ideal. Though theoretically constructive, in practice this is unlikely. Next, we derive expressions for the estimators when

g is unassuming, i.e. g may not be perfect and P and N may have overlapping support.

Define $\Delta p_o := \frac{|P \cap N|}{|P \cup N|}$ as the fraction of overlapping examples in \mathcal{D} and remember that $\Delta g(x) := g(x) - g^*(x)$. Denote $LB_{y=1}^* = (1 - \rho_1)(1 - \pi_1) + \rho_0\pi_1$, $UB_{y=0}^* = (1 - \rho_1)\pi_0 + \rho_0(1 - \pi_0)$. We have

Lemma 10.1. *When g is unassuming, we have*

$$\begin{cases} LB_{y=1} = LB_{y=1}^* + E_{x \in \tilde{P}}[\Delta g(x)] - \frac{(1-\rho_1-\rho_0)^2}{p_{s1}}\Delta p_o \\ UB_{y=0} = UB_{y=0}^* + E_{x \in \tilde{N}}[\Delta g(x)] + \frac{(1-\rho_1-\rho_0)^2}{1-p_{s1}}\Delta p_o \\ \hat{\rho}_1^{conf} = \rho_1 + \frac{1-\rho_1-\rho_0}{|P|-|\Delta P_1|+|\Delta N_1|}|\Delta N_1| \\ \hat{\rho}_0^{conf} = \rho_0 + \frac{1-\rho_1-\rho_0}{|N|-|\Delta N_0|+|\Delta P_0|}|\Delta P_0| \end{cases} \quad (8.8)$$

where

$$\begin{cases} \Delta P_1 = \{x \in P \mid g(x) < LB_{y=1}\} \\ \Delta N_1 = \{x \in N \mid g(x) \geq LB_{y=1}\} \\ \Delta P_0 = \{x \in P \mid g(x) \leq UB_{y=0}\} \\ \Delta N_0 = \{x \in N \mid g(x) > UB_{y=0}\} \end{cases}$$

The second term on the R.H.S. of the $\hat{\rho}_i^{conf}$ expressions captures the deviation of $\hat{\rho}_i^{conf}$ from ρ_i , $i = 0, 1$. This term results from both imperfect $g(x)$ and overlapping support. Because the term is non-negative, $\hat{\rho}_i^{conf} \geq \rho_i$, $i = 0, 1$ in the limit of infinite examples. In other words, $\hat{\rho}_i^{conf}$ is an *upper bound* for the noise rates ρ_i , $i = 0, 1$. From Lemma 10.1, it also follows:

Theorem 11. *Given non-overlapping support condition,*

If $\forall x \in N, \Delta g(x) < LB_{y=1} - \rho_0$, then $\hat{\rho}_1^{conf} = \rho_1$.

If $\forall x \in P, \Delta g(x) > -(1 - \rho_1 - UB_{y=0})$, then $\hat{\rho}_0^{conf} = \rho_0$.

Theorem 11 shows that $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$ are robust to imperfect probability estimation. As long as $\Delta g(x)$ does not exceed the distance between the threshold in Eq. (8.3) and the

perfect $g^*(x)$ value, $\hat{\rho}_1^{conf}$ and $\hat{\rho}_0^{conf}$ are consistent estimators for ρ_1 and ρ_0 . Our numerical experiments in Section 8.4 suggest this is reasonable for $\Delta g(x)$. The average $|\Delta g(x)|$ for the MNIST training dataset across different (ρ_1, π_1) varies between 0.01 and 0.08 for a logistic regression classifier, 0.01~0.03 for a CNN classifier, and 0.05~0.10 for the CIFAR dataset with a CNN classifier. Thus, when $LB_{y=1} - \rho_0$ and $1 - \rho_1 - UB_{y=0}$ are above 0.1 for these datasets, from Theorem 11 we see that $\hat{\rho}_i^{conf}$ still accurately estimates ρ_i .

8.3.3 The Rank Pruning Algorithm

Using $\hat{\rho}_1$ and $\hat{\rho}_0$, we must uncover the classifications of f from g . In this section, we describe how Rank Pruning selects confident examples, removes the rest, and trains on the pruned set using a reweighted loss function.

First, we obtain the inverse noise rates $\hat{\pi}_1, \hat{\pi}_0$ from $\hat{\rho}_1, \hat{\rho}_0$:

$$\hat{\pi}_1 = \frac{\hat{\rho}_0}{p_{s1}} \frac{1 - p_{s1} - \hat{\rho}_1}{1 - \hat{\rho}_1 - \hat{\rho}_0}, \quad \hat{\pi}_0 = \frac{\hat{\rho}_1}{1 - p_{s1}} \frac{p_{s1} - \hat{\rho}_0}{1 - \hat{\rho}_1 - \hat{\rho}_0} \quad (8.9)$$

Next, we prune the $\hat{\pi}_1|\tilde{P}|$ examples in \tilde{P} with smallest $g(x)$ and the $\hat{\pi}_0|\tilde{N}|$ examples in \tilde{N} with highest $g(x)$ and denote the pruned sets \tilde{P}_{conf} and \tilde{N}_{conf} . To prune, we define k_1 as the $(\hat{\pi}_1|\tilde{P}|)^{th}$ smallest $g(x)$ for $x \in \tilde{P}$ and k_0 as the $(\hat{\pi}_0|\tilde{N}|)^{th}$ largest $g(x)$ for $x \in \tilde{N}$. BFPRT ($\mathcal{O}(n)$) [BFP⁺73] is used to compute k_1 and k_0 and pruning is reduced to the following $\mathcal{O}(n)$ filter:

$$\tilde{P}_{conf} := \{x \in \tilde{P} \mid g(x) \geq k_1\}, \quad \tilde{N}_{conf} := \{x \in \tilde{N} \mid g(x) \leq k_0\} \quad (8.10)$$

Lastly, we refit the classifier to $X_{conf} = \tilde{P}_{conf} \cup \tilde{N}_{conf}$ by class-conditionally reweighting the loss function for examples in \tilde{P}_{conf} with weight $\frac{1}{1 - \hat{\rho}_1}$ and examples in \tilde{N}_{conf} with weight $\frac{1}{1 - \hat{\rho}_0}$ to recover the estimated balance of positive and negative examples. The entire Rank Pruning algorithm is presented in Alg. 5 and illustrated step-by-step on a synthetic dataset in Fig. 8-1.

We conclude this section with a formal discussion of the loss function and efficiency of Rank Pruning. Define \hat{y}_i as the predicted label of example i for the classifier fit to X_{conf}, s_{conf} and

let $l(\hat{y}_i, s_i)$ be the original loss function for $x_i \in D_\rho$. Then the loss function for Rank Pruning is simply the original loss function exerted on the pruned X_{conf} , with class-conditional weighting:

$$\tilde{l}(\hat{y}_i, s_i) = \frac{1}{1 - \hat{\rho}_1} l(\hat{y}_i, s_i) \cdot \mathbb{1}[[x_i \in \tilde{P}_{conf}]] + \frac{1}{1 - \hat{\rho}_0} l(\hat{y}_i, s_i) \cdot \mathbb{1}[[x_i \in \tilde{N}_{conf}]] \quad (8.11)$$

Effectively this loss function uses a zero-weight for pruned examples. Other than potentially fewer examples, the only difference in the loss function for Rank Pruning and the original loss function is the class-conditional weights. These constant factors do not increase the complexity of the minimization of the original loss function. In other words, we can fairly report the running time of Rank Pruning in terms of the running time ($\mathcal{O}(T)$) of the choice of probabilistic estimator. Combining noise estimation ($\mathcal{O}(T)$), pruning ($\mathcal{O}(n)$), and the final fitting ($\mathcal{O}(T)$), Rank Pruning has a running time of $\mathcal{O}(T) + \mathcal{O}(n)$, which is $\mathcal{O}(T)$ for typical classifiers.

8.3.4 Rank Pruning: A simple summary

Recognizing that formalization can create obfuscation, in this section we describe the entire algorithm in a few sentences. Rank Pruning takes as input training examples X , noisy labels s , and a probabilistic classifier clf and finds a subset of X, s that is likely to be correctly labeled, i.e. a subset of X, y . To do this, we first find two thresholds, $LB_{y=1}$ and $UB_{y=0}$, to *confidently* guess the correctly and incorrectly labeled examples in each of \tilde{P} and \tilde{N} , forming four sets, then use the set sizes to estimate the noise rates $\rho_1 = P(s = 0|y = 1)$ and $\rho_0 = P(s = 1|y = 0)$. We then use the noise rates to estimate the number of examples with observed label $s = 1$ and hidden label $y = 0$ and remove that number of examples from \tilde{P} by removing those with lowest predicted probability $g(x)$. We prune \tilde{N} similarly. Finally, the classifier is fit to the pruned set, which is intended to represent a subset of the correctly labeled data.

Algorithm 5 Rank Pruning

Input: Examples X , corrupted labels s , classifier clf

Part 1. Estimating Noise Rates:

(1.1) $\text{clf.fit}(X, s)$

$$g(x) \leftarrow \text{clf.predict_crossval_probability}(\hat{s} = 1|x)$$

$$p_{s1} = \frac{\text{count}(s=1)}{\text{count}(s=0 \vee s=1)}$$

$$LB_{y=1} = E_{x \in \tilde{P}}[g(x)], UB_{y=0} = E_{x \in \tilde{N}}[g(x)]$$

$$(1.2) \hat{\rho}_1 = \hat{\rho}_1^{\text{conf}} = \frac{|\tilde{N}_{y=1}|}{|\tilde{N}_{y=1}| + |\tilde{P}_{y=1}|}, \hat{\rho}_0 = \hat{\rho}_0^{\text{conf}} = \frac{|\tilde{P}_{y=0}|}{|\tilde{P}_{y=0}| + |\tilde{N}_{y=0}|}$$

$$\hat{\pi}_1 = \frac{\hat{\rho}_0}{p_{s1}} \frac{1-p_{s1}-\hat{\rho}_1}{1-\hat{\rho}_1-\hat{\rho}_0}, \hat{\pi}_0 = \frac{\hat{\rho}_1}{1-p_{s1}} \frac{p_{s1}-\hat{\rho}_0}{1-\hat{\rho}_1-\hat{\rho}_0}$$

Part 2. Prune Inconsistent Examples:

(2.1) Remove $\hat{\pi}_1|\tilde{P}|$ examples in \tilde{P} with least $g(x)$, Remove $\hat{\pi}_0|\tilde{N}|$ examples in \tilde{N} with greatest $g(x)$,

Denote the remaining training set $(X_{\text{conf}}, s_{\text{conf}})$

(2.2) $\text{clf.fit}(X_{\text{conf}}, s_{\text{conf}})$, with sample weight $w(x) = \frac{1}{1-\hat{\rho}_1} \mathbb{1}[[s_{\text{conf}} = 1]] + \frac{1}{1-\hat{\rho}_0} \mathbb{1}[[s_{\text{conf}} = 0]]$

Output: clf

8.3.5 Expected Risk Evaluation

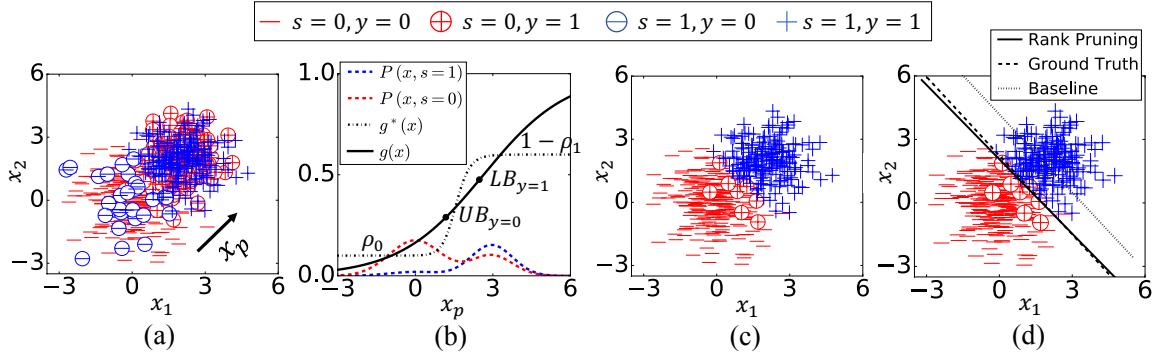
In this section, we prove Rank Pruning exactly uncovers the classifier f fit to hidden y labels when g range separates P and N and ρ_1 and ρ_0 are given.

Denote $f_\theta \in \mathcal{F} : x \rightarrow \hat{y}$ as a classifier's prediction function belonging to some function space \mathcal{F} , where θ represents the classifier's parameters. f_θ represents f , but without θ necessarily fit to the training data. \hat{f} is the Rank Pruning estimate of f .

Denote the empirical risk of f_θ w.r.t. the loss function \tilde{l} and corrupted data D_ρ as $\hat{R}_{\tilde{l}, D_\rho}(f_\theta) = \frac{1}{n} \sum_{i=1}^n \tilde{l}(f_\theta(x_i), s_i)$, and the expected risk of f_θ w.r.t. the corrupted distribution \mathcal{D}_ρ as $R_{\tilde{l}, \mathcal{D}_\rho}(f_\theta) = E_{(x, s) \sim \mathcal{D}_\rho}[\hat{R}_{\tilde{l}, D_\rho}(f_\theta)]$. Similarly, denote $R_{l, \mathcal{D}}(f_\theta)$ as the expected risk of f_θ w.r.t. the hidden distribution \mathcal{D} and loss function l . We show that using Rank Pruning, a classifier \hat{f} can be learned for the hidden data D , given the corrupted data D_ρ , by minimizing the empirical risk:

$$\hat{f} = \arg \min_{f_\theta \in \mathcal{F}} \hat{R}_{\tilde{l}, D_\rho}(f_\theta) = \arg \min_{f_\theta \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \tilde{l}(f_\theta(x_i), s_i) \quad (8.12)$$

Under the *range separability* condition, we have



Theorem 12. If g range separates P and N and $\hat{\rho}_i = \rho_i$, $i = 0, 1$, then for any classifier f_θ and any bounded loss function $l(\hat{y}_i, y_i)$, we have

$$R_{\tilde{l}, \mathcal{D}_\rho}(f_\theta) = R_{l, \mathcal{D}}(f_\theta) \quad (8.13)$$

where $\tilde{l}(\hat{y}_i, s_i)$ is Rank Pruning's loss function (Eq. 8.11).

The proof of Theorem 12 is in Appendix A.7.1. Intuitively, Theorem 12 tells us that if g range separates P and N , then given exact noise rate estimates, Rank Pruning will exactly prune out the positive examples in \tilde{N} and negative examples in \tilde{P} , leading to the same expected risk as learning from uncorrupted labels. Thus, Rank Pruning can exactly uncover the classifications of f (with infinite examples) because the expected risk is equivalent for any f_θ . Note Theorem 12 also holds when g is ideal, since *ideal* \subset *range separability*. In practice, *range separability* encompasses a wide range of imperfect $g(x)$ scenarios, e.g. $g(x)$ can have large fluctuation in both P and N or have systematic drift w.r.t. to $g^*(x)$ due to underfitting.

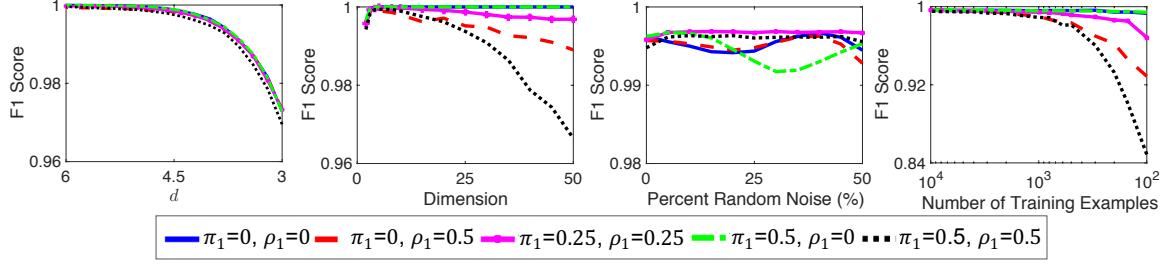


Figure 8-2: Comparison of Rank Pruning with different noise ratios (π_1, ρ_1) on a synthetic dataset for varying separability d , dimension, added random noise and number of training examples. Default settings for Fig. 8-2, 8-3 and 8-4: $d = 4$, 2-dimension, 0% random noise, and 5000 training examples with $p_{y1} = 0.2$. The lines are an average of 200 trials.

8.4 Experimental Results

In Section 8.3, we developed a theoretical framework for Rank Pruning, proved exact noise estimation and equivalent expected risk when conditions are ideal, and derived closed-form solutions when conditions are non-ideal. Our theory suggests that, in practice, Rank Pruning should (1) accurately estimate ρ_1 and ρ_0 , (2) typically achieve as good or better F1, error and AUC-PR [DG06b] as state-of-the-art methods, and (3) be robust to both mislabeling and added noise.

In this section, we support these claims with an evaluation of the comparative performance of Rank Pruning in non-ideal conditions across thousands of scenarios. These include less complex (MNIST) and more complex (CIFAR) datasets, simple (logistic regression) and complex (CNN) classifiers, the range of noise rates, added random noise, separability of P and N , input dimension, and number of training examples to ensure that Rank Pruning is a general, agnostic solution for $\tilde{P}\tilde{N}$ learning.

In our experiments, we adjust π_1 instead of ρ_0 because binary noisy classification problems (e.g. detection and recognition tasks) often have that $|P| \ll |N|$. This choice allows us to adjust both noise rates with respect to P , i.e. the fraction of true positive examples that are mislabeled as negative (ρ_1) and the fraction of observed positive labels that are actually mislabeled negative examples (π_1). The $\tilde{P}\tilde{N}$ learning algorithms are trained with corrupted labels s , and tested on an unseen test set by comparing predictions \hat{y} with the true test labels y using F1 score, error, and AUC-PR metrics. We include all three to emphasize our apathy

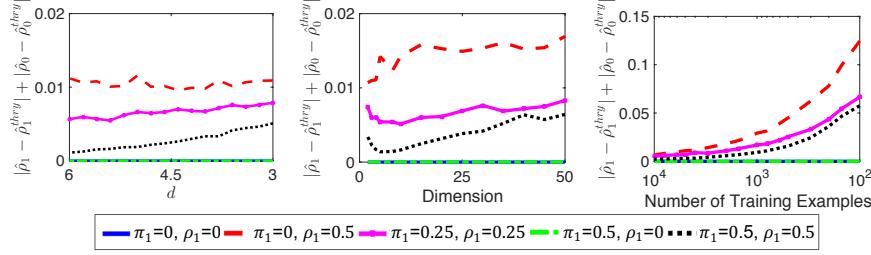


Figure 8-3: Sum of absolute difference between theoretically estimated $\hat{\rho}_i^{thry}$ and empirical $\hat{\rho}_i$, $i = 0, 1$, with five different (π_1, ρ_1) , for varying separability d , dimension, and number of training examples. Note that no figure exists for percent random noise because the theoretical estimates in Eq. (8.8) do not address added noise examples.

toward tuning results to any single metric. We provide F1 scores in this section with error and AUC-PR scores in Appendix A.7.3.

8.4.1 Synthetic Dataset

The synthetic dataset is comprised of a Gaussian positive class and a Gaussian negative classes such that negative examples ($y = 0$) obey an m -dimensional Gaussian distribution $N(\mathbf{0}, \mathbf{I})$ with unit variance $\mathbf{I} = diag(1, 1, \dots, 1)$, and positive examples obey $N(d\mathbf{1}, 0.8\mathbf{I})$, where $d\mathbf{1} = (d, d, \dots, d)$ is an m -dimensional vector, and d measures the separability of the positive and negative set.

We test Rank Pruning by varying 4 different settings of the environment: separability d , dimension, number of training examples n , and percent (of n) added random noise drawn from a uniform distribution $U([-10, 10]^m)$. In each scenario, we test 5 different (π_1, ρ_1) pairs: $(\pi_1, \rho_1) \in \{(0, 0), (0, 0.5), (0.25, 0.25), (0.5, 0), (0.5, 0.5)\}$. From Fig. 8-2, we observe that across these settings, the F1 score for Rank Pruning is fairly agnostic to magnitude of mislabeling (noise rates). As a validation step, in Fig. 8-3 we measure how closely our empirical estimates match our theoretical solutions in Eq. (8.8) and find near equivalence except when the number of training examples approaches zero.

For significant mislabeling ($\rho_1 = 0.5, \pi_1 = 0.5$), Rank Pruning often outperforms other methods (Fig. 8-4). In the scenario of different separability d , it achieves nearly the same F1 score as the ground truth classifier. Remarkably, from Fig. 8-2 and Fig. 8-4, we observe

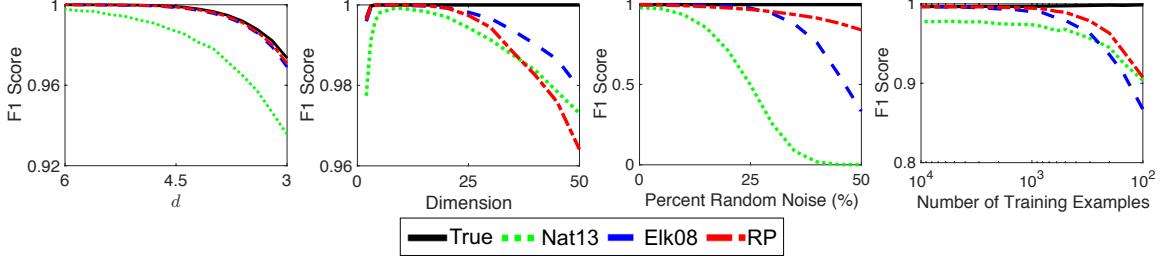


Figure 8-4: Comparison of $\tilde{P}\tilde{N}$ methods for varying separability d , dimension, added random noise, and number of training examples for $\pi_1 = 0.5$, $\rho_1 = 0.5$ (given to all methods).

that when added random noise comprises 50% of total training examples, Rank Pruning still achieves $F1 > 0.85$, compared with $F1 < 0.5$ for all other methods. This emphasizes a unique feature of Rank Pruning, it will also remove added random noise because noise drawn from a third distribution is unlikely to appear confidently positive or negative.

8.4.2 MNIST and CIFAR Datasets

We consider the binary classification tasks of one-vs-rest for the MNIST [LC10] and CIFAR-10 ([KNH]) datasets, e.g. the “car vs rest” task in CIFAR is to predict if an image is a “car” or “not”. ρ_1 and π_1 are given to all $\tilde{P}\tilde{N}$ learning methods for fair comparison, except for RP_ρ which is Rank Pruning including noise rate estimation. RP_ρ metrics measure our performance on the unadulterated $\tilde{P}\tilde{N}$ learning problem.

As evidence that Rank Pruning is dataset and classifier agnostic, we demonstrate its superiority with both (1) a linear logistic regression model with unit L2 regularization and (2) an AlexNet CNN variant with max pooling and dropout, modified to have a two-class output. The CNN structure is adapted from [Cho16b] for MNIST and [Cho16a] for CIFAR. CNN training ends when a 10% holdout set shows no loss decrease for 10 epochs (max 50 for MNIST and 150 for CIFAR).

We consider noise rates $\pi_1, \rho_1 \in \{(0, 0.5), (0.25, 0.25), (0.5, 0), (0.5, 0.5)\}$ for both MNIST and CIFAR, with additional settings for MNIST in Table 8.3 to emphasize Rank Pruning performance is noise rate agnostic. The $\rho_1 = 0, \pi_1 = 0$ case is omitted because when given

ρ_1, π_1 , all methods have the same loss function as the ground truth classifier, resulting in nearly identical F1 scores. Note that in general, Rank Pruning does not require perfect probability estimation to achieve perfect F1-score. As an example, this occurs when P and N are range-separable, and the rank order of the sorted $g(x)$ probabilities in P and N is consistent with the rank of the perfect probabilities, regardless of the actual values of $g(x)$.

For MNIST using logistic regression, we evaluate the consistency of our noise rate estimates with actual noise rates and theoretical estimates (Eq. 8.8) across $\pi_1 \in [0, 0.8] \times \rho_1 \in [0, 0.9]$. The computing time for one setting was ~ 10 minutes on a single CPU core. The results for $\hat{\rho}_1$ and $\hat{\pi}_1$ (Fig. 8-5) are satisfactorily consistent, with mean absolute difference $MD_{\hat{\rho}_1, \rho_1} = 0.105$ and $MD_{\hat{\pi}_1, \pi_1} = 0.062$, and validate our theoretical solutions ($MD_{\hat{\rho}_1, \hat{\rho}_1^{thry}} = 0.0028$, $MD_{\hat{\pi}_1, \hat{\pi}_1^{thry}} = 0.0058$). The deviation of the theoretical and empirical estimates reflects the assumption that we have infinite examples, whereas empirically, the number of examples is finite.

We emphasize two observations from our analysis on CIFAR and MNIST. First, Rank Pruning performs well in nearly every scenario and boasts the most dramatic improvement over prior state-of-the-art in the presence of extreme noise ($\pi_1 = 0.5, \rho_1 = 0.5$). This is easily observed in the right-most quadrant of Table 8.4. The $\pi_1 = 0.5, \rho_1 = 0$ quadrant is

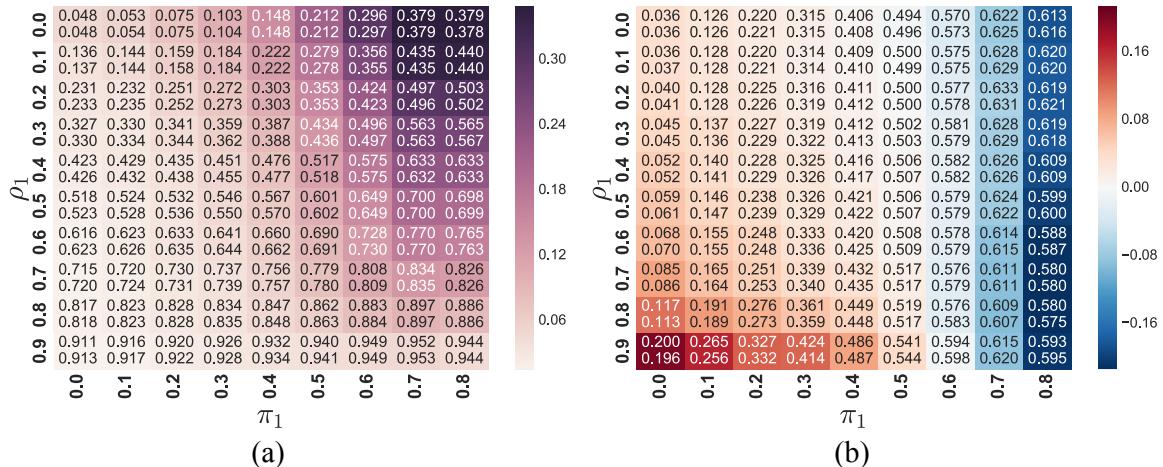


Figure 8-5: Rank Pruning $\hat{\rho}_1$ and $\hat{\pi}_1$ estimation consistency, averaged over all digits in MNIST. **(a)** Color depicts $\hat{\rho}_1 - \rho_1$ with $\hat{\rho}_1$ (upper) and theoretical $\hat{\rho}_1^{thry}$ (lower) in each block. **(b)** Color depicts $\hat{\pi}_1 - \pi_1$ with $\hat{\pi}_1$ (upper) and $\hat{\pi}_1^{thry}$ (lower) in each block.

Table 8.3: Comparison of F1 score for one-vs-rest MNIST and CIFAR-10 (averaged over all digits/images) using logistic regression. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if greater than non-RP models. Due to sensitivity to imperfect $g(x)$, *Liu16* often predicts the same label for all examples.

DATASET MODEL, $\rho_1 =$	CIFAR				MNIST																		
	0.0	0.25	0.5	0.5	$\pi_1 = 0.0$				$\pi_1 = 0.25$				$\pi_1 = 0.5$				$\pi_1 = 0.75$						
	0.5	0.25	0.0	0.5	0.25	0.5	0.75		0.0	0.25	0.5	0.75		0.0	0.25	0.5	0.75		0.0	0.25	0.5	0.75	
TRUE	0.248	0.248	0.248	0.248	0.894	0.894	0.894		0.894	0.894	0.894	0.894		0.894	0.894	0.894	0.894		0.894	0.894	0.894	0.894	
RP_ρ	0.301	0.316	0.308	0.261	0.883	0.874	0.843	0.881	0.876	0.863	0.799	0.823	0.831	0.819	0.762	0.583	0.603	0.587	0.532				
RP	0.256	0.262	0.244	0.209	0.885	0.873	0.839	0.890	0.879	0.863	0.812	0.879	0.862	0.838	0.770	0.855	0.814	0.766	0.617				
NAT13	0.226	0.219	0.194	0.195	0.860	0.830	0.774	0.865	0.836	0.802	0.748	0.839	0.810	0.777	0.721	0.809	0.776	0.736	0.640				
ELK08	0.221	0.226	0.228	0.210	0.862	0.830	0.771	0.864	0.847	0.819	0.762	0.843	0.835	0.814	0.736	0.674	0.669	0.599	0.473				
LIU16	0.182	0.182	0.000	0.182	0.021	0.000	0.000	0.000	0.147	0.147	0.073	0.000	0.164	0.163	0.163	0.047	0.158	0.145	0.164				

nearest to $\pi_1 = 0$, $\rho_1 = 0$ and mostly captures CNN prediction variation because $|\tilde{P}| \ll |\tilde{N}|$.

Second, RP_ρ often achieves equivalent (MNIST in Table 8.4) or significantly higher (CIFAR in Tables 8.3 and 8.4) F1 score than Rank Pruning when ρ_1 and π_1 are provided, particularly when noise rates are large. This effect is exacerbated for harder problems (lower F1 score for the ground truth classifier) like the “cat” in CIFAR or the “9” digit in MNIST likely because these problems are more complex, resulting in less confident predictions, and therefore more pruning.

Remember that ρ_1^{conf} and ρ_0^{conf} are upper bounds when g is unassuming. Noise rate overestimation accounts for the complexity of harder problems. As a downside, Rank Pruning may remove correctly labeled examples that “confuse” the classifier, instead fitting only the confident examples in each class. We observe this on CIFAR in Table 8.3 where logistic regression severely underfits so that RP_ρ has significantly higher F1 score than the ground truth classifier. Although Rank Pruning with noisy labels seemingly outperforms the ground truth model, if we lower the classification threshold to 0.3 instead of 0.5, the performance difference goes away by accounting for the lower probability predictions.

8.5 Discussion

To our knowledge, Rank Pruning is the first time-efficient algorithm, w.r.t. classifier fitting time, for $\tilde{P}\tilde{N}$ learning that achieves similar or better F1, error, and AUC-PR than current state-of-the-art methods across practical scenarios for synthetic, MNIST, and CIFAR datasets,

Table 8.4: F1 score comparison on MNIST and CIFAR-10 using a CNN. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods.

MNIST/CIFAR		$\pi_1 = 0.0$				$\pi_1 = 0.25$				$\pi_1 = 0.5$						
CLASS	TRUE	$\rho_1 = 0.5$		$\rho_1 = 0.25$		$\rho_1 = 0.0$		$\rho_1 = 0.5$		$\rho_1 = 0.0$	$\rho_1 = 0.5$	$\rho_1 = 0.5$				
		RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16
0	0.993	0.991	0.988	0.977	0.976	0.179	0.991	0.992	0.982	0.981	0.179	0.991	0.992	0.984	0.987	0.985
1	0.993	0.990	0.991	0.989	0.985	0.204	0.992	0.992	0.984	0.987	0.204	0.990	0.991	0.992	0.993	0.990
2	0.987	0.973	0.976	0.972	0.969	0.187	0.984	0.983	0.978	0.975	0.187	0.985	0.986	0.985	0.986	0.975
3	0.990	0.984	0.984	0.972	0.981	0.183	0.986	0.986	0.978	0.978	0.183	0.990	0.987	0.989	0.989	0.984
4	0.994	0.981	0.979	0.981	0.977	0.179	0.985	0.987	0.971	0.964	0.179	0.987	0.990	0.990	0.989	0.985
5	0.989	0.982	0.980	0.978	0.979	0.164	0.985	0.982	0.964	0.965	0.164	0.988	0.987	0.987	0.984	0.987
6	0.989	0.986	0.985	0.972	0.982	0.175	0.985	0.987	0.978	0.981	0.175	0.985	0.985	0.988	0.987	0.985
7	0.987	0.981	0.980	0.967	0.948	0.186	0.976	0.975	0.971	0.971	0.186	0.976	0.980	0.985	0.982	0.983
8	0.989	0.975	0.978	0.943	0.967	0.178	0.982	0.981	0.967	0.951	0.178	0.982	0.984	0.982	0.979	0.983
9	0.982	0.966	0.974	0.972	0.935	0.183	0.976	0.974	0.967	0.967	0.183	0.976	0.975	0.974	0.978	0.970
AVG _{MN}	0.989	0.981	0.981	0.972	0.970	0.182	0.984	0.984	0.974	0.972	0.182	0.985	0.986	0.986	0.985	0.984
PLANE	0.755	0.689	0.634	0.619	0.585	0.182	0.695	0.702	0.671	0.640	0.182	0.757	0.746	0.716	0.735	0.000
AUTO	0.891	0.791	0.785	0.761	0.768	0.000	0.832	0.824	0.771	0.783	0.182	0.862	0.866	0.869	0.865	0.000
BIRD	0.669	0.504	0.483	0.445	0.389	0.182	0.543	0.515	0.469	0.426	0.182	0.577	0.619	0.543	0.551	0.000
CAT	0.487	0.350	0.279	0.310	0.313	0.000	0.426	0.317	0.350	0.345	0.182	0.489	0.433	0.426	0.347	0.000
DEER	0.726	0.593	0.540	0.455	0.522	0.182	0.585	0.554	0.480	0.569	0.182	0.614	0.630	0.643	0.633	0.000
DOG	0.569	0.544	0.577	0.429	0.456	0.000	0.579	0.559	0.569	0.576	0.182	0.647	0.637	0.667	0.630	0.000
FROG	0.815	0.746	0.727	0.733	0.718	0.000	0.729	0.750	0.630	0.584	0.182	0.767	0.782	0.777	0.770	0.000
HORSE	0.805	0.690	0.670	0.624	0.672	0.182	0.710	0.669	0.683	0.627	0.182	0.761	0.776	0.769	0.753	0.000
SHIP	0.851	0.791	0.783	0.719	0.758	0.182	0.810	0.801	0.758	0.723	0.182	0.816	0.822	0.830	0.831	0.000
TRUCK	0.861	0.744	0.722	0.655	0.665	0.182	0.814	0.826	0.798	0.774	0.182	0.810	0.830	0.826	0.824	0.000
AVG _{CF}	0.743	0.644	0.620	0.575	0.585	0.109	0.672	0.652	0.618	0.605	0.182	0.710	0.714	0.707	0.694	0.000

with logistic regression and CNN classifiers, across all noise rates, ρ_1, ρ_0 , for varying added noise, dimension, separability, and number of training examples. By *learning with confident examples*, we discover provably consistent estimators for noise rates, ρ_1, ρ_0 , derive theoretical solutions when g is unassuming, and accurately uncover the classifications of f fit to hidden labels, perfectly when g range separates P and N .

We recognize that disambiguating whether we are in the unassuming or range separability condition may be desirable. Although knowing $g^*(x)$ and thus $\Delta g(x)$ is impossible, if we assume randomly uniform noise, and toggling the $LB_{y=1}$ threshold does not change ρ_1^{conf} , then g range separates P and N . When g is unassuming, Rank Pruning is still robust to imperfect $g(x)$ within a range separable subset of P and N by training with confident examples even when noise rate estimates are inexact.

An important contribution of Rank Pruning is generality, both in classifier and implementation. The use of logistic regression and a generic CNN in our experiments emphasizes that our findings are not dependent on model complexity. We evaluate thousands of scenarios to avoid findings that are an artifact of problem setup. A key point of Rank Pruning is that we only report the simplest, non-parametric version. For example, we use 3-fold cross-validation to compute $g(x)$ even though we achieved improved performance with larger folds. We tried many variants of pruning and achieved significant higher F1 for MNIST and

CIFAR, but to maintain generality, we present only the basic model.

At its core, Rank Pruning is a simple, robust, and general solution for noisy binary classification by *learning with confident examples*, but it also challenges how we think about training data. For example, SVM showed how a decision boundary can be recovered from only support vectors. Yet, when training data contains significant mislabeling, confident examples, many of which are far from the boundary, are informative for uncovering the true relationship $P(y = 1|x)$. Although modern affordances of “big data” emphasize the value of *more* examples for training, through Rank Pruning we instead encourage a rethinking of learning with *confident* examples.

Chapter 9

Conclusion and Prospects

9.1 Conclusions

In this thesis, I have addressed several key aspects of intelligence: few-shot learning, representation learning, causal learning, lifelong learning, improving robustness, and improving intelligibility. Different works usually involve more than one aspect, and use one aspect to improve some others. Also, physics and information play pivotal roles (Fig. 1-1). In AI Physicist (Chapter 2), we apply four physicist strategies to build an agent that has the capability of predicting the future, few-shot learning, lifelong learning, and interpretability. The agent we develop can in turn also learn physics theories in prototypical environments. In MeLA (Chapter 7), we show that learning good representation can help predicting the future in a few-shot way. To understand the two-term tradeoff in representation learning, inspired by phase transitions in physics, we study the phase transitions in the Information Bottleneck (IB) (Chapter 3 and Chapter 4). We derive formulas for giving the condition for IB phase transitions. Based on the formulas, we show the close interplay between the information objective, the dataset and the learned representation, by revealing that each phase transition corresponds to learning of a new component of nonlinear maximum correlation between the input and the target. We also show how the phase transitions depend on the model capacity. In addition, for binary classification, we study the mutual information with target vs. entropy of the representation tradeoff (Chapter 5), by proving that we can reach the

Pareto frontier by binning a uniformized sorted probability of the target given the input, and illustrate how it can be interpreted as an information-theoretically optimal image clustering algorithm. To enable machines to understand causality from observations, we introduce an algorithm that combines predicting the future with minimizing information from the input, for exploratory causal discovery of observational time series, and demonstrate its effectiveness in synthetic, video game, breath rate vs. heart rate and *C.elegans* datasets. To improve robustness of classifiers to noisy labels, we introduce Rank Pruning for learning under noisy labels. Under mild assumptions, we prove that it can achieve the same accuracy as if the labels are not corrupted. We also demonstrate that it improves the state-of-the-art in noisy label classification. I believe this progress will bring us one step closer to building intelligent machines that can make sense of the world and become better at learning.

9.2 Prospects

Looking ahead, I believe there are vast opportunities ahead. The graph in Fig. 1-1 is far from a complete graph. And I believe an ideal graph will look like Fig. 9-1, where all the aspects work integrally together. As a first step, any edge, and with either of the two directions, implies an opportunity. Take some aspects which have not been addressed by my thesis for example: learning good representations can improve robustness; learning good representation can help lifelong learning; causal learning can improve robustness, improving interpretability can help few-shot learning; to name just a few. Besides, there exists many other aspects either mentioned or not mentioned in Section 1.1, which we can also draw many edges with. Suppose that there are N aspects. Then we have $N(N - 1)$ directed edges above. Moreover, we can try to combine three or more aspects together, just like my AI Physicist work, which implies on the order of N^3 or larger number of combinations.

Ultimately, we want an agent that can combine all the aspects integrally together in an elegant way. Although daunting, I think it is possible, and it may not be so far away. As we have seen, physics and information have provided valuable tools underlying my thesis and many other works, and I believe they will certainly play a central role, in combining

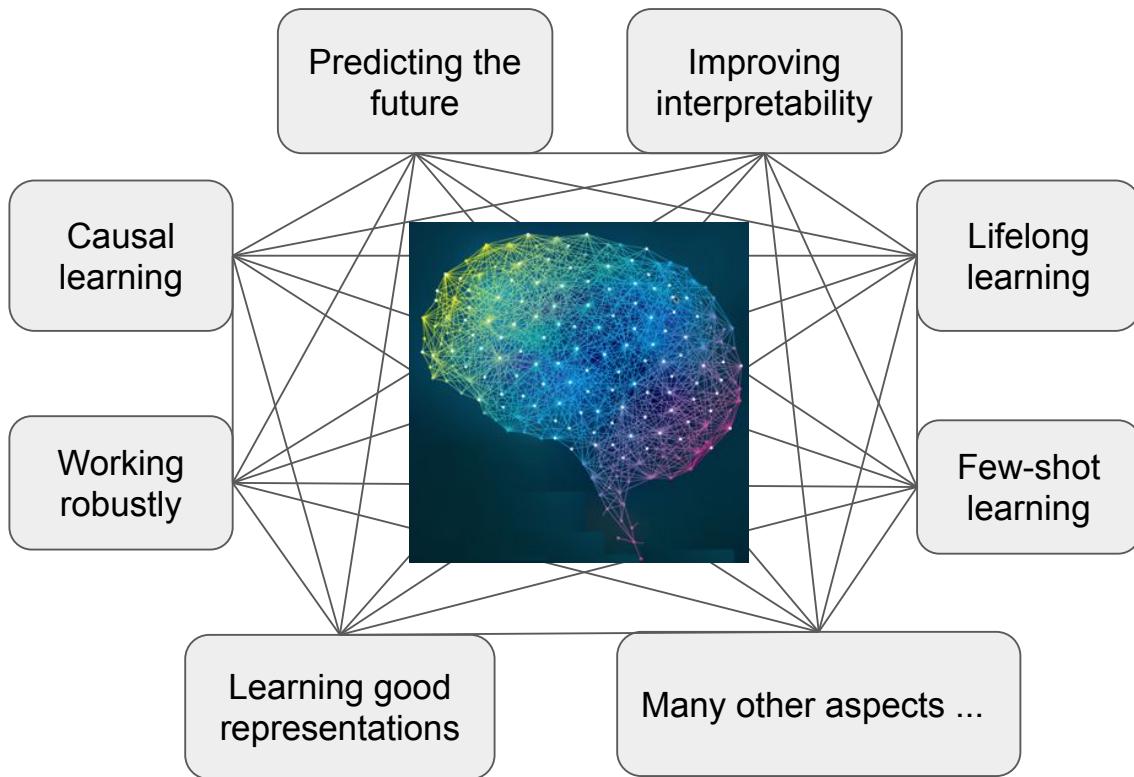


Figure 9-1: Prospect of different aspects combined integrally together.

theses aspects integrally together, to build an agent that can make sense of the world, and help solve many problems in the society.

Appendix A

Appendix

A.1 Appendix for Chapter 2

A.1.1 AI Physicist Algorithm

The detailed AI Physicist algorithm is presented¹ in Algorithm 6, with links to each of the individual sub-algorithms. Like most numerical methods, the algorithm contains a number of hyperparameters that can be tuned to optimize performance; Table A.3 lists them and their settings for our numerical experiments.

Algorithm 6 AI Physicist: Overall algorithm

Given observations $D = \{(\mathbf{x}_t, \mathbf{y}_t)\}$ from new environment:

- 1: $\mathcal{T}_{M_0} \leftarrow \mathbf{Hub}.\text{propose-theories}(D, M_0)$ (Alg. 10)
- 2: $\mathcal{T} \leftarrow \text{differentiable-divide-and-conquer}(D, \mathcal{T}_{M_0})$ (Alg. 7)
- 3: $\mathbf{Hub}.\text{add-theories}(\mathcal{T}, D)$ (Alg. 11)

Organizing theory hub:

- $$\begin{aligned}\mathcal{T} &\leftarrow \mathbf{Hub}.\text{Occam's-Razor-with-MDL}(\mathcal{T}, D) \text{ (Alg. 8)} \\ \mathcal{T} &\leftarrow \mathbf{Hub}.\text{unify}(\mathcal{T}) \text{ (Alg. 9)}\end{aligned}$$
-

A.1.2 The Differentiable Divide-and-Conquer (DDAC) Algorithm

Here we elaborate on our differentiable divide-and-conquer (DDAC) algorithm with generalized-mean loss \mathcal{L}_γ (Eq. (2.2)). This loss with $\gamma < 0$ works with a broad range of distance functions ℓ satisfying Theorem 1. Since the goal of our AI Physicist is to minimize the overall description length (DL) from equation (2.4), we choose ℓ to be the DL loss function of equation (2.7) together with $\gamma = -1$ (harmonic loss), which works quite well in practice.

Algorithm 7 describes our differentiable divide-and-conquer implementation, which consists of two stages. In the first stage (steps 2-6), it applies the subroutine $\text{IterativeTrain}(\mathcal{T}, D, \ell_{\text{DL}, \epsilon}, \mathcal{L}_{-1})$ with harmonic loss \mathcal{L}_{-1} to train the theories \mathcal{T} a few times with the precision floor ϵ gradually lowered according to the following annealing schedule. We set the initial precision floor ϵ to be quite large so that ℓ initially approximates an MSE loss function. After each successive iteration, we reset ϵ to the median prediction error.

¹The full code is open-sourced at github.com/tailintalent/AI_physicist.

The DL loss function from equation (2.7) is theoretically desirable but tricky to train, both because it is non-convex and because it is quite flat and uninformative far from its minimum. Our annealing schedule helps overcome both problems: initially when ϵ is large, it approximates MSE-loss which is convex and guides the training to a good approximate minimum, which further training accurately pinpoints as ϵ is reduced.

The subroutine `IterativeTrain` forms the core of the algorithm. In the first stage (steps 2-6), it uses the harmonic mean of the DL-loss of multiple prediction functions $\mathbf{f}_\theta = (\mathbf{f}_1, \dots, \mathbf{f}_M)$ (*i.e.*, equation (2.2) with $\gamma = -1$ and $\ell = \text{DL}$) to simultaneously train these functions, encouraging them to each specialize in the domains where they predict best (as proven by Theorem 1), and simultaneously trains the domain classifier $\mathbf{c}_\phi = (c_1, \dots, c_M)$ using each example's best-performing prediction function as target, with categorical cross-entropy loss. After several rounds of `IterativeTrain` with successively lower precision floors, each prediction function typically becomes good at predicting part of the dataset, and the domain classifier becomes good at predicting for each example which prediction function will predict best.

`AddTheories`($\mathcal{T}, D, \ell, \mathcal{L}$) inspects each theory \mathcal{T}_i describing at least a large fraction η_{insp} (we use 30%) of the examples to see if a non-negligible proportion η_{split} of examples (we use a threshold of 5%) of the examples inside its domain have MSE larger than a certain limit ϵ_{add} (we use 2×10^{-6}) and thus warrant splitting off into a separate domain.

If so, it uses those examples to initialize a new theory \mathcal{T}_{M+1} , and performs tentative training together with other theories using `IterativeTrain` without steps s8 and s9 (it is also possible to allow steps s8 and s9 in this recursive calling of `IterativeTrain`, which will enable a recursive adding of theories for not-well-explained data, and may enable a more powerful DDAC algorithm). If the resulting loss \mathcal{L} is smaller than before adding the new theory, \mathcal{T}_{M+1} is accepted and retained, otherwise it is rejected and training reverts to the checkpoint before adding the theory. `DeleteTheories`(\mathcal{T}, D, ℓ) deletes theories whose domain or best-predicted examples cover a negligible fraction of the examples (we use a delete threshold $\eta_{\text{del}} = 0.5\%$).

In the second stage (steps 7-10), the `IterativeTrain` is applied again, but the loss for each example $(\mathbf{x}_t, \mathbf{y}_t)$ is using only the theory that the domain classifier $\mathbf{c}_\phi = (c_1, c_2, \dots, c_M)$

predicts (having the largest logit). In this way, we iteratively fine-tune the prediction functions $\{\mathbf{f}_i\}$ w.r.t. each of its domain, and fine-tune the domain to the best performing theory at each point. The reason that we assign examples to domains using our domain classifier rather than prediction accuracy is that the trained domains are likely to be simpler and more contiguous, thus generalizing better to unseen examples than, *e.g.*, the nearest neighbor algorithm.

We now specify the default hyperparameters used for Algorithm 1 in our experiments (unless otherwise specified). We set the initial total number of theories $M = 4$, from which $M_0 = 2$ theories are proposed from the theory hub. The initial precision floor $\epsilon_0 = 10$ and the number of gradient iterations $K = 10000$. We use the Adam [KB14] optimizer with default parameters for the optimization of both the prediction function and the domain classifier. We randomly split each dataset D into D_{train} and D_{test} with 4:1 ratio. The D_{test} is used only for evaluation of performance. The batch size is set to $\min(2000, |D_{train}|)$. We set the initial learning rate $\beta_f = 5 \times 10^{-3}$ for the prediction functions \mathbf{f}_θ and $\beta_c = 10^{-3}$ for the domain classifier \mathbf{c}_ϕ . We also use a learning rate scheduler that monitors the validation loss every 10 epochs, and divides the learning rate by 10 if the validation loss has failed to decrease after 40 monitoring points and stops training early if there is no decrease after 200 epochs — or if the entire MSE loss for all the theories in their respective domains drops below 10^{-12} .

To the main harmonic loss \mathcal{L}_γ , we add two regularization terms. One is L_1 loss whose strength increases quadratically from 0 to ϵ_{L1} during the first 5000 epochs and remains constant thereafter. The second regularization term is a very small MSE loss of strength ϵ_{MSE} , to encourage the prediction functions to remain not too far away from the target outside their domain.

A.1.3 Occam’s Razor with MDL Algorithm

Pushing on after the DDAC algorithm with harmonic loss that minimizes the $\sum_t \text{DL}(\mathbf{u}_t)$ term in Eq. (2.4), the AI Physicist then strives to minimize the $\text{DL}(\mathcal{T})$ term, which can be decomposed as $\text{DL}(\mathcal{T}) = \text{DL}(\mathbf{f}_\theta) + \text{DL}(\mathbf{c}_\phi)$, where $\mathbf{f}_\theta = (\mathbf{f}_1, \dots, \mathbf{f}_M)$ and $\mathbf{c}_\phi = (c_1, \dots, c_M)$. We focus on minimizing $\text{DL}(\mathbf{f}_\theta)$, since in different environments the prediction functions \mathbf{f}_i

can often be reused, while the domains may differ. As mentioned, we define $\text{DL}(\mathbf{f}_\theta)$ simply as the sum of the description lengths of the numbers parameterizing \mathbf{f}_θ :

$$\text{DL}(\mathbf{f}_\theta) = \sum_j \text{DL}(\theta_j). \quad (\text{A.1})$$

This means that $\text{DL}(\mathbf{f}_\theta)$ can be significantly reduced if an irrational parameter is replaced by a simpler rational number.

If a physics experiment or neural net training produces a parameter $p = 1.999942$, it would be natural to interpret this as a hint, and to check if $p = 2$ gives an equally acceptable fit to the data. We formalize this by replacing any real-valued parameter p_i in our theory \mathbf{T} by its nearest integer if this reduces the total description length in equation (2.4), as detailed below. We start this search for integer candidates with the parameter that is closest to an integer, refitting for the other parameters after each successful “integer snap”.

What if we instead observe a parameter $p = 1.5000017$? Whereas generic real numbers have a closest integer, they lack a closest rational number. Moreover, as illustrated in Figure 2-2, we care not only about closeness (to avoid increasing the second term in equation (2.4)), but also about simplicity (to reduce the first term). To rapidly find the best “rational snap” candidates (dots in Figure 2-2 that lie both near p and far down), we perform a continued fraction expansion of p and use each series truncation as a rational candidate. We repeat this for all parameters in the theory \mathbf{T} , again accepting only those snaps that reduce the total description length. We again wish to try the most promising snap candidates first; to rapidly identify promising candidates without having to recompute the second term in equation (2.4), we evaluate all truncations of all parameters as in Figure 2-3, comparing the description length of the rational approximation $q = m/n$ with the description length of the approximation error $|p - q|$. The most promising candidate minimizes their sum, *i.e.*, lies furthest down to the left of the diagonal in the figure. The figure illustrates how, given the parameter vector $\mathbf{p} = \{\pi, \sqrt{2}, 3.43180632382353\}$, the first snap to be attempted will replace the third parameter by $53/17$.

We propose Algorithm 8 to implement the above minimization of $\text{DL}(\mathbf{f}_\theta)$ without increasing

$\text{DL}(\mathcal{T}, D)$ (Eq. 2.4). For each theory $\mathbf{T}_i = (\mathbf{f}_i, c_i)$, we first extract the examples $D^{(i)}$ inside its domain, then perform a series of tentative transformations (simplifications) of the prediction function \mathbf{f}_i using the `MinimizeDL` subroutine. This subroutine takes \mathbf{f}_i , the transformation, and $D^{(i)}$ as inputs and repeatedly applies the transformation to \mathbf{f}_i . After each such transformation, it fine-tunes the fit of \mathbf{f}_i to $D^{(i)}$ using gradient descent. For determining whether to accept the transformation, Algorithm 8 presents the simplest 0-step patience implementation: if the description length $\text{dl} = \text{DL}(\mathbf{f}_i) + \sum_{(\mathbf{x}_t, \mathbf{y}_t) \in D^{(i)}} \ell_{\text{DL}, \epsilon}(\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t)$ for theory i decreases, then apply the transformation again if possible, otherwise exit the loop. In general, to allow for temporary increase of DL during the transformations, a non-zero patience can be adopted: at each step, save the best performing model as the pivot model, and if DL does not decrease during n consecutive transformations inside `MinimizeDL`, exit the loop. In our implementation, we use a 4-step patience.

We now detail the five transformations used in Algorithm 8. The `collapseLayer` transformation finds all successive layers of a neural net where the lower layer has linear activation, and combines them into one. The `toSymbolic` transformation transforms \mathbf{f}_i from the form of a neural net into a symbolic expression (in our implementation, from a PyTorch net to a SymPy symbolic lambda expression). These two transformations are one-time transformations (for example, once \mathbf{f}_i has been transformed to a symbolic expression, `toSymbolic` cannot be applied to it again.) The `localSnap` transformation successively sets the incoming weights in the first layer to 0, thus favoring inputs that are closer to the current time step. The `integerSnap` transformation finds the (non-snapped) parameters in \mathbf{f}_i that is closest to an integer, and snaps it to that integer. The `rationalSnap` transformation finds the (non-snapped) parameter in \mathbf{f}_i that has the lowest bit sum when replaced by a rational number, as described in section 2.2.4, and snaps it to that rational number. The latter three transformations can be applied multiple times to \mathbf{f}_i , until there are no more parameters to snap in \mathbf{f}_i , or the transformation followed by fine-tuning fails to reduce the description length.

In the bigger picture, Algorithm 8 is an implementation of minimizing the $\text{DL}(\mathbf{f}_\theta)$ without increasing the total $\text{DL}(\mathcal{T}, D)$, if the description length of \mathbf{f}_θ is given by Eq. (A.1). There can be other ways to encode \mathcal{T} with a different formula for $\text{DL}(\mathcal{T})$, in which case the

transformations for decreasing $\text{DL}(\mathcal{T})$ may be different. But the structure of the Algorithm 8 remains the same, with the goal of minimizing $\text{DL}(\mathbf{f}_\theta)$ without increasing $\text{DL}(\mathcal{T}, D)$ w.r.t. whatever DL formula it is based on.

In the still bigger picture, Algorithm 8 is a computationally efficient approximate implementation of the MDL formalism, involving the following two approximations:

1. The description lengths $\text{DL}(x)$ for various types of numbers are approximate, for convenience. For example, the length of the shortest self-terminating bit-string encoding an arbitrary natural number n grows slightly faster than our approximation $\log_2 n$, because self-termination requires storing not only the binary digits of the integer, but also the length of said bit string, recursively, requiring $\log_2 n + \log_2 \log_2 n + \log_2 \log_2 \log_2 n + \dots$, where only the positive terms are included [Ris83]. Slight additional overhead is required to upgrade the encodings to actual programs in some suitable language, including encoding of whether bits encode integers, rational numbers, floating-point numbers, *etc.*.
2. If the above-mentioned $\text{DL}(x)$ -formulas were made exact, they would be mere *upper bounds* on the true minimum description length. For example, our algorithm gives a gigabyte description length for $\sqrt{2}$ with precision $\epsilon = 256^{-10^9}$, even though it can be computed by a rather short program, and there is no simple algorithm for determining which numbers can be accurately approximated by algebraic numbers. Computing the true minimum description length is a famous numerically intractable problem.

A.1.4 Unification Algorithm

The unification process takes as input the symbolic prediction functions $\{(\mathbf{f}_i, \cdot)\}$, and outputs master theories $\mathcal{T} = \{(\mathbf{f}_p, \cdot)\}$ such that by varying each p in \mathbf{f}_p , we can generate a continuum of prediction functions \mathbf{f}_i within a certain class of prediction functions. The symbolic expression consists of 3 building blocks: operators (e.g. $+, -, \times, /$), input variables (e.g. x_1, x_2), and coefficients that can be either a rational number or irrational number.

The unification algorithm first calculates the DL $\text{dl}^{(i)}$ of each prediction function, then clusters them into K clusters using e.g. K-means clustering. Within each cluster S_k , it first canonicalizes each $\mathbf{f}_{i_k} \in S_k$ into a 2-tuple $(\mathbf{g}_{i_k}, \mathbf{h}_{i_k})$, where \mathbf{g}_{i_k} is a tree-form expression of \mathbf{f}_{i_k} where each internal node is an operator, and each leaf is an input variable or a coefficient. When multiple orderings are equivalent (e.g. $x_1 + x_2 + x_3$ vs. $x_1 + x_3 + x_2$), it always uses a predefined partial ordering. \mathbf{h}_{i_k} is the structure of \mathbf{g}_{i_k} where all coefficients are replaced by an s symbol. Then the algorithm obtains a set of \mathbf{g}_{i_k} that has the same structure \mathbf{h}_{i_k} with the largest cardinality (steps 7-8). This will eliminate some expressions within the cluster that might interfere with the following unification process. Step 9 is the core part, where it traverses each $\mathbf{g}_{i_k} \in G_k$ with synchronized steps using e.g. depth-first search or breath-first search. This is possible since each $\mathbf{g}_{i_k} \in G_k$ has the same tree structure h_k^* . During traversing, whenever encountering a coefficient and not all coefficients across G_k at this position are the same, replace the coefficients by some symbol \mathbf{p}_{j_k} that has not been used before. Essentially, we are turning all coefficients that varies across G_k into a parameter, and the coefficients that do not vary stay as they are. In this way, we obtain a master prediction function $\mathbf{f}_{\mathbf{p}_k}$. Finally, at step 13, the algorithm merges the master prediction functions in $\mathcal{T} = \{(\mathbf{f}_{\mathbf{p}_k}, \cdot)\}$ that have the exact same form, and return \mathcal{T} . The domain classifier is neglected during the unification process, since at different environments, each prediction function can have vastly different spacial domains. It is the prediction function (which characterizes the equation of motion) that is important for generalization.

A.1.5 Adding and Proposing Theories

Here we detail the algorithms adding theories to the hub and proposing them for use in new environments. Alg. 10 provides a simplest version of the theory proposing algorithm. Given a new dataset D , the theory hub inspects all theories i , and for each one, counts the number n_i of data points where it outperforms all other theories. The top M_0 theories with largest n_i are then proposed.

For theory adding after training with DDAC (Alg. 7), each theory i calculates its description length $\text{dl}^{(i)}$ inside its domain. If its $\text{dl}^{(i)}$ is smaller than a threshold η , then the theory (\mathbf{f}_i, c_i)

with its corresponding examples $D^{(i)}$ are added to the theory hub. The reason why the data $D^{(i)}$ are also added to the hub is that $D^{(i)}$ gives a reference for how the theory (\mathbf{f}_i, c_i) was trained, and is also needed in the Occam's razor algorithm.

A.1.6 Time complexity

In this appendix, we list crude estimates of the time complexity of our AI physicist algorithm, *i.e.*, of how its runtime scales with key parameters.

DDAC, the differentiable divide-and-conquer algorithm, algorithm (Alg. 7), is run once for each of the n_{myst} different mystery worlds, with a total runtime scaling roughly as

$$\mathcal{O}(n_{\text{myst}} n_{\text{par}} n_{\text{data}} n_{\text{dom}}^2),$$

where n_{par} is the average number of neural-network parameter in a theory, n_{data} is the average number of data points (time steps) per mystery and n_{dom} is the number of discovered domains (in our case ≤ 4). The power of two for n_{dom} appears because the time to evaluate the loss function scales as n_{dom} , and we need to perform of order n_{dom} training cycles to add the right number of theories. The n_{par} scaling is due to that the forward and backward propagation of neural net involves successive matrix multiplied by a vector, which scales as $O(n^2)$ where $n \simeq \sqrt{n_{\text{par}}/N_{\text{lay}}^f}$ is the matrix dimension for each layer and N_{lay}^f is the number of layers. Accumulating all layers we have $N_{\text{lay}}^f n^2 = n_{\text{par}}$. We make no attempt to model how the number of training epochs needed to attain the desired accuracy depends on parameters.

Our **Lifelong learning** algorithm is also run once per mystery, with a time cost dominated by that for proposing new theories (Alg. 10), which scales as

$$\mathcal{O}(n_{\text{myst}} n_{\text{data}} n_{\text{theo}}).$$

Here n_{theo} is the number of theories in theory hub.

In contrast, our **Occam's Razor** algorithm (Alg. 8) and **unification** algorithm (Alg. 9) are

run once per learned theory, not once per mystery. For Occam’s Razor, the total runtime is dominated by that for snapping to rational numbers, which scales as

$$\mathcal{O}(n_{\text{par}} n_{\text{data}} n_{\text{theo}}).$$

For the unification, the total runtime scales as $\mathcal{O}(n_{\text{par}} n_{\text{theo}})$, which can be neglected relative to the cost of Occam’s razor.

We note that all these algorithms have merely polynomial time complexity. The DDAC algorithm dominates the time cost; our mystery worlds were typically solved in about 1 hour each on a single CPU. If vast amounts of data are available, it may suffice to analyze a random subset of much smaller size.

A.1.7 Proof of Theorem 1 and Corollary

Here we give the proof for Theorem 1, restated here for convenience.

Theorem 1 Let $\hat{\mathbf{y}}_t^{(i)} \equiv \mathbf{f}_i(\mathbf{x}_t)$ denote the prediction of the target \mathbf{y}_t by the function \mathbf{f}_i , $i = 1, 2, \dots, M$. Suppose that $\gamma < 0$ and $\ell(\hat{\mathbf{y}}_t, \mathbf{y}_t) = \ell(|\hat{\mathbf{y}}_t - \mathbf{y}_t|)$ for a monotonically increasing function $\ell(u)$ that vanishes on $[0, u_0]$ for some $u_0 \geq 0$, with $\ell(u)^\gamma$ differentiable and strictly convex for $u > u_0$.

Then if $0 < \ell(\hat{\mathbf{y}}_t^{(i)}, \mathbf{y}_t) < \ell(\hat{\mathbf{y}}_t^{(j)}, \mathbf{y}_t)$, we have

$$\left| \frac{\partial \mathcal{L}_\gamma}{\partial u_t^{(i)}} \right| > \left| \frac{\partial \mathcal{L}_\gamma}{\partial u_t^{(j)}} \right|, \quad (\text{A.2})$$

where $u_t^{(i)} \equiv |\hat{\mathbf{y}}_t^{(i)} - \mathbf{y}_t|$.

Proof. Since $u_t^{(i)} \equiv |\hat{\mathbf{y}}_t^{(i)} - \mathbf{y}_t|$ and $\ell(\hat{\mathbf{y}}_t, \mathbf{y}_t) = \ell(|\hat{\mathbf{y}}_t - \mathbf{y}_t|)$, the generalized mean loss L_γ as defined in Eq. 2.3 can be rewritten as

$$\mathcal{L}_\gamma = \sum_t \left(\frac{1}{M} \sum_{k=1}^M \ell(u_t^{(k)})^\gamma \right)^{\frac{1}{\gamma}}, \quad (\text{A.3})$$

which implies that

$$\begin{aligned} \left| \frac{\partial \mathcal{L}_\gamma}{\partial u_t^{(i)}} \right| &= \left| \frac{1}{\gamma M} \left(\frac{1}{M} \sum_{k=1}^M \ell(u_t^{(k)})^\gamma \right)^{\frac{1}{\gamma}-1} \frac{d\ell(u_t^{(i)})^\gamma}{du_t^{(i)}} \right| \\ &= \frac{1}{|\gamma|M} \left(\frac{1}{M} \sum_{k=1}^M \ell(u_t^{(k)})^\gamma \right)^{\frac{1}{\gamma}-1} \left| \frac{d\ell(u_t^{(i)})^\gamma}{du_t^{(i)}} \right|. \end{aligned}$$

Since only the last factor depends on i , proving equation (A.2) is equivalent to proving that

$$\left| \frac{d\ell(u_t^{(i)})^\gamma}{du_t^{(i)}} \right| > \left| \frac{d\ell(u_t^{(j)})^\gamma}{du_t^{(j)}} \right|. \quad (\text{A.4})$$

Let us henceforth consider only the case $u > u_0$, since the conditions $\ell(u_t^{(j)}) > \ell(u_t^{(i)}) > 0$ imply $u_t^{(j)} > u_t^{(i)} > u_0$. Since $\gamma < 0$, $\ell(u) > 0$ and $\ell'(u) \geq 0$, we have $\frac{d\ell(u)^\gamma}{du} = \gamma \ell(u)^{\gamma-1} \ell'(u) \leq 0$, so that $\left| \frac{d\ell(u)^\gamma}{du} \right| = -\frac{d\ell(u)^\gamma}{du}$. Because $\ell(u)^\gamma$ is differentiable and strictly convex, its derivative $\frac{d\ell(u)^\gamma}{du}$ is monotonically increasing, implying that $\left| \frac{d\ell(u)^\gamma}{du} \right| = -\frac{d\ell(u)^\gamma}{du}$ is monotonically decreasing. Thus $\left| \frac{d\ell(u_1)^\gamma}{du_1} \right| > \left| \frac{d\ell(u_2)^\gamma}{du_2} \right|$ whenever $u_1 < u_2$. Setting $u_1 = |\hat{\mathbf{y}}_t^{(i)} - \mathbf{y}_t|$ and $u_2 = |\hat{\mathbf{y}}_t^{(j)} - \mathbf{y}_t|$ therefore implies equation (A.4), which completes the proof.

The following corollary 12.1 demonstrates that the theorem applies to several popular loss functions as well as our two description-length loss functions.

Corollary 12.1. *Defining $u \equiv |\hat{\mathbf{y}} - \mathbf{y}|$, the following loss functions which depend only on u satisfy the conditions for Theorem 1:*

1. $\ell(u) = u^r$ for any $r > 0$, which includes MSE loss ($r = 2$) and mean-absolute-error loss ($r = 1$).

2. *Huber loss:* $\ell_\delta(u) = \begin{cases} \frac{1}{2}u^2, & u \in [0, \delta] \\ \delta(u - \frac{\delta}{2}), & \text{otherwise,} \end{cases}$
where $\delta > 0$.

3. *Description length loss*

$$\ell_{\text{DL},\epsilon}(u) = \frac{1}{2} \log_2 \left(1 + \left(\frac{u}{\epsilon} \right)^2 \right).$$

4. Hard description length loss

$$\ell_{DLhard,\epsilon}(u) = \log_2 \max \left(1, \frac{u}{\epsilon} \right).$$

Proof. We have $u_0 = 0$ for (1), (2), (3), and $u_0 = \epsilon$ for (4). All four functions ℓ are monotonically increasing, satisfy $\ell(0) = 0$ and are differentiable for $u > u_0$, so all that remains to be shown is that $\ell(u)^\gamma$ is strictly convex for $u > u_0$, i.e., that $\frac{d^2\ell(u)^\gamma}{du^2} > 0$ when $u > u_0$.

(1) For $\ell(u) = u^r$ and $u > 0$, we have $\frac{d^2\ell(u)^\gamma}{du^2} = \gamma r(\gamma r - 1)u^{\gamma r - 2} > 0$, since $\gamma < 0$ and $r > 0$ implies that $\gamma r < 0$ and $\gamma r - 1 < 0$, so $\ell(u)^\gamma$ is strictly convex for $u > 0$.

(2) The Huber loss $\ell_\delta(u)$ is continuous with a continuous derivative. It satisfies $\frac{d^2\ell(u)^\gamma}{du^2} > 0$ both for $0 < u < \delta$ and for $\delta < u$ according to the above proof of (1), since $\ell_\delta(u)$ is proportional to ℓ^r in these two intervals with $r = 2$ and $r = 1$, respectively. At the transition point $u = \delta$, this second derivative is discontinuous, but takes positive value approaching both from the left and from the right, so $\ell(u)^\gamma$ is strictly convex. More generally, any function $\ell(u)$ built by smoothly connecting functions $\ell_i(u)$ in different intervals will satisfy our theorem if the functions $\ell_i(u)$ do.

(3) Proving strict convexity of $\ell(u)^\gamma$ when ℓ is the description length loss $\ell_{DL,\epsilon}(u) = \frac{1}{2}\log_2 \left[1 + \left(\frac{u}{\epsilon} \right)^2 \right]$ is equivalent to proving it when $\ell(u) = \rho(u) \equiv \ln(1+u^2)$, since convexity is invariant under horizontal and vertical scaling. We thus need to prove that

$$\frac{d^2\rho(u)^\gamma}{du^2} = -\frac{2\gamma[\ln(1+u^2)]^{\gamma-2}}{(1+u^2)^2[2u^2(1-\gamma)+(u^2-1)\ln(1+u^2)]}$$

is positive when $u > 0$. The factor $\frac{-2\gamma[\ln(1+u^2)]^{\gamma-2}}{(1+u^2)^2}$ is always positive. The other factor

$$2u^2(1-\gamma)+(u^2-1)\log(1+u^2) > 2u^2+(u^2-1)\log(1+u^2),$$

since $\gamma < 0$. Now we only have to prove that the function

$$\chi(u) \equiv 2u^2 + (u^2 - 1)\log(1 + u^2) > 0$$

when $u > 0$. We have $\chi(0) = 0$ and

$$\chi'(u) = 2u \left[\frac{1+3u^2}{1+u^2} + \log(1+u^2) \right] > 0$$

when $u > 0$. Therefore $\chi(u) = \chi(0) + \int_0^u \chi'(u')du' > 0$ when $u > 0$, which completes the proof that $\ell_{\text{DL},\epsilon}(u)^\gamma$ is strictly convex for $u > 0$.

(4) For the hard description length loss $\ell_{\text{DLhard},\epsilon}(u) = \log_2 \max(1, \frac{u_t}{\epsilon})$, we have $u_0 = \epsilon$.

When $u > \epsilon$, we have $\ell'_{\text{DLhard},\epsilon}(u) = \frac{1}{u \ln 2} > 0$ and

$$\frac{d^2 \ell'_{\text{DLhard},\epsilon}(u)}{du^2} = \frac{\gamma}{\ln 2} \left(-1 + \gamma - \ln \frac{u}{\epsilon} \right) \frac{\left(\ln \frac{u}{\epsilon} \right)^{\gamma-2}}{u^2}.$$

For $u > \epsilon$, the factor $\frac{\left(\ln \frac{u}{\epsilon} \right)^{\gamma-2}}{u^2}$ is always positive, as is the factor $\gamma(-1 + \gamma - \ln \frac{u}{\epsilon})$, since $\gamma < 0$. $\ell'_{\text{DLhard},\epsilon}(u)$ is therefore strictly convex for $u > \epsilon$.

A.1.8 Eliminating Transition Domains

In this appendix, we show how the only hard problem our AI Physicist need solve is to determine the laws of motion far from domain boundaries, because once this is done, the exact boundaries and transition regions can be determined automatically.

Our AI Physicist tries to predict the next position vector $\mathbf{y}_t \in R^d$ from the concatenation $\mathbf{x}_t = (\mathbf{y}_{t-T}, \dots, \mathbf{y}_{t-1})$ of the last T positions vectors. Consider the example shown in Figure A-1, where motion is predicted from the last $T = 3$ positions in a space with $d = 2$ dimensions containing $n = 2$ domains with different physics (an electromagnetic field in the upper left quadrant and free motion elsewhere), as well as perfectly reflective boundaries. Although there are only two physics domains in the 2-dimensional space, there are many more types of domains in the $Td = 6$ -dimensional space of \mathbf{x}_t from which the AI Physicist makes its predictions of \mathbf{y}_t . When a trajectory crosses the boundary between the two spatial regions, there can be instances where \mathbf{x}_t contains 3, 2, 1 or 0 points in the first domain and correspondingly 0, 1, 2 or 3 points in the second domain. Similarly, when the ball bounces, there can be instances where \mathbf{x}_t contains 3, 2, 1 or 0 points before the bounce

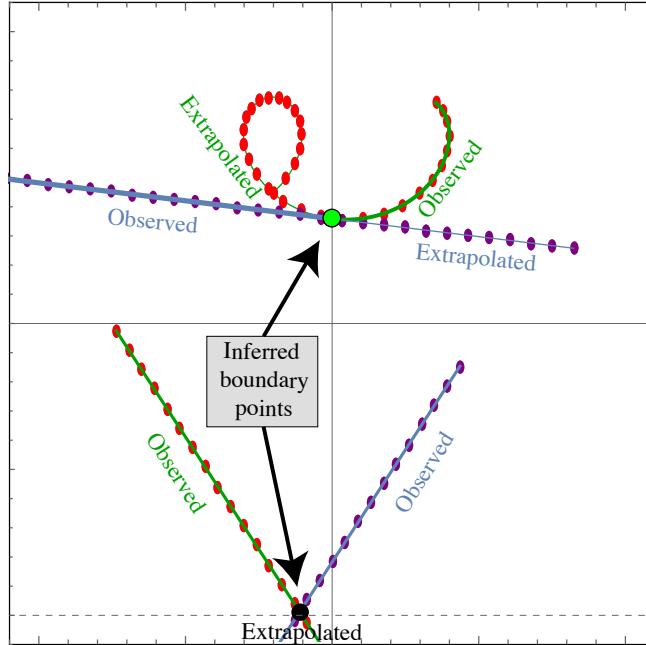


Figure A-1: Points where forward and backward extrapolations agree (large black dots) are boundary points. The tangent vectors agree for region boundaries (upper example), but not for bounce boundaries (lower example).

and correspondingly 0, 1, 2 or 3 points after. Each of these situations involves a different function $\mathbf{x}_t \mapsto \mathbf{y}_t$ and a corresponding 6-dimensional domain of validity for the AI Physicist to learn.

Our numerical experiments showed that the AI Physicist typically solves the big domains (where all vectors in \mathbf{x}_t lie in the same spatial region), but occasionally fails to find an accurate solution in some of the many small transition domains involving boundary crossings or bounces, where data is insufficient. Fortunately, simple post-processing can automatically eliminate these annoying transition domains with an algorithm that we will now describe.

The first step of the algorithm is illustrated in Figure A-1. For each big domain where our AI Physicist has discovered the future-predicting function $\mathbf{x}_t \mapsto \mathbf{y}_t$, we determine the corresponding function that predicts the *past* ($\mathbf{x}_t \mapsto \mathbf{y}_{t-T-1}$) by fitting to forward trajectories generated with random initial conditions. Now whenever a trajectory passes from a big domain through a transition region into another big domain, two different extrapolations can be performed: forward in time from the first big domain or backward in time from the second big domain. Using cubic spline interpolation, we fit continuous functions $\mathbf{y}_f(t)$ and $\mathbf{y}_b(t)$

(smooth curves in Figure A-1) to these forward-extrapolated and backward-extrapolated trajectories, and numerically find the time

$$t_* \equiv \arg \min |\mathbf{y}_f(t) - \mathbf{y}_b(t)| \quad (\text{A.5})$$

when the distance between the two predicted ball positions is minimized. If this minimum is numerically consistent with zero, so that $\mathbf{y}_f(t_*) \approx \mathbf{y}_b(t_*)$, then we record this as being a boundary point. If both extrapolations have the same derivative there, *i.e.*, if $\mathbf{y}'_f(t_*) \approx \mathbf{y}'_b(t_*)$, then it is an interior boundary point between two different regions (Figure A-1, top), otherwise it is an external boundary point where the ball bounces (Figure A-1, bottom).

Figure A-2 show these two types of automatically computed boundary points in green and black, respectively. These can now be used to retrain the domain classifiers to extend the big domains to their full extent, eliminating the transition regions.

Occasionally the boundary point determinations fail because of multiple transitions within T time steps, Figure A-2 illustrates that these failures (red dots) forces us to discard merely a tiny fraction of all cases, thus having a negligible affect on the ability to fit for the domain boundaries.

A.1.9 Numerical Experiment Details

In this appendix, we provide supplementary details on our benchmark problems.

Mystery Worlds

World generation Our mystery worlds consist of a ball elastically bouncing against the square boundary of the two-dimensional spatial region where $|x| \leq 2$ and $|y| \leq 2$ (see Figure 2-4). In each of the four quadrants, one of the following laws of physics are selected, together with their parameters sampled from distributions as follows:

1. Free motion.

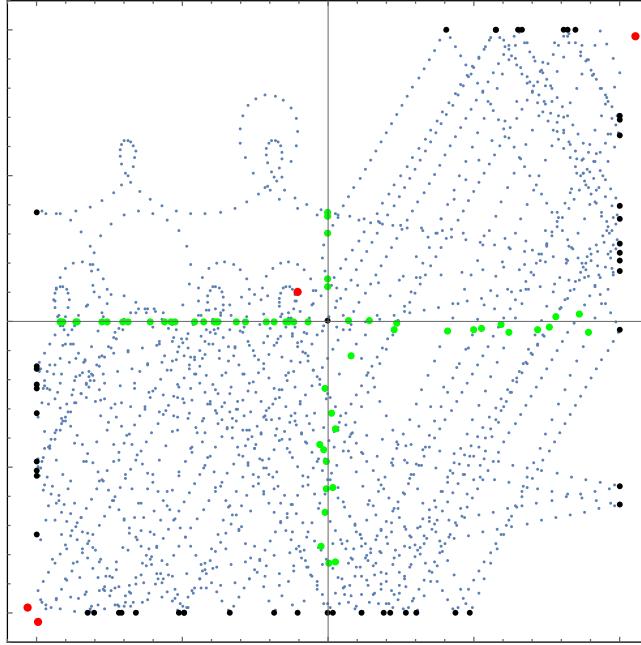


Figure A-2: Example of automatically determined boundary points, for region boundary points (green), bounce boundary points (black) and failed cases (red).

2. A uniform gravitational field $\mathbf{g} = (g_x, g_y, 0)$ with g_x, g_y drawn from a uniform distribution: $g_x, g_y \sim U[-5, 5]$.
3. Harmonic motion with frequency ω around a line a distance a from the origin, making an angle ϕ with the x -axis; $\omega \sim U[1, 4]$, $a \sim U[0.2, 0.5]$, $\phi \sim U[0, 2\pi]$.
4. A uniform electric field $\mathbb{E} = (E_x, E_y, 0)$ and magnetic field $\mathbf{B} = (0, 0, B_z)$; $E_x, E_y \sim U[-5, 5]$, $B_z \sim U[0, 10]$.

To control the difficulty of the tasks and avoid near-degenerate scenarios, we keep only mystery worlds satisfying the following two criteria: (1) At least 0.01 separation between all equations of motion (EOM) in the same world, defined as the Euclidean distance between the vectors of coefficients specifying the EOM difference equations, and (2) at least 0.0015 of any non-integer parameter from its nearest integer.

Trajectory simulation Within each world, we initialize the ball with a random position $(x, y) \sim U[-1, 1]^2$ and velocity $(v_0 \cos \theta_0, v_0 \sin \theta_0, 0)$; $v_0 \sim U[0.1, 0.5]$, $\theta_0 \sim U[0, 2\pi]$. We then compute its position for $N = 4,000$ time steps $t = 1, 2, \dots, N$ with time interval 0.05.

Although the above-mentioned laws of physics are linear, the mapping from past points $(\mathbf{y}_{t-T}, \dots, \mathbf{y}_{t-1})$ to the next points \mathbf{y}_t is generally non-linear because of region boundaries where the ball either bounces or transitions to a different physics region. An exception is when three successive points lie within the same region (with the same physics), which happens far from boundaries: in this case, the mapping from $(\mathbf{y}_{t-2}, \mathbf{y}_{t-1}) \mapsto \mathbf{y}_t$ is deterministic and linear thanks to the differential equations of motion being second-order and linear.

Architecture For the Newborn and AI Physicist agents, each prediction function f_i is implemented as a N_{lay}^f -layer neural network with linear activation, with N_{neur}^f -neuron hidden layers (we use $N_{lay}^f = 3$, $N_{neur}^f = 8$ for our main experiments; see Table A.3). Each domain sub-classifier c_i is implemented as a N_{lay}^c -layer neural net, with two hidden N_{neur}^c -neuron layers with leakyReLU activation $\sigma(x) = \max\{0.3x, x\}$, and the output layer having linear activation (we use $N_{lay}^c = 3$, $N_{neur}^c = 8$ for our main experiments). The baseline model is implemented as a single N_{lay}^f -layer neural net with two hidden 16-neuron layers with leakyReLU activation followed by a linear output layer. Note that for a fair comparison, the baseline model has more hidden neurons, to roughly compensate for the Newborn and AI Physicist agents typically having multiple theories. The baseline network is nonlinear to boost its expressive power for modeling the nonlinear prediction function of each world as a whole. For the domain classifier $\mathbf{c} = (c_1, c_2, \dots, c_M)$, it is a N_{lay}^c -layer neural net where each hidden layer has $N_{neur}^c = 8$ neurons and leakyReLU activation. The last layer has linear activation. See Table A.3 for a list of hyperparameters.

Evaluation The unsupervised classification accuracy is defined as the fraction of correctly classified points, using the permutation of the learned domain labels that best matches the hidden ground truth domain labels. It is “unsupervised” in the sense that there is no external supervision signal as to which domain label each point should be assigned to: the AI Physicist has to figure out the number of domains and their boundaries and assign each point to a domain, which is a difficult task.

We define a domain as *solved* if the agent discovers the its law of motion as difference equation (prediction function) within the following stringent tolerance: all rational coefficients in the difference equation are exactly matched, and all irrational coefficients agree to an

accuracy better than 10^{-4} . Because of the nature of the physics problems, some of these difference equation coefficients take on the values 0, -1 , or 2 , so solving a region requires successful integer snapping as described in Section 2.2.4. To make the problem even harder, we also fine-tune the magnetic field in five of the electromagnetic regions to make some of the coefficients simple fractions such as $1/3$ and $1/4$, thus making solving those regions contingent on successful rational snapping as described in Section 2.2.4. Domain solving can fail either by “undersnapping” (failing to approximate a floating-point number by a rational number) or ‘oversnapping’ (mistakenly rounding to a rational number). All our mystery worlds can be downloaded at space.mit.edu/home/tegmark/aiphysicist.html.

As shown in Appendix A.1.8, the only hard problem our AI Physicist or other algorithms need to solve is to determine the laws of motion away from domain boundaries. Therefore, we evaluate, tabulate and compare the performance of the algorithms only on interior points, *i.e.*, excluding data points $(\mathbf{x}_t, \mathbf{y}_t)$ straddling a boundary encounter.

Double Pendulum

Our double pendulum is implemented as two connected pendulums with massless rods of length 1 and that each have a point charge of 1 at their end. As illustrated in Figure 2-5, the system state is fully determined by the 4-tuple $\mathbf{y} = (\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2)$ and immersed in a piecewise constant electric field \mathbb{E} : $\mathbb{E} = (0, -E_1)$ in the upper half plane $y \geq -1.05$, and $\mathbb{E} = (0, E_2)$ in the lower half plane $y < -1.05$, using coordinates where y increases vertically and the origin is at the pivot point of the upper rod.

We generate 7 environments by setting (E_1, E_2) equal to $(E_0, 2E_0)$, $(E_0, 1.5E_0)$, (E_0, E_0) , $(E_0, 0.5E_0)$, $(2E_0, E_0)$, $(1.5E_0, E_0)$, and $(0.5E_0, E_0)$, where $E_0 = 9.8$. We see that there are two different EOMs for the double pendulum system depending on which of the two fields the lower charge is in (the upper charge is always in E_1). We use Runge-Kutta numerical integration to simulate $\mathbf{y} = (\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2)$ for 10,000 time steps with interval of 0.05, and the algorithms’ task is to predict the future (\mathbf{y}_{t+1}) based on the past ($\mathbf{x}_t \equiv \mathbf{y}_t$; history length $T = 1$), and simultaneously discover the two domains and their different EOMs unsupervised.

Regions	\log_{10} MSE			Classification accuracy			Unsolved domains			Description length		
	Base-line	New-born	AI phys	Base-line	New-born	AI phys	Base-line	New-born	AI phys	Base-line	New-born	AI phys
Free + gravity	-4.12	-14.07	-14.08	72.88%	100.00%	100.00%	2	0	0	11310.4	59.4	73.5
Free + gravity	-4.21	-14.02	-14.04	88.59%	100.00%	100.00%	2	0	0	11271.5	60.3	60.3
Free + gravity	-3.69	-14.03	-14.03	67.65%	100.00%	100.00%	2	0	0	11364.2	60.2	41.9
Free + gravity	-4.18	-13.98	-13.98	80.98%	100.00%	100.00%	2	0	0	11341.7	60.6	57.6
Free + gravity	-4.51	-14.06	-14.07	87.66%	100.00%	100.00%	2	0	0	11289.3	5.2	59.8
Free + harmonic	-3.77	-13.99	-13.94	73.54%	100.00%	100.00%	2	0	0	11333.8	94.4	139.9
Free + harmonic	-3.60	-14.05	-13.89	66.92%	100.00%	100.00%	2	0	0	11337.4	173.0	172.8
Free + harmonic	-3.77	-14.04	-13.95	59.46%	100.00%	100.00%	2	0	0	11317.5	156.0	173.8
Free + harmonic	-5.32	-10.48	-13.14	80.29%	100.00%	100.00%	2	1	0	11219.5	91.6	90.5
Free + harmonic	-3.64	-14.00	-13.89	71.70%	100.00%	100.00%	2	0	0	11369.6	143.7	136.6
Free + EM	-3.62	-13.95	-13.96	82.77%	100.00%	100.00%	2	0	0	11397.5	142.8	284.9
Free + EM	-4.13	-13.82	-13.67	76.55%	100.00%	100.00%	2	0	0	11283.0	306.2	306.2
Free + EM	-4.03	-13.45	-13.47	74.56%	99.97%	99.97%	2	0	0	11388.1	305.9	307.9
Free + EM	-4.31	-13.77	-13.62	86.68%	99.91%	99.91%	2	0	0	11257.7	152.0	133.5
Free + EM	-4.32	-14.00	-14.05	84.55%	100.00%	100.00%	2	0	0	11258.9	303.7	303.8
Free + EM rational	-3.45	-13.96	-13.95	77.88%	99.96%	99.93%	2	0	0	11414.9	194.2	195.8
Free + EM rational	-3.90	-13.96	-13.91	71.13%	100.00%	100.00%	2	0	0	11340.0	199.0	199.0
Free + EM rational	-4.12	-13.97	-13.90	72.78%	100.00%	100.00%	2	0	0	11330.7	198.8	198.8
Free + EM rational	-4.02	-14.07	-14.00	77.92%	100.00%	100.00%	2	0	0	11323.5	197.8	197.8
Free + EM rational	-4.83	-13.87	-13.86	91.14%	100.00%	100.00%	2	0	0	11247.1	10.3	13.9
Free + gravity + harmonic	-4.08	-14.03	-13.95	60.08%	100.00%	100.00%	3	0	0	11269.0	191.8	191.9
Free + gravity + harmonic	-4.31	-14.02	-13.66	63.01%	100.00%	100.00%	3	0	0	11334.2	170.4	83.1
Free + gravity + harmonic	-4.01	-14.01	-13.99	67.48%	100.00%	100.00%	3	0	0	11351.0	168.7	198.9
Free + gravity + harmonic	-3.64	-13.97	-13.88	60.02%	99.97%	99.93%	3	0	0	11374.6	225.7	225.7
Free + gravity + harmonic	-4.11	-7.42	-7.43	51.63%	100.00%	99.97%	3	1	1	11313.7	193.5	179.2
Free + gravity + EM	-3.79	-13.93	-13.47	57.89%	100.00%	100.00%	3	0	0	11334.0	323.9	346.8
Free + gravity + EM	-4.18	-14.00	-14.00	77.16%	100.00%	100.00%	3	1	1	11301.0	277.9	96.2
Free + gravity + EM	-3.38	-13.58	-13.87	53.33%	100.00%	99.96%	3	0	0	11381.4	360.4	364.0
Free + gravity + EM	-3.46	-13.87	-13.89	49.08%	100.00%	100.00%	3	0	0	11370.1	354.0	350.4
Free + gravity + EM	-3.54	-13.69	-13.83	51.28%	100.00%	100.00%	3	0	0	11370.3	331.1	320.7
Free + harmonic + EM	-3.87	-13.82	-13.55	67.27%	100.00%	100.00%	3	0	0	11404.0	267.1	275.4
Free + harmonic + EM	-3.69	-13.87	-10.93	56.02%	99.97%	99.94%	3	0	0	11413.4	468.5	464.9
Free + harmonic + EM	-4.06	-13.39	-13.56	70.87%	100.00%	100.00%	3	0	0	11340.0	452.3	452.3
Free + harmonic + EM	-3.46	-13.94	-10.51	59.02%	99.97%	99.93%	3	0	0	11416.0	475.5	471.9
Free + harmonic + EM	-3.70	-13.75	-13.82	61.67%	100.00%	100.00%	3	0	0	11354.9	466.8	466.8
Free + gravity + harmonic + EM	-3.76	-13.82	-9.48	27.93%	100.00%	99.94%	4	0	0	11358.8	526.9	530.4
Free + gravity + harmonic + EM	-3.74	-13.00	-13.18	40.80%	100.00%	99.97%	4	1	1	11284.8	418.5	389.1
Free + gravity + harmonic + EM	-4.09	-13.97	-13.75	35.69%	100.00%	100.00%	4	0	0	11297.4	504.6	504.6
Free + gravity + harmonic + EM	-3.63	-13.80	-9.99	31.61%	100.00%	99.97%	4	0	0	11407.4	526.3	526.2
Free + gravity + harmonic + EM	-3.51	-6.37	-13.52	32.97%	100.00%	100.00%	4	0	0	11445.8	527.4	527.5
Median	-3.89	-13.95	-13.88	67.56%	100.00%	100.00%	2.5	0.00	0.00	11338.7	198.9	198.9
Mean	-3.94	-13.44	-13.29	65.51%	99.99%	99.99%	2.6	0.10	0.07	11337.9	253.7	252.9

Table A.1: Results for each of our first 40 mystery world benchmarks, as described in the section A.1.9. Each number is the best out of ten trials with random initializations (using seeds 0, 30, 60, 90, 120, 150, 180, 210, 240, 270), and refers to big domains only. Based on the “Unsolved domain” column, we count out of 40 worlds what’s the percentage Baseline, Newborn and AI Physicist completely solve (has unsolved domain of 0), which goes to the “Fraction of worlds solved” row in Table 2.2.

Regions	Epochs to 10^{-2}			Epochs to 10^{-4}			Epochs to 10^{-6}			Epochs to 10^{-8}		
	Base-line	New-born	AI-phys									
Free+gravity	100	85	85	8440	120	120	∞	4175	3625	∞	6315	4890
Free+gravity	100	70	10	4680	190	35	∞	2900	4650	∞	2995	6500
Free+gravity	85	100	15	∞	135	30	∞	8205	3815	∞	9620	6455
Free+gravity	95	75	20	7495	140	25	∞	6735	1785	∞	8040	2860
Free+gravity	110	75	0	1770	295	35	∞	3740	3240	∞	7030	3460
Free + harmonic	80	75	20	∞	145	25	∞	2725	4050	∞	2830	6145
Free + harmonic	85	75	20	∞	80	25	∞	7965	1690	∞	10000	3400
Free + harmonic	95	75	30	∞	110	30	∞	1805	3895	∞	1855	3900
Free + harmonic	25	20	5	1285	460	10	∞	5390	1060	∞	7225	6385
Free + harmonic	80	95	5	∞	110	20	∞	4380	3300	∞	4800	4035
Free + EM	90	85	20	∞	1190	115	∞	6305	3380	∞	6590	3435
Free + EM	125	120	0	6240	885	70	∞	7310	1865	∞	7565	1865
Free + EM	115	115	15	15260	600	70	∞	2430	1225	∞	2845	4435
Free + EM	145	90	0	6650	140	0	∞	3000	5205	∞	4530	8735
Free + EM	80	80	10	965	200	25	∞	4635	1970	∞	4690	2870
Free + EM rational	80	75	0	∞	580	70	∞	5415	4150	∞	5445	4175
Free + EM rational	100	100	10	∞	460	45	∞	2560	965	∞	2575	5760
Free + EM rational	140	95	10	11050	455	65	∞	1960	1150	∞	6295	4005
Free + EM rational	120	100	5	13315	325	175	∞	3970	1290	∞	4335	3560
Free + EM rational	35	30	35	1155	335	35	∞	3245	2130	∞	5115	5610
Free + gravity + harmonic	150	75	25	9085	130	30	∞	3870	6145	∞	5555	6185
Free + gravity + harmonic	145	90	5	6915	140	25	∞	4525	3720	∞	10275	4430
Free + gravity + harmonic	105	100	15	6925	155	40	∞	6665	6560	∞	8915	6845
Free + gravity + harmonic	95	95	5	∞	120	30	∞	5790	10915	∞	18450	13125
Free + gravity + harmonic	135	95	15	7970	190	45	∞	13125	7045	∞	∞	∞
Free + gravity + EM	130	100	20	∞	575	40	∞	3215	5095	∞	3215	5100
Free + gravity + EM	125	110	15	5650	160	30	∞	6085	4720	∞	8025	4980
Free + gravity + EM	80	65	15	∞	630	120	∞	4100	6250	∞	4100	6570
Free + gravity + EM	80	75	5	∞	90	45	∞	5910	5815	∞	7295	6090
Free + gravity + EM	80	85	20	∞	1380	465	∞	2390	11425	∞	7450	11510
Free + harmonic + EM	85	75	25	∞	600	150	∞	3775	4525	∞	4675	5070
Free + harmonic + EM	85	90	25	∞	1245	200	∞	6225	2340	∞	6390	3180
Free + harmonic + EM	115	85	15	16600	190	35	∞	6035	1515	∞	10065	2110
Free + harmonic + EM	80	70	35	∞	720	195	∞	6990	3895	∞	6995	6115
Free + harmonic + EM	85	65	10	∞	985	165	∞	5660	1670	∞	5820	1820
Free + gravity + harmonic + EM	90	75	0	∞	540	255	∞	8320	7390	∞	9770	7590
Free + gravity + harmonic + EM	95	80	15	∞	1265	635	∞	6520	6365	∞	8475	6475
Free + gravity + harmonic + EM	130	85	10	8620	575	105	∞	6320	4035	∞	9705	7685
Free + gravity + harmonic + EM	75	80	0	∞	815	425	∞	7575	8405	∞	10440	8620
Free + gravity + harmonic + EM	80	65	20	∞	735	280	∞	6715	4555	∞	12495	8495
Median	95	83	15	∞	330	45	∞	5403	3895	∞	6590	5100
Mean	98	82	15	∞	455	109	∞	5217	4171	∞	6892	5499

Table A.2: Same as previous table, but showing number of training epochs required to reach various MSE prediction accuracies. We record the metrics every 5 epochs, so all the epochs are multiples of 5. Note that the AI Physicist has superseded 10^{-2} MSE already by 0 epochs for some environments, showing that thanks to the lifelong learning strategy which proposes previously learned theories in novel environments, reasonably good predictions can sometimes be achieved even without gradient descent training.

In this experiment, we implement prediction function of the Baseline and Newborn both as a N_{lay}^f -layer neural net (we use $N_{lay}^f = 6$) during DDAC. For the Newborn, each hidden layer has $N_{neur}^f = 160$ neurons with hyperbolic tangent (\tanh) activation, and for the Baseline, each hidden layer has $N_{neur}^f = 320$ neurons with \tanh activation for a fair comparison. For the Newborn, the optional AddTheories(\mathcal{T}, D) (step s8 in Alg. 7) is turned off to prevent unlimited adding of theories. The initial number M of theories for Newborn is set to $M = 2$ and $M = 3$, each run with 10 instances with random initialization. Its domain classifier $\mathbf{c} = (c_1, c_2, \dots, c_M)$ is a N_{lay}^c -layer neural net (we use $N_{lay}^c = 3$) where each hidden layer has $N_{neur}^c = 6$ neurons and leakyReLU activation. The last layer has linear activation.

	Hyperparameter	Environments	Baseline	Newborn	AI Physicist
γ	Generalized-mean-loss exponent	All	-1	-1	-1
β_f	Initial learning rate for f_θ	All	0.005	0.005	0.005
β_c	Initial learning rate for c_ϕ	All	0.001	0.001	0.001
K	Number of gradient iterations	All	10000	10000	10000
σ_c	Hidden layer activation function in c_ϕ	All	-	leakyReLU	leakyReLU
N_{lay}^c	Number of layers in c_ϕ	All	-	3	3
C	Initial number of clusters in theory unification	All	4	4	4
ϵ_{MSE}	MSE regularization strength	All	10^{-7}	10^{-7}	10^{-7}
ϵ_{L1}	Final L_1 regularization strength	Mystery worlds Double Pendulum	10^{-8} 10^{-7}	10^{-8} 10^{-7}	10^{-8} 10^{-7}
N_{lay}^f	Number of layers in f_θ	Mystery worlds Double Pendulum	3 6	3 6	3 6
N_{neur}^f	Number of neurons in f_θ	Mystery worlds Double Pendulum	16 320	8 160	8 -
N_{neur}^c	Number of neurons in c_ϕ	Mystery worlds Double Pendulum	- -	8 6	8 -
T	Maximum time horizon for input	Mystery worlds Double Pendulum	2 1	2 1	2 1
σ_f	Hidden layer activation function in f_θ	Mystery worlds Double Pendulum	leakyReLU tanh	linear tanh	linear -
M_0	Initial number of theories	Mystery worlds Double Pendulum	1 1	2 2 & 3	2 -
M	Maximum number of theories	Mystery worlds Double Pendulum	1 1	4 2 & 3	4 -
ϵ_{add}	MSE threshold for theory adding	Mystery worlds Double Pendulum	- -	2×10^{-6} ∞	2×10^{-6} -
η_{insp}	Inspection threshold for theory adding	Mystery worlds Double Pendulum	- -	30% ∞	30% -
η_{split}	Splitting threshold for theory adding	Mystery worlds Double Pendulum	- -	5% ∞	5% -
η_{del}	Fraction threshold for theory deletion	Mystery worlds Double Pendulum	- -	0.5% 100%	0.5% -

Table A.3: Hyperparameter settings in the numerical experiments. For a fair comparison between Baseline and the other agents that can have up to 4 theories, the number of neurons in each layer of Baseline is larger so that the total number of parameters is roughly the same for all agents. The Baseline agent in Mystery worlds has leakyReLU activation to be able to account for different domains.

Algorithm 7 AI Physicist: Differentiable Divide-and-Conquer with Harmonic Loss

Require Dataset $\mathbf{D} = \{(\mathbf{x}_t, \mathbf{y}_t)\}$
Require M : number of initial total theories for training
Require $\mathcal{T}_{M_0} = \{(\mathbf{f}_i, c_i)\}, i = 1, \dots, M_0, 0 \leq M_0 \leq M$: theories proposed from theory hub
Require K : number of gradient iterations
Require β_f, β_c : learning rates
Require ϵ_0 : initial precision floor

1: Randomly initialize $M - M_0$ theories $\mathbf{T}_i, i = M_0 + 1, \dots, M$. Denote $\mathcal{T} = (\mathbf{T}_1, \dots, \mathbf{T}_M)$, $\mathbf{f}_\theta = (\mathbf{f}_1, \dots, \mathbf{f}_M)$, $\mathbf{c}_\phi = (c_1, \dots, c_M)$ with learnable parameters θ and ϕ .

// Harmonic training with DL loss:

2: $\epsilon \leftarrow \epsilon_0$
3: **for** k in $\{1, 2, 3, 4, 5\}$ **do**:
4: $\mathcal{T} \leftarrow \text{IterativeTrain}(\mathcal{T}, \mathbf{D}, \ell_{\text{DL}, \epsilon}, \mathcal{L}_{-1})$, where

$$\mathcal{L}_{-1} \equiv \sum_t \left(\frac{1}{M} \sum_{i=1}^M \ell[\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t]^{-1} \right)^{-1} \quad (\text{Eq. 2.2})$$

5: $\epsilon \leftarrow \text{set_epsilon}(\mathcal{T}, \mathbf{D})$ // median prediction error
6: **end for**

// Fine-tune each theory and its domain:

7: **for** k in $\{1, 2\}$ **do**:
8: $\mathcal{T} \leftarrow \text{IterativeTrain}(\mathcal{T}, \mathbf{D}, \ell_{\text{DL}, \epsilon}, \mathcal{L}_{\text{dom}})$, where

$$\mathcal{L}_{\text{dom}} \equiv \sum_t \ell[\mathbf{f}_{i_t}(\mathbf{x}_t), \mathbf{y}_t] \text{ with } i_t = \arg \max_i \mathbf{c}_i(\mathbf{x}_t)$$

9: $\epsilon \leftarrow \text{set_epsilon}(\mathcal{T}, \mathbf{D})$ // median prediction error
10: **end for**
11: **return** \mathcal{T}

subroutine $\text{IterativeTrain}(\mathcal{T}, \mathbf{D}, \ell, \mathcal{L})$:
s1: **for** k in $\{1, \dots, K\}$ **do**:
 // Gradient descent on \mathbf{f}_θ with loss \mathcal{L} :
s2: $\mathbf{g}_f \leftarrow \nabla_\theta \mathcal{L}[\mathcal{T}, \mathbf{D}, \ell]$
s3: Update θ using gradients \mathbf{g}_f (e.g. Adam [KB14] or SGD)
 // Gradient descent on \mathbf{c}_ϕ with the best performing theory index as target:
s4: $b_t \leftarrow \arg \min_i \{\ell[\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t]\}, \forall t$
s5: $\mathbf{g}_c \leftarrow \nabla_\phi \sum_{(\mathbf{x}_t, \cdot) \in \mathbf{D}} \text{CrossEntropy}[\text{softmax}(\mathbf{c}_\phi(\mathbf{x}_t)), b_t]$
s6: Update ϕ using gradients \mathbf{g}_c (e.g. Adam [KB14] or SGD)
s7: **end for**
s8: $\mathcal{T} \leftarrow \text{AddTheories}(\mathcal{T}, \mathbf{D}, \ell, \mathcal{L})$ //Optional
s9: $\mathcal{T} \leftarrow \text{DeleteTheories}(\mathcal{T}, \mathbf{D}, \ell)$ //Optional
s10: **return** \mathcal{T}

Algorithm 8 AI Physicist: Occam's Razor with MDL

Require Dataset $D = \{(\mathbf{x}_t, \mathbf{y}_t)\}$
Require $\mathcal{T} = \{(\mathbf{f}_i, c_i)\}, i = 1, \dots, M$: theories trained after Alg. 7
Require ϵ : Precision floor for $\ell_{\text{DL},\epsilon}$

1: **for** i in $\{1, \dots, M\}$ **do**:

2: $D^{(i)} \leftarrow \{(\mathbf{x}_t, \mathbf{y}_t) | \arg \max_j \{c_j(\mathbf{x}_t)\} = i\}$

3: $\mathbf{f}_i \leftarrow \text{MinimizeDL}(\text{collapseLayers}, \mathbf{f}_i, D^{(i)}, \epsilon)$

4: $\mathbf{f}_i \leftarrow \text{MinimizeDL}(\text{localSnap}, \mathbf{f}_i, D^{(i)}, \epsilon)$

5: $\mathbf{f}_i \leftarrow \text{MinimizeDL}(\text{integerSnap}, \mathbf{f}_i, D^{(i)}, \epsilon)$

6: $\mathbf{f}_i \leftarrow \text{MinimizeDL}(\text{rationalSnap}, \mathbf{f}_i, D^{(i)}, \epsilon)$

7: $\mathbf{f}_i \leftarrow \text{MinimizeDL}(\text{toSymbolic}, \mathbf{f}_i, D^{(i)}, \epsilon)$

8: **end for**

9: **return** \mathcal{T}

subroutine $\text{MinimizeDL}(\text{transformation}, \mathbf{f}_i, D^{(i)}, \epsilon)$:

s1: **while** $\text{transformation.is_applicable}(\mathbf{f}_i)$ **do**:

s2: $dl_0 \leftarrow \text{DL}(\mathbf{f}_i) + \sum_{(\mathbf{x}_t, \mathbf{y}_t) \in D^{(i)}} \ell_{\text{DL},\epsilon}[\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t]$

s3: $f_{\text{clone}} \leftarrow \mathbf{f}_i // \text{clone } \mathbf{f}_i \text{ in case transformation fails}$

s4: $\mathbf{f}_i \leftarrow \text{transformation}(\mathbf{f}_i)$

s5: $\mathbf{f}_i \leftarrow \text{Minimize}_{\mathbf{f}_i} \sum_{(\mathbf{x}_t, \mathbf{y}_t) \in D^{(i)}} \ell_{\text{DL},\epsilon}[\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t]$

s6: $dl_1 \leftarrow \text{DL}(\mathbf{f}_i) + \sum_{(\mathbf{x}_t, \mathbf{y}_t) \in D^{(i)}} \ell_{\text{DL},\epsilon}[\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t]$

s7: **if** $dl_1 > dl_0$ **return** f_{clone}

s8: **end while**

s9: **return** \mathbf{f}_i

Algorithm 9 AI Physicist: Theory Unification

Require Hub: theory hub
Require C : initial number of clusters
1: **for** (\mathbf{f}_i, c_i) **in** Hub.all-symbolic-theories **do**:
2: $\text{dl}^{(i)} \leftarrow \text{DL}(\mathbf{f}_i)$
3: **end for**
4: $\{S_k\} \leftarrow$ Cluster $\{\mathbf{f}_i\}$ into C clusters based on $\text{dl}^{(i)}$
5: **for** S_k **in** $\{S_k\}$ **do**:
6: $(\mathbf{g}_{i_k}, \mathbf{h}_{i_k}) \leftarrow \text{Canonicalize}(\mathbf{f}_{i_k}), \forall \mathbf{f}_{i_k} \in S_k$
7: $\mathbf{h}_k^* \leftarrow \text{Mode of } \{\mathbf{h}_{i_k} | \mathbf{f}_{i_k} \in S_k\}$.
8: $G_k \leftarrow \{\mathbf{g}_{i_k} | \mathbf{h}_{i_k} = \mathbf{h}_k^*\}$
9: $\mathbf{g}_{p_k} \leftarrow$ Traverse all $\mathbf{g}_{i_k} \in G_k$ with synchronized steps,
 replacing the coefficient by a p_{j_k} when not all
 coefficients at the same position are identical.
10: $\mathbf{f}_{p_k} \leftarrow \text{toPlainForm}(\mathbf{g}_{p_k})$
11: **end for**
12: $\mathcal{T} \leftarrow \{(\mathbf{f}_{p_k}, \cdot)\}, k = 1, 2, \dots C$
13: $\mathcal{T} \leftarrow \text{MergeSameForm}(\mathcal{T})$
14: **return** \mathcal{T}

subroutine **Canonicalize**(\mathbf{f}_i):

s1: $\mathbf{g}_i \leftarrow \text{ToTreeForm}(\mathbf{f}_i)$
s2: $\mathbf{h}_i \leftarrow$ Replace all non-input coefficient by a symbol s
return $(\mathbf{g}_i, \mathbf{h}_i)$

Algorithm 10 AI Physicist: Theory Proposing from Hub

Require Hub: theory hub
Require Dataset $D = \{(\mathbf{x}_t, \mathbf{y}_t)\}$
Require M_0 : number of theories to propose from the hub
1: $\{(\mathbf{f}_i, c_i)\} \leftarrow \text{Hub.retrieve-all-theories}()$
2: $D_{\text{best}}^{(i)} \leftarrow \{(\mathbf{x}_t, \mathbf{y}_t) | \text{argmin}_j \ell_{\text{DL}, \epsilon}[\mathbf{f}_j(\mathbf{x}_t), \mathbf{y}_t] = i\}, \forall i$
3: $\mathcal{T}_{M_0} \leftarrow \{(\mathbf{f}_i, c_i) | D_{\text{best}}^{(i)} \text{ ranks among } M_0 \text{ largest sets in } \{D_{\text{best}}^{(i)}\}\}$
4: **return** \mathcal{T}_{M_0}

Algorithm 11 AI Physicist: Adding Theories to Hub

Require **Hub**: theory hub
Require $\mathcal{T} = \{(\mathbf{f}_i, c_i)\}$: Trained theories from Alg. 7
Require Dataset $\mathbf{D} = \{(\mathbf{x}_t, \mathbf{y}_t)\}$
Require η : DL threshold for adding theories to hub

- 1: $D^{(i)} \leftarrow \{(\mathbf{x}_t, \mathbf{y}_t) | \arg \max_j \{c_j(\mathbf{x}_t)\} = i\}, \forall i$
- 2: $dl^{(i)} \leftarrow \frac{1}{|D^{(i)}|} \sum_{(\mathbf{x}_t, \mathbf{y}_t) \in D^{(i)}} \ell_{DL, \epsilon}(\mathbf{f}_i(\mathbf{x}_t), \mathbf{y}_t), \forall i$
- 3: **for** i **in** $\{1, 2, \dots, |\mathcal{T}|\}$ **do**:
- 4: **if** $dl^{(i)} < \eta$ **do**
- 5: **Hub.addIndividualTheory** $((\mathbf{f}_i, c_i), D^{(i)})$
- 6: **end if**
- 7: **end for**

A.2 Appendix for Chapter 3

The structure of the Appendix is as follows. In Section A.2.1, we provide preliminaries for the first-order and second-order variations on functionals. We prove Theorem 1.1 and Theorem 2 in Section A.2.2 and A.2.3, respectively. In Section A.2.4, we prove Theorem 3, the sufficient condition 1 for IB-Learnability. In Section A.2.5, we calculate the first and second variations of $\text{IB}_\beta[p(z|x)]$ at the trivial representation $p(z|x) = p(z)$, which is used in proving the Sufficient Condition 2 for IB_β -learnability (Section A.2.7). In Appendix A.2.8, we prove Eq. (3.3) at the onset of learning. After these preparations, we prove the key result of this paper, Theorem 5, in Section A.2.9. Then two important corollaries 5.1, 5.2 are proved in Section A.2.10. In Section A.2.11 we explore the deep relation between $\beta_0, \beta_0[h(x)]$, the hypercontractivity coefficient, contraction coefficient and maximum correlation. Finally in Section A.2.12, we provide details for the experiments.

A.2.1 Preliminaries: first-order and second-order variations

Let functional $F[f(x)]$ be defined on some normed linear space \mathcal{R} . Let us add a perturbative function $\epsilon \cdot h(x)$ to $f(x)$, and now the functional $F[f(x) + \epsilon \cdot h(x)]$ can be expanded as

$$\begin{aligned}\Delta F[f(x)] &= F[f(x) + \epsilon \cdot h(x)] - F[f(x)] \\ &= \varphi_1[f(x)] + \varphi_2[f(x)] + \mathcal{O}(\epsilon^3 \|h\|^2)\end{aligned}$$

where $\|h\|$ denotes the norm of h , $\varphi_1[f(x)] = \epsilon \frac{dF[f(x)]}{d\epsilon}$ is a linear functional of $\epsilon \cdot h(x)$, and is called the *first-order variation*, denoted as $\delta F[f(x)]$. $\varphi_2[f(x)] = \frac{1}{2}\epsilon^2 \frac{d^2F[f(x)]}{d\epsilon^2}$ is a quadratic functional of $\epsilon \cdot h(x)$, and is called the *second-order variation*, denoted as $\delta^2 F[f(x)]$.

If $\delta F[f(x)] = 0$, we call $f(x)$ a stationary solution for the functional $F[\cdot]$.

If $\Delta F[f(x)] \geq 0$ for all $h(x)$ such that $f(x) + \epsilon \cdot h(x)$ is at the neighborhood of $f(x)$, we call $f(x)$ a (local) minimum of $F[\cdot]$.

A.2.2 Proof of Lemma 1.1

Proof. If (X, Y) is IB_β -learnable, then there exists $Z \in \mathcal{Z}$ given by some $p_1(z|x)$ such that $\text{IB}_\beta(X, Y; Z) < \text{IB}(X, Y; Z_{\text{trivial}}) = 0$, where Z_{trivial} satisfies $p(z|x) = p(z)$. Since $X' = g(X)$ is a invertible map (if X is continuous variable, g is additionally required to be continuous), and mutual information is invariant under such an invertible map ([KSG04]), we have that $\text{IB}_\beta(X', Y; Z) = I(X'; Z) - \beta I(Y; Z) = I(X; Z) - \beta I(Y; Z) = \text{IB}_\beta(X, Y; Z) < 0 = \text{IB}(X', Y; Z_{\text{trivial}})$, so (X', Y) is IB_β -learnable. On the other hand, if (X, Y) is not IB_β -learnable, then $\forall Z$, we have $\text{IB}_\beta(X, Y; Z) \geq \text{IB}(X, Y; Z_{\text{trivial}}) = 0$. Again using mutual information's invariance under g , we have for all Z , $\text{IB}_\beta(X', Y; Z) = \text{IB}_\beta(X, Y; Z) \geq \text{IB}(X, Y; Z_{\text{trivial}}) = 0$, leading to that (X', Y) is not IB_β -learnable. Therefore, we have that (X, Y) and (X', Y) have the same IB_β -learnability. ■

A.2.3 Proof of Theorem 2

Proof. At the trivial representation $p(z|x) = p(z)$, we have $I(X; Z) = 0$, and $I(Y; Z) = 0$ due to the Markov chain, so $\text{IB}_\beta(X, Y; Z)|_{p(z|x)=p(z)} = 0$ for any β . Since (X, Y) is IB_{β_1} -learnable, there exists a Z given by a $p_1(z|x)$ such that $\text{IB}_{\beta_1}(X, Y; Z)|_{p_1(z|x)} < 0$. Since $\beta_2 > \beta_1$, and $I(Y; Z) \geq 0$, we have $\text{IB}_{\beta_2}(X, Y; Z)|_{p_1(z|x)} \leq \text{IB}_{\beta_1}(X, Y; Z)|_{p_1(z|x)} < 0 = \text{IB}_{\beta_2}(X, Y; Z)|_{p(z|x)=p(z)}$. Therefore, (X, Y) is IB_{β_2} -learnable. ■

A.2.4 Proof of Theorem 3

Proof. To prove Theorem 3, we use the Theorem 1 of Chapter 5 of [GS⁺00] which gives a necessary condition for $F[f(x)]$ to have a minimum at $f_0(x)$. Adapting to our notation, we have:

Theorem 13 ([GS⁺00]). *A necessary condition for the functional $F[f(x)]$ to have a minimum at $f(x) = f_0(x)$ is that for $f(x) = f_0(x)$ and all admissible $\epsilon \cdot h(x)$,*

$$\delta^2 F[f(x)] \geq 0$$

Applying to our functional $\text{IB}_\beta[p(z|x)]$, an immediate result of Theorem 13 is that, if at $p(z|x) = p(z)$, there exists an $\epsilon \cdot h(z|x)$ such that $\delta^2 \text{IB}_\beta[p(z|x)] < 0$, then $p(z|x) = p(z)$ is not a minimum for $\text{IB}_\beta[p(z|x)]$. Using the definition of IB_β learnability, we have that (X, Y) is IB_β -learnable.

■

A.2.5 First- and second-order variations of $\text{IB}_\beta[p(z|x)]$

In this section, we derive the first- and second-order variations of $\text{IB}_\beta[p(z|x)]$, which are needed for proving Lemma 2.1 and Theorem 4.

Lemma 13.1. *Using perturbative function $h(z|x)$, we have*

$$\begin{aligned} \delta \text{IB}_\beta[p(z|x)] &= \int dx dz p(x) h(z|x) \log \frac{p(z|x)}{p(z)} - \beta \int dx dy dz p(x, y) h(z|x) \log \frac{p(z|y)}{p(z)} \\ \delta^2 \text{IB}_\beta[p(z|x)] &= \\ &\frac{1}{2} \left[\int dx dz \frac{p(x)^2}{p(x, z)} h(z|x)^2 - \beta \int dx dx' dy dz \frac{p(x, y)p(x', y)}{p(y, z)} h(z|x) h(z|x') + \right. \\ &\left. (\beta - 1) \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x) h(z|x') \right] \end{aligned}$$

Proof. Since $\text{IB}_\beta[p(z|x)] = I(X; Z) - \beta I(Y; Z)$, let us calculate the first and second-order variation of $I(X; Z)$ and $I(Y; Z)$ w.r.t. $p(z|x)$, respectively. Through this derivation, we use $\epsilon \cdot h(z|x)$ as a perturbative function, for ease of deciding different orders of variations. We assume that $h(z|x)$ is continuous, and there exists a constant M such that $\left| \frac{h(z|x)}{p(z|x)} \right| < M$, $\forall (x, z) \in \mathcal{X} \times \mathcal{Z}$. We will finally absorb ϵ into $h(z|x)$.

Denote $I(X; Z) = F_1[p(z|x)]$. We have

$$F_1[p(z|x)] = I(X; Z) = \int dx dz p(z|x) p(x) \log \frac{p(z|x)}{p(z)}$$

In this paper, we implicitly assume that the integral (or summing) are only on the support of $p(x, y, z)$.

Since

$$p(z) = \int p(z|x)p(x)dx$$

We have

$$p(z)|_{p(z|x)+\epsilon h(z|x)} = p(z)|_{p(z|x)} + \epsilon \int h(z|x)p(x)dx$$

Expanding $F_1[p(z|x) + \epsilon h(z|x)]$ to the second order of ϵ , we have

$$\begin{aligned} & F_1[p(z|x) + \epsilon h(z|x)] \\ &= \int dxdz p(x)[p(z|x) + \epsilon h(z|x)] \log \frac{p(z|x) + \epsilon h(z|x)}{p(z) + \epsilon \int h(z|x')p(x')dx'} \\ &= \int dxdz p(x)p(z|x) \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right) \log \frac{p(z|x) \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right)}{p(z) \left(1 + \epsilon \frac{\int h(z|x')p(x')dx'}{p(z)}\right)} \\ &= \int dxdz p(x)p(z|x) \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right) \log \left[\frac{p(z|x)}{p(z)} \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right) \left(1 - \epsilon \frac{\int h(z|x')p(x')dx'}{p(z)} \right. \right. \\ &\quad \left. \left. + \epsilon^2 \left(\frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 \right) \right] + \mathcal{O}(\epsilon^3) \\ &= \int dxdz p(x)p(z|x) \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right) \log \left[\frac{p(z|x)}{p(z)} \left(1 + \epsilon \left(\frac{h(z|x)}{p(z|x)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right) \right. \right. \\ &\quad \left. \left. + \epsilon^2 \left(\frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 - \epsilon^2 \frac{h(z|x)}{p(z|x)} \frac{\int h(z|x')p(x')dx'}{p(z)} \right) \right] + \mathcal{O}(\epsilon^3) \\ &= \int dxdz p(x)p(z|x) \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right) \left[\log \frac{p(z|x)}{p(z)} + \epsilon \left(\frac{h(z|x)}{p(z|x)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right) \right. \\ &\quad \left. + \epsilon^2 \left(\frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 - \epsilon^2 \frac{h(z|x)}{p(z|x)} \frac{\int h(z|x')p(x')dx'}{p(z)} - \frac{1}{2}\epsilon^2 \left(\frac{h(z|x)}{p(z|x)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 \right] + \mathcal{O}(\epsilon^3) \end{aligned}$$

Collecting the first order terms of ϵ , we have

$$\begin{aligned}
& \delta F_1[p(z|x)] \\
&= \epsilon \int dx dz p(x)p(z|x) \left(\frac{h(z|x)}{p(z|x)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right) + \epsilon \int dx dz p(x)p(z|x) \frac{h(z|x)}{p(z|x)} \log \frac{p(z|x)}{p(z)} \\
&= \epsilon \int dx dz p(x)h(z|x) - \epsilon \int dx' dz p(x')h(z|x') + \epsilon \int dx dz p(x)h(z|x) \log \frac{p(z|x)}{p(z)} \\
&= \epsilon \int dx dz p(x)h(z|x) \log \frac{p(z|x)}{p(z)}
\end{aligned}$$

Collecting the second order terms of ϵ^2 , we have

$$\begin{aligned}
& \delta^2 F_1[p(z|x)] \\
&= \epsilon^2 \int dx dz p(x)p(z|x) \left[\left(\frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 - \frac{h(z|x)}{p(z|x)} \frac{\int h(z|x')p(x')dx'}{p(z)} \right. \\
&\quad \left. - \frac{1}{2} \left(\frac{h(z|x)}{p(z|x)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 \right] \\
&\quad + \epsilon^2 \int dx dz p(x)p(z|x) \frac{h(z|x)}{p(z|x)} \left(\frac{h(z|x)}{p(z|x)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right) \\
&= \frac{\epsilon^2}{2} \int dx dz \frac{p(x)^2}{p(x,z)} h(z|x)^2 - \frac{\epsilon^2}{2} \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')
\end{aligned}$$

Now let us calculate the first and second-order variation of $F_2[p(z|x)] = I(Z; Y)$. We have

$$F_2[p(z|x)] = I(Y; Z) = \int dy dz p(z|y)p(y) \log \frac{p(y, z)}{p(y)p(z)} = \int dx dy dz p(z|y)p(x, y) \log \frac{p(y, z)}{p(y)p(z)}$$

Using the Markov chain $Z \leftarrow X \leftrightarrow Y$, we have

$$p(y, z) = \int p(z|x)p(x, y)dx$$

Hence

$$p(y, z)|_{p(z|x)+\epsilon h(z|x)} = p(y, z)|_{p(z|x)} + \epsilon \int h(z|x)p(x, y)dx$$

Then expanding $F_2[p(z|x) + \epsilon h(z|x)]$ to the second order of ϵ , we have

$$\begin{aligned}
& F_2[p(z|x) + \epsilon h(z|x)] \\
&= \int dxdydz p(x,y)p(z|x) \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right) \log \frac{p(y,z) \left(1 + \epsilon \frac{\int h(z|x')p(x',y)dx'}{p(y,z)}\right)}{p(y)p(z)(1 + \epsilon \frac{\int h(z|x'')p(x'')dx''}{p(z)})} \\
&= \int dxdydz p(x,y)p(z|x) \left(1 + \epsilon \frac{h(z|x)}{p(z|x)}\right) \left\{ \log \frac{p(y,z)}{p(y)p(z)} \right. \\
&\quad + \epsilon \left(\frac{\int h(z|x')p(x',y)dx'}{p(y,z)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right) \\
&\quad + \epsilon^2 \left[\left(\frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 - \frac{\int h(z|x')p(x',y)dx'}{p(y,z)} \frac{\int h(z|x'')p(x'')dx''}{p(z)} \right. \\
&\quad \left. \left. - \frac{1}{2} \left(\frac{\int h(z|x')p(x',y)dx'}{p(y,z)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 \right] \right\} \\
&\quad + \mathcal{O}(\epsilon^3)
\end{aligned}$$

Collecting the first order terms of ϵ , we have

$$\begin{aligned}
& \delta F_2[p(z|x)] \\
&= \epsilon \int dxdydz p(x,y)h(z|x) \log \frac{p(y,z)}{p(y)p(z)} + \epsilon \int dxdydz p(x,y)p(z|x) \frac{\int h(z|x')p(x',y)dx'}{p(y,z)} \\
&\quad - \epsilon \int dxdydz p(x,y)p(z|x) \frac{\int h(z|x')p(x')dx'}{p(z)} \\
&= \epsilon \int dxdydz p(x,y)h(z|x) \log \frac{p(y,z)}{p(y)p(z)} + \epsilon \int dx'dydz h(z|x')p(x',y) - \epsilon \int dz h(z|x')p(x')dx' \\
&= \epsilon \int dxdydz p(x,y)h(z|x) \log \frac{p(z|y)}{p(z)}
\end{aligned}$$

Collecting the second order terms of ϵ , we have

$$\begin{aligned}
& \delta^2 F_2[p(z|x)] \\
&= \epsilon^2 \int dxdydz p(x,y)p(z|x) \left[\left(\frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 - \frac{\int h(z|x')p(x',y)dx'}{p(y,z)} \frac{\int h(z|x'')p(x'')dx''}{p(z)} \right] \\
&\quad - \frac{\epsilon^2}{2} \int dxdydz p(x,y)p(z|x) \left(\frac{\int h(z|x')p(x',y)dx'}{p(y,z)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right)^2 \\
&\quad + \epsilon^2 \int dxdydz p(x,y)p(z|x) \frac{h(z|x)}{p(z|x)} \left(\frac{\int h(z|x')p(x',y)dx'}{p(y,z)} - \frac{\int h(z|x')p(x')dx'}{p(z)} \right) \\
&= \frac{\epsilon^2}{2} \int dxdx'dydz \frac{p(x,y)p(x',y)}{p(y,z)} h(z|x)h(z|x') - \frac{\epsilon^2}{2} \int dxdx'dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')
\end{aligned}$$

Finally, we have

$$\begin{aligned}
\delta \text{IB}_\beta[p(z|x)] &= \delta F_1[p(z|x)] - \beta \cdot \delta F_2[p(z|x)] \\
&= \epsilon \left(\int dxdz p(x)h(z|x) \log \frac{p(z|x)}{p(z)} - \beta \int dxdydz p(x,y)h(z|x) \log \frac{p(z|y)}{p(z)} \right)
\end{aligned} \tag{A.6}$$

$$\begin{aligned}
\delta^2 \text{IB}_\beta[p(z|x)] &= \delta^2 F_1[p(z|x)] - \beta \cdot \delta^2 F_2[p(z|x)] \\
&= \frac{\epsilon^2}{2} \int dxdz \frac{p(x)^2}{p(x,z)} h(z|x)^2 - \frac{\epsilon^2}{2} \int dxdx'dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x') \\
&\quad - \beta \epsilon^2 \left[\frac{1}{2} \int dxdx'dydz \frac{p(x,y)p(x',y)}{p(y,z)} h(z|x)h(z|x') \right. \\
&\quad \left. - \frac{1}{2} \int dxdx'dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x') \right] \\
&= \frac{\epsilon^2}{2} \left[\int dxdz \frac{p(x)^2}{p(x,z)} h(z|x)^2 \right. \\
&\quad \left. - \beta \int dxdx'dydz \frac{p(x,y)p(x',y)}{p(y,z)} h(z|x)h(z|x') \right. \\
&\quad \left. + (\beta - 1) \int dxdx'dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x') \right]
\end{aligned}$$

Absorb ϵ into $h(z|x)$, we get rid of the ϵ factor and obtain the final expression in Lemma 13.1.

■

A.2.6 Proof of Lemma 2.1

Proof. Using Lemma 13.1, we have

$$\delta \text{IB}_\beta[p(z|x)] = \int dx dz p(x) h(z|x) \log \frac{p(z|x)}{p(z)} - \beta \int dx dy dz p(x, y) h(z|x) \log \frac{p(z|y)}{p(z)}$$

Let $p(z|x) = p(z)$ (the trivial representation), we have that $\log \frac{p(z|x)}{p(z)} \equiv 0$. Therefore, the two integrals are both 0. Hence,

$$\delta \text{IB}_\beta[p(z|x)] \Big|_{p(z|x)=p(z)} \equiv 0$$

Therefore, the $p(z|x) = p(z)$ is a stationary solution for $\text{IB}_\beta[p(z|x)]$.

■

A.2.7 Proof of Theorem 4

Proof. Firstly, from the necessary condition of $\beta > 1$ in Section 3.3, we have that any sufficient condition for IB_β -learnability should be able to deduce $\beta > 1$.

Now using Theorem 3, a sufficient condition for (X, Y) to be IB_β -learnable is that there exists $h(z|x)$ with $\int h(z|x)dx = 0$ such that $\delta^2 \text{IB}_\beta[p(z|x)] < 0$ at $p(z|x) = p(x)$.

At the trivial representation, $p(z|x) = p(z)$ and hence $p(x, z) = p(x)p(z)$. Due to the Markov chain $Z \leftarrow X \leftrightarrow Y$, we have $p(y, z) = p(y)p(z)$. Substituting them into the $\delta^2 \text{IB}_\beta[p(z|x)]$ in Lemma 13.1, the condition becomes: there exists $h(z|x)$ with $\int h(z|x)dz = 0$, such that

$$\begin{aligned} 0 &> \delta^2 \text{IB}_\beta[p(z|x)] = \\ &\frac{1}{2} \left[\int dx dz \frac{p(x)^2}{p(x)p(z)} h(z|x)^2 - \beta \int dx dx' dy dz \frac{p(x, y)p(x', y)}{p(y)p(z)} h(z|x)h(z|x') \right. \\ &\quad \left. + (\beta - 1) \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x') \right] \end{aligned} \quad (\text{A.7})$$

Rearranging terms and simplifying, we have

$$\begin{aligned} \int \frac{dz}{p(z)} G[h(z|x)] &= \int \frac{dz}{p(z)} \left[\int dx h(z|x)^2 p(x) - \beta \int \frac{dy}{p(y)} \left(\int dx h(z|x)p(x)p(y|x) \right)^2 \right. \\ &\quad \left. + (\beta - 1) \left(\int dx h(z|x)p(x) \right)^2 \right] < 0 \end{aligned}$$

where

$$G[h(x)] = \int dx h(x)^2 p(x) - \beta \int \frac{dy}{p(y)} \left(\int dx h(x)p(x)p(y|x) \right)^2 + (\beta - 1) \left(\int dx h(x)p(x) \right)^2$$

Now we prove that the condition that $\exists h(z|x)$ s.t. $\int \frac{dz}{p(z)} G[h(z|x)] < 0$ is equivalent to the condition that $\exists h(x)$ s.t. $G[h(x)] < 0$.

If $\forall h(z|x)$, $G[h(z|x)] \geq 0$, then we have $\forall h(z|x)$, $\int \frac{dz}{p(z)} G[h(z|x)] \geq 0$. Therefore, if $\exists h(z|x)$ s.t. $\int \frac{dz}{p(z)} G[h(z|x)] < 0$, we have that $\exists h(z|x)$ s.t. $G[h(z|x)] < 0$. Since the functional $G[h(z|x)]$ does not contain integration over z , we can treat the z in $G[h(z|x)]$ as a parameter and we have that $\exists h(x)$ s.t. $G[h(x)] < 0$.

Conversely, if there exists an certain function $h(x)$ such that $G[h(x)] < 0$, we can find some $h_2(z)$ such that $\int h_2(z) dz = 0$ and $\int \frac{h_2^2(z)}{p(z)} dz > 0$, and let $h_1(z|x) = h(x)h_2(z)$. Now we have

$$\int \frac{dz}{p(z)} G[h(z|x)] = \int \frac{h_2^2(z) dz}{p(z)} G[h(x)] = G[h(x)] \int \frac{h_2^2(z) dz}{p(z)} < 0$$

In other words, the condition Eq. (A.7) is equivalent to requiring that there exists an $h(x)$ such that $G[h(x)] < 0$. Hence, a sufficient condition for IB_β -learnability is that there exists an $h(x)$ such that

$$G[h(x)] = \int dx h(x)^2 p(x) - \beta \int \frac{dy}{p(y)} \left(\int dx h(x)p(x)p(y|x) \right)^2 + (\beta - 1) \left(\int dx h(x)p(x) \right)^2 < 0 \quad (\text{A.8})$$

When $h(x) = C = \text{constant}$ in the entire input space \mathcal{X} , Eq. (A.8) becomes:

$$C^2 - \beta C^2 + (\beta - 1)C^2 < 0$$

which cannot be true. Therefore, $h(x) = \text{constant}$ cannot satisfy Eq. (A.8).

Rearranging terms and simplifying, we have

$$\beta \left[\int \frac{dy}{p(y)} \left(\int dx h(x)p(x)p(y|x) \right)^2 - \left(\int dx h(x)p(x) \right)^2 \right] > \int dx h(x)^2 p(x) - \left(\int dx h(x)p(x) \right)^2 \quad (\text{A.9})$$

Written in the form of expectations, we have

$$\beta \cdot \left(\mathbb{E}_{y \sim p(y)} \left[\left(\mathbb{E}_{x \sim p(x|y)} [h(x)] \right)^2 \right] - \left(\mathbb{E}_{x \sim p(x)} [h(x)] \right)^2 \right) > \mathbb{E}_{x \sim p(x)} [h(x)^2] - \left(\mathbb{E}_{x \sim p(x)} [h(x)] \right)^2 \quad (\text{A.10})$$

Since the square function is convex, using Jensen's inequality on the L.H.S. of Eq. A.10, we have

$$\mathbb{E}_{y \sim p(y)} \left[\left(\mathbb{E}_{x \sim p(x|y)} [h(x)] \right)^2 \right] \geq \left(\mathbb{E}_{y \sim p(y)} \left[\mathbb{E}_{x \sim p(x|y)} [h(x)] \right] \right)^2 = \left(\mathbb{E}_{x \sim p(x)} [h(x)] \right)^2$$

The equality holds iff $\mathbb{E}_{x \sim p(x|y)} [h(x)]$ is constant w.r.t. y , i.e. Y is independent of X . Therefore, in order for Eq. (A.10) to hold, we require that Y is not independent of X .

Using Jensen's inequality on the inner expectation on the L.H.S. of Eq. (A.10), we have

$$\mathbb{E}_{y \sim p(y)} \left[\left(\mathbb{E}_{x \sim p(x|y)} [h(x)] \right)^2 \right] \leq \mathbb{E}_{y \sim p(y)} [\mathbb{E}_{x \sim p(x|y)} [h(x)^2]] = \mathbb{E}_{x \sim p(x)} [h(x)^2] \quad (\text{A.11})$$

The equality holds when $h(x)$ is a constant. Since we require that $h(x)$ is not a constant, we have that the equality cannot be reached.

Similarly, using Jensen's inequality on the R.H.S. of Eq. A.10, we have that

$$\mathbb{E}_{x \sim p(x)}[h(x)^2] > (\mathbb{E}_{x \sim p(x)}[h(x)])^2$$

where we have used the requirement that $h(x)$ cannot be constant.

Under the constraint that Y is not independent of X , we can divide both sides of Eq. A.10, and obtain the condition: there exists an $h(x)$ such that

$$\beta > \frac{\mathbb{E}_{x \sim p(x)}[h(x)^2] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}{\mathbb{E}_{y \sim p(y)}[(\mathbb{E}_{x \sim p(x|y)}[h(x)])^2] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}$$

i.e.

$$\beta > \inf_{h(x)} \frac{\mathbb{E}_{x \sim p(x)}[h(x)^2] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}{\mathbb{E}_{y \sim p(y)}[(\mathbb{E}_{x \sim p(x|y)}[h(x)])^2] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}$$

which proves the condition of Theorem 4.

Furthermore, from Eq. (A.11) we have

$$\beta_0[h(x)] > 1$$

for $h(x) \not\equiv \text{const}$, which satisfies the necessary condition of $\beta > 1$ in Section 3.3.

Proof of lower bound of slope of the Pareto frontier at the origin:

Now we prove the second statement of Theorem 4. Since $\delta I(X; Z) = 0$ and $\delta I(Y; Z) = 0$ according to Lemma 2.1, we have $\left(\frac{\Delta I(Y; Z)}{\Delta I(X; Z)}\right)^{-1} = \left(\frac{\delta^2 I(Y; Z)}{\delta^2 I(X; Z)}\right)^{-1}$. Substituting into the expression of $\delta^2 I(Y; Z)$ and $\delta^2 I(X; Z)$ from Lemma 13.1, we have

$$\begin{aligned}
& \left(\frac{\Delta I(Y; Z)}{\Delta I(X; Z)} \right)^{-1} \\
&= \left(\frac{\delta^2 I(Y; Z)}{\delta^2 I(X; Z)} \right)^{-1} \\
&= \frac{\frac{\epsilon^2}{2} \int dx dz \frac{p(x)^2}{p(x)p(z)} h(z|x)^2 - \frac{\epsilon^2}{2} \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')}{\frac{\epsilon^2}{2} \int dx dx' dy dz \frac{p(x,y)p(x',y)}{p(y)p(z)} h(z|x)h(z|x') - \frac{\epsilon^2}{2} \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')} \\
&= \frac{\left(\int dx p(x)h(x)^2 - \int dx dx' p(x)p(x')h(x)h(z|x') \right) \int \frac{h_2(z)^2}{p(z)} dz}{\left(\int dx dx' dy \frac{p(x,y)p(x',y)}{p(y)} h(x)h(z|x') - \int dx dx' p(x)p(x')h(x)h(z|x') \right) \int \frac{h_2(z)^2}{p(z)} dz} \\
&= \frac{\int dx p(x)h(x)^2 - \int dx dx' p(x)p(x')h(x)h(z|x')}{\int dx dx' dy \frac{p(x,y)p(x',y)}{p(y)} h(x)h(z|x') - \int dx dx' p(x)p(x')h(x)h(z|x')} \\
&= \frac{\mathbb{E}_{x \sim p(x)}[h(x)^2] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}{\mathbb{E}_{y \sim p(y)} \left[(\mathbb{E}_{x \sim p(x|y)}[h(x)])^2 \right] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2} \\
&= \frac{\frac{\mathbb{E}_{x \sim p(x)}[h(x)^2]}{(\mathbb{E}_{x \sim p(x)}[h(x)])^2} - 1}{\mathbb{E}_{y \sim p(y)} \left[\left(\frac{\mathbb{E}_{x \sim p(x|y)}[h(x)]}{\mathbb{E}_{x \sim p(x)}[h(x)]} \right)^2 \right] - 1} \\
&= \beta_0[h(x)]
\end{aligned}$$

Therefore, $(\inf_{h(x)} \beta_0[h(x)])^{-1}$ gives the largest slope of $\Delta I(Y; Z)$ vs. $\Delta I(X; Z)$ for perturbation function of the form $h_1(z|x) = h(x)h_2(z)$ satisfying $\int h_2(z)dz = 0$ and $\int \frac{h_2(z)^2}{p(z)} dz > 0$, which is a lower bound of slope of $\Delta I(Y; Z)$ vs. $\Delta I(X; Z)$ for all possible perturbation function $h_1(z|x)$. The latter is the slope of the Pareto frontier of the $I(Y; Z)$ vs. $I(X; Z)$ curve at the origin.

Inflection point for general Z : If we *do not* assume that Z is at the origin of the information plane, but at some general stationary solution Z^* with $p(z|x)$, we define

$$\begin{aligned}
\beta^{(2)}[h(x)] &= \left(\frac{\delta^2 I(Y; Z)}{\delta^2 I(X; Z)} \right)^{-1} \\
&= \frac{\frac{\epsilon^2}{2} \int dx dz \frac{p(x)^2}{p(x, z)} h(z|x)^2 - \frac{\epsilon^2}{2} \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')}{\frac{\epsilon^2}{2} \int dx dx' dy dz \frac{p(x,y)p(x',y)}{p(y,z)} h(z|x)h(z|x') - \frac{\epsilon^2}{2} \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')} \\
&= \frac{\int dx dz \frac{p(x)^2}{p(x, z)} h(z|x)^2 - \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')}{\int dx dx' dy dz \frac{p(x,y)p(x',y)}{p(y,z)} h(z|x)h(z|x') - \int dx dx' dz \frac{p(x)p(x')}{p(z)} h(z|x)h(z|x')} \\
&= \frac{\int \frac{dz}{p(z)} \left[\int dx \frac{p(x)^2}{p(x|z)} h(z|x)^2 - (\int dx p(x)h(z|x))^2 \right]}{\int \frac{dz}{p(z)} \left[\int \frac{dy}{p(y|z)} (\int dx p(x,y)h(z|x))^2 - (\int dx p(x)h(z|x))^2 \right]} \\
&= \frac{\int \frac{dz}{p(z)} \left[\frac{\int dx \frac{p(x)^2}{p(x|z)} h(z|x)^2}{(\int dx p(x)h(z|x))^2} - 1 \right]}{\int \frac{dz}{p(z)} \left[\frac{\int \frac{dy}{p(y|z)} (\int dx p(x,y)h(z|x))^2}{(\int dx p(x)h(z|x))^2} - 1 \right]} \\
&= \frac{\int dz \left[\frac{\int dx \frac{p(x)^2}{p(z|x)} h(z|x)^2}{(\int dx p(x)h(z|x))^2} - \frac{1}{p(z)} \right]}{\int dz \left[\frac{\int \frac{dy}{p(z|y)p(y)} (\int dx p(x,y)h(z|x))^2}{(\int dx p(x)h(z|x))^2} - \frac{1}{p(z)} \right]} \\
&= \frac{\int dz \left[\int dx \frac{p(x)}{p(z|x)} h(z|x)^2 - \frac{1}{p(z)} (\int dx p(x)h(z|x))^2 \right]}{\int dz \left[\int \frac{dy}{p(z|y)p(y)} (\int dx p(x,y)h(z|x))^2 - \frac{1}{p(z)} (\int dx p(x)h(z|x))^2 \right]}
\end{aligned}$$

which reduces to $\beta_0[h(x)]$ when $p(z|x) = p(z)$. When

$$\beta > \inf_{h(z|x)} \beta^{(2)}[h(z|x)] \quad (\text{A.12})$$

it becomes a non-stable solution (non-minimum), and we will have other Z that achieves a better $\text{IB}_\beta(X, Y; Z)$ than the current Z^* .

■

A.2.8 What IB first learns at its onset of learning

In this section, we prove that at the onset of learning, if letting $h(z|x) = h^*(x)h_2(z)$, we have

$$p_\beta(y|x) = p(y) + \epsilon^2 C_z (h^*(x) - \bar{h}_x^*) \int p(x,y) (h^*(x) - \bar{h}_x^*) dx \quad (\text{A.13})$$

where $p_\beta(y|x)$ is the estimated $p(y|x)$ by IB for a certain β , $h^*(x) = \inf_{h(x)} \beta_0[h(x)]$, $\bar{h}_x^* = \int h^*(x)p(x)dx$, $C_z = \int \frac{h_x^2(z)}{p(z)} dz$ is a constant.

Proof. In IB, we use $p_\beta(z|x)$ to obtain Z from X , then obtain the prediction of Y from Z using $p_\beta(y|z)$. Here we use subscript β to denote the probability (density) at the optimum of $\text{IB}_\beta[p(z|x)]$ at a specific β . We have

$$\begin{aligned} p_\beta(y|x) &= \int p_\beta(y|z)p_\beta(z|x)dz \\ &= \int dz \frac{p_\beta(y,z)p_\beta(z|x)}{p_\beta(z)} \\ &= \int dz \frac{p_\beta(z|x)}{p_\beta(z)} \int p(x',y)p_\beta(z|x')dx' \end{aligned}$$

When we have a small perturbation $\epsilon \cdot h(z|x)$ at the trivial representation, $p_\beta(z|x) = p_{\beta_0}(z) + \epsilon \cdot h(z|x)$, we have $p_\beta(z) = p_{\beta_0}(z) + \epsilon \cdot \int h(z|x'')p(x'')dx''$. Substituting, we have

$$\begin{aligned} p_\beta(y|x) &= \int dz \frac{p_{\beta_0}(z) \left(1 + \epsilon \cdot \frac{h(z|x)}{p_{\beta_0}(z)}\right)}{p_{\beta_0}(z) \left(1 + \epsilon \cdot \frac{\int h(z|x'')p(x'')dx''}{p_{\beta_0}(z)}\right)} \int p(x',y)p_{\beta_0}(z) \left(1 + \epsilon \cdot \frac{h(z|x')}{p_{\beta_0}(z)}\right) dx' \\ &= \int dz \frac{1 + \epsilon \cdot \frac{h(z|x)}{p_{\beta_0}(z)}}{1 + \epsilon \cdot \frac{\int h(z|x'')p(x'')dx''}{p_{\beta_0}(z)}} \int p(x',y)p_{\beta_0}(z) \left(1 + \epsilon \cdot \frac{h(z|x')}{p_{\beta_0}(z)}\right) dx' \end{aligned}$$

The 0th-order term is $\int dz dx' p(x',y)p_{\beta_0}(z) = p(y)$. The first-order term is

$$\begin{aligned}
\delta p_\beta(z|x) &= \epsilon \cdot \int dz dx' \left(h(z|x) + h(z|x') - \int h(z|x'') p(x'') dx'' \right) p(x', y) \\
&= \epsilon \cdot \int dx' \left(\int dz h(z|x) + \int dz h(z|x') \right) - \epsilon \cdot \int dx' dx'' p(x', y) p(x'') \int dz h(z|x'') \\
&= 0 - 0 \\
&= 0
\end{aligned}$$

since we have $\int h(z|x) dz = 0$ for any x .

For the second-order term, using $h(z|x) = h^*(x)h_2(z)$ and $C_z = \int \frac{dz}{p_{\beta_0}(z)} h_2^2(z)$, it is

$$\begin{aligned}
\delta^2 p_\beta(y|x) &= \epsilon^2 \cdot \int dz \left(\frac{\int h(z|x'') p(x'') dx''}{p_{\beta_0}(z)} \right)^2 \int p(x', y) p_{\beta_0}(z) dx' \\
&\quad - \epsilon^2 \cdot \int dz \frac{h(z|x) \int h(z|x'') p(x'') dx''}{(p_{\beta_0}(z))^2} \int p(x', y) p_{\beta_0}(z) dx' \\
&\quad + \epsilon^2 \int dz \left(h(z|x) - \int h(z|x'') p(x'') dx'' \right) \int p(x', y) \frac{h(z|x')}{p_{\beta_0}(z)} dx' \\
&= \epsilon^2 C_z \cdot \left(\int h^*(x'') p(x'') dx'' \right)^2 p(y) \\
&\quad - \epsilon^2 C_z \cdot h^*(x) \int h^*(x'') p(x'') dx'' p(y) \\
&\quad + \epsilon^2 C_z \cdot h^*(x) \int p(x', y) h^*(x') dx' \\
&\quad - \epsilon^2 C_z \cdot \int h^*(x'') p(x'') dx'' \int p(x', y) h^*(x') dx' \\
&= \epsilon^2 C_z (h^*(x) - \bar{h}_x^*) \left[\left(\int p(x', y) h^*(x') dx' \right) - \bar{h}_x^* p(y) \right] \\
&= \epsilon^2 C_z (h^*(x) - \bar{h}_x^*) \int p(x', y) (h^*(x') - \bar{h}_x^*) dx'
\end{aligned}$$

where $\bar{h}_x^* = \int h^*(x) p(x) dx$. Combining everything, we have up to the second order,

$$p_\beta(y|x) = p(y) + \epsilon^2 C_z (h^*(x) - \bar{h}_x^*) \int p(x, y) (h^*(x) - \bar{h}_x^*) dx$$

■

A.2.9 Proof of Theorem 5

Proof. According to Theorem 4, a sufficient condition for (X, Y) to be IB_β -learnable is that X and Y are not independent, and

$$\beta > \inf_{h(x)} \frac{\frac{\mathbb{E}_{x \sim p(x)}[h(x)^2]}{(\mathbb{E}_{x \sim p(x)}[h(x)])^2} - 1}{\mathbb{E}_{y \sim p(y)} \left[\left(\frac{\mathbb{E}_{x \sim p(x|y)}[h(x)]}{\mathbb{E}_{x \sim p(x)}[h(x)]} \right)^2 \right] - 1} \quad (\text{A.14})$$

We can assume a specific form of $h(x)$, and obtain a (potentially stronger) sufficient condition. Specifically, we let

$$h(x) = \begin{cases} 1, & x \in \Omega_x \\ 0, & \text{otherwise} \end{cases} \quad (\text{A.15})$$

for certain $\Omega_x \subset \mathcal{X}$. Substituting into Eq. (A.15), we have that a sufficient condition for (X, Y) to be IB_β -learnable is

$$\beta > \inf_{\Omega_x \subset \mathcal{X}} \frac{\frac{p(\Omega_x)}{p(\Omega_x)^2} - 1}{\int dy p(y) \left(\frac{\int_{x \in \Omega_x} dx p(x|y) dx}{p(\Omega_x)} \right)^2 - 1} > 0 \quad (\text{A.16})$$

where $p(\Omega_x) = \int_{x \in \Omega_x} p(x) dx$.

The denominator of Eq. (A.16) is

$$\begin{aligned} & \int dy p(y) \left(\frac{\int_{x \in \Omega_x} dx p(x|y) dx}{p(\Omega_x)} \right)^2 - 1 \\ &= \int dy p(y) \left(\frac{p(\Omega_x|y)}{p(\Omega_x)} \right)^2 - 1 \\ &= \int dy \frac{p(y|\Omega_x)^2}{p(y)} - 1 \\ &= \mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right] \end{aligned}$$

Using the inequality $x - 1 \geq \log x$, we have

$$\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right] \geq \mathbb{E}_{y \sim p(y|\Omega_x)} \left[\log \frac{p(y|\Omega_x)}{p(y)} \right] \geq 0$$

Both equalities hold iff $p(y|\Omega_x) \equiv p(y)$, at which the denominator of Eq. (A.16) is equal to 0 and the expression inside the infimum diverge, which will not contribute to the infimum. Except this scenario, the denominator is greater than 0. Substituting into Eq. (A.16), we have that a sufficient condition for (X, Y) to be IB_β -learnable is

$$\beta > \inf_{\Omega_x \subset \mathcal{X}} \frac{\frac{p(\Omega_x)}{p(\Omega_x)^2} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]} \quad (\text{A.17})$$

Since Ω_x is a subset of \mathcal{X} , by the definition of $h(x)$ in Eq. (A.15), $h(x)$ is not a constant in the entire \mathcal{X} . Hence the numerator of Eq. (A.17) is positive. Since its denominator is also positive, we can then neglect the “ > 0 ”, and obtain the condition in Theorem 5.

Since the $h(x)$ used in this theorem is a subset of the $h(x)$ used in Theorem 4, the infimum for Eq. (3.5) is greater than or equal to the infimum in Eq. (3.2). Therefore, according to the second statement of Theorem 4, we have that the $(\inf_{\Omega_x \subset \mathcal{X}} \beta_0(\Omega_x))^{-1}$ is also a lower bound of the slope for the Pareto frontier of $I(Y; Z)$ vs. $I(X; Z)$ curve.

Now we prove that the condition Eq. (3.5) is invariant to invertible mappings of X . In fact, if $X' = g(X)$ is a uniquely invertible map (if X is continuous, g is additionally required to be continuous), let $\mathcal{X}' = \{g(x)|x \in \Omega_x\}$, and denote $g(\Omega_x) \equiv \{g(x)|x \in \Omega_x\}$ for any $\Omega_x \subset \mathcal{X}$, we have $p(g(\Omega_x)) = p(\Omega_x)$, and $p(y|g(\Omega_x)) = p(y|\Omega_x)$. Then for dataset (X, Y) , let $\Omega'_x = g(\Omega_x)$, we have

$$\frac{\frac{1}{p(\Omega'_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega'_x)} \left[\frac{p(y|\Omega'_x)}{p(y)} - 1 \right]} = \frac{\frac{1}{p(\Omega_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]} \quad (\text{A.18})$$

Additionally we have $\mathcal{X}' = g(\mathcal{X})$. Then

$$\inf_{\Omega'_x \subset \mathcal{X}'} \frac{\frac{1}{p(\Omega'_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega'_x)} \left[\frac{p(y|\Omega'_x)}{p(y)} - 1 \right]} = \inf_{\Omega_x \subset \mathcal{X}} \frac{\frac{1}{p(\Omega_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]} \quad (\text{A.19})$$

For dataset $(X', Y) = (g(X), Y)$, applying Theorem 5 we have that a sufficient condition for it to be IB_β -learnable is

$$\beta > \inf_{\Omega'_x \subset \mathcal{X}'} \frac{\frac{1}{p(\Omega'_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega'_x)} \left[\frac{p(y|\Omega'_x)}{p(y)} - 1 \right]} = \inf_{\Omega_x \subset \mathcal{X}} \frac{\frac{1}{p(\Omega_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]} \quad (\text{A.20})$$

where the equality is due to Eq. (A.19). Comparing with the condition for IB_β -learnability for (X, Y) (Eq. (3.5)), we see that they are the same. Therefore, the condition given by Theorem 5 is invariant to invertible mapping of X . ■

A.2.10 Proof of Corollary 5.1 and Corollary 5.2

Proof of Corollary 5.1

Proof. We use Theorem 5. Let Ω_x contain all elements x whose true class is y^* for some certain y^* , and 0 otherwise. Then we obtain a (potentially stronger) sufficient condition. Since the probability $p(y|y^*, x) = p(y|y^*)$ is class-conditional, we have

$$\begin{aligned} & \inf_{\Omega_x \subset \mathcal{X}} \frac{\frac{1}{p(\Omega_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]} \\ &= \inf_{y^*} \frac{\frac{1}{p(y^*)} - 1}{\mathbb{E}_{y \sim p(y|y^*)} \left[\frac{p(y|y^*)}{p(y)} - 1 \right]} \end{aligned}$$

By requiring $\beta > \inf_{y^*} \frac{\frac{1}{p(y^*)} - 1}{\mathbb{E}_{y \sim p(y|y^*)} \left[\frac{p(y|y^*)}{p(y)} - 1 \right]}$, we obtain a sufficient condition for IB_β learnability. ■

Proof of Corollary 5.2

Proof. We again use Theorem 5. Since Y is a deterministic function of X , let $Y = f(X)$. By the assumption that Y contains at least one value y such that its probability $p(y) > 0$, we let Ω_x contain only x such that $f(x) = y$. Substituting into Eq. (3.5), we have

$$\begin{aligned} & \frac{\frac{1}{p(\Omega_x)} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{p(y|\Omega_x)}{p(y)} - 1 \right]} \\ &= \frac{\frac{1}{p(y)} - 1}{\mathbb{E}_{y \sim p(y|\Omega_x)} \left[\frac{1}{p(y)} - 1 \right]} \\ &= \frac{\frac{1}{p(y)} - 1}{\frac{1}{p(y)} - 1} \\ &= 1 \end{aligned}$$

■

Therefore, the sufficient condition becomes $\beta > 1$.

A.2.11 β_0 hypercontractivity coefficient, contraction coefficient, $\beta_0[h(x)]$ and maximum correlation

In this section, we prove the relations between the IB-Learnability threshold β_0 , the hypercontractivity coefficient $\xi(X; Y)$, the contraction coefficient $\eta_{\text{KL}}(p(y|x), p(x))$, $\beta_0[h(x)]$ in Eq. (3.2), and maximum correlation $\rho_m(X, Y)$, as follows:

$$\frac{1}{\beta_0} = \xi(X; Y) = \eta_{\text{KL}}(p(y|x), p(x)) \geq \sup_{h(x)} \frac{1}{\beta_0[h(x)]} = \rho_m^2(X; Y) \quad (\text{A.21})$$

Proof. The hypercontractivity coefficient ξ is defined as [AGKN13]:

$$\xi(X; Y) \equiv \sup_{Z-X-Y} \frac{I(Y; Z)}{I(X; Z)}$$

By our definition of IB-learnability, (X, Y) is IB-Learnable iff there exists Z obeying the Markov chain $Z - X - Y$, such that

$$I(X; Z) - \beta \cdot I(Y; Z) < 0 = IB_\beta(X, Y; Z)|_{p(z|x)=p(z)}$$

Or equivalently there exists Z obeying the Markov chain $Z - X - Y$ such that

$$0 < \frac{1}{\beta} < \frac{I(Y; Z)}{I(X; Z)} \quad (\text{A.22})$$

By Theorem 2, the IB-Learnability region for β is $(\beta_0, +\infty)$, or equivalently the IB-Learnability region for $1/\beta$ is

$$0 < \frac{1}{\beta} < \frac{1}{\beta_0} \quad (\text{A.23})$$

Comparing Eq. (A.22) and Eq. (A.23), we have that

$$\frac{1}{\beta_0} = \sup_{Z-X-Y} \frac{I(Y; Z)}{I(X; Z)} = \xi(X; Y) \quad (\text{A.24})$$

In [AGKN13], the authors prove that

$$\xi(X; Y) = \eta_{\text{KL}}(p(y|x), p(x)) \quad (\text{A.25})$$

where the contraction coefficient $\eta_{\text{KL}}(p(y|x), p(x))$ is defined as

$$\eta_{\text{KL}}(p(y|x), p(x)) = \sup_{r(x) \neq p(x)} \frac{\mathbb{D}_{\text{KL}}(r(y)||p(y))}{\mathbb{D}_{\text{KL}}(r(x)||p(x))}$$

where $p(y) = \mathbb{E}_{x \sim p(x)}[p(y|x)]$ and $r(y) = \mathbb{E}_{x \sim r(x)}[p(y|x)]$. Treating $p(y|x)$ as a channel, the contraction coefficient measures how much the two distributions $r(x)$ and $p(x)$ becomes “nearer” (as measured by the KL-divergence) after passing through the channel.

In [AGKN13], the authors also provide a counterexample to an earlier result by [EC98] that incorrectly proved $\xi(X; Y) = \rho_m^2(X; Y)$. In the specific counterexample [AGKN13] design, $\xi(X; Y) > \rho_m^2(X; Y)$.

The maximum correlation is defined as $\rho_m(X; Y) \equiv \max_{f,g} \mathbb{E}[f(X)g(Y)]$ where $f(X)$ and $g(Y)$ are real-valued random variables such that $\mathbb{E}[f(X)] = \mathbb{E}[g(Y)] = 0$ and $\mathbb{E}[f^2(X)] = \mathbb{E}[g^2(Y)] = 1$ [Hir35, Geb41].

Now we prove $\xi(X; Y) \geq \rho_m^2(X; Y)$, based on Theorem 4. To see this, we use the alternate characterization of $\rho_m(X; Y)$ by [Rén59]:

$$\rho_m^2(X; Y) = \max_{f(X): \mathbb{E}[f(X)] = 0, \mathbb{E}[f^2(X)] = 1} \mathbb{E}[(\mathbb{E}[f(X)|Y])^2] \quad (\text{A.26})$$

Denoting $\bar{h} = \mathbb{E}_{p(x)}[h(x)]$, we can transform $\beta_0[h(x)]$ in Eq. (3.2) as follows:

$$\begin{aligned} \beta_0[h(x)] &= \frac{\mathbb{E}_{x \sim p(x)}[h(x)^2] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2}{\mathbb{E}_{y \sim p(y)} \left[(\mathbb{E}_{x \sim p(x|y)}[h(x)])^2 \right] - (\mathbb{E}_{x \sim p(x)}[h(x)])^2} \\ &= \frac{\mathbb{E}_{x \sim p(x)}[h(x)^2] - \bar{h}^2}{\mathbb{E}_{y \sim p(y)} \left[(\mathbb{E}_{x \sim p(x|y)}[h(x)])^2 \right] - \bar{h}^2} \\ &= \frac{\mathbb{E}_{x \sim p(x)}[(h(x) - \bar{h})^2]}{\mathbb{E}_{y \sim p(y)} \left[(\mathbb{E}_{x \sim p(x|y)}[h(x) - \bar{h}])^2 \right]} \\ &= \frac{1}{\mathbb{E}_{y \sim p(y)} \left[(\mathbb{E}_{x \sim p(x|y)}[f(x)])^2 \right]} \\ &= \frac{1}{\mathbb{E}[(\mathbb{E}[f(X)|Y])^2]} \end{aligned}$$

where we denote $f(x) = \frac{h(x) - \bar{h}}{(\mathbb{E}_{x \sim p(x)}[(h(x) - \bar{h})^2])^{1/2}}$, so that $\mathbb{E}[f(X)] = 0$ and $\mathbb{E}[f^2(X)] = 1$.

Combined with Eq. (A.26), we have

$$\sup_{h(x)} \frac{1}{\beta_0[h(x)]} = \rho_m^2(X; Y) \quad (\text{A.27})$$

Our Theorem 4 states that

$$\sup_{h(x)} \frac{1}{\beta_0[h(x)]} \leq \frac{1}{\beta_0} \quad (\text{A.28})$$

Combining Eqs. (A.23), (A.27) and Eq. (A.28), we have

$$\rho_m^2(X; Y) \leq \xi(X; Y) \quad (\text{A.29})$$

In summary, the relations among the quantities are:

$$\frac{1}{\beta_0} = \xi(X; Y) = \eta_{\text{KL}}(p(y|x), p(x)) \geq \sup_{h(x)} \frac{1}{\beta_0[h(x)]} = \rho_m^2(X; Y) \quad (\text{A.30})$$

■

A.2.12 Experiment Details

We use the Variational Information Bottleneck (VIB) objective from [AFDM16]. For the synthetic experiment, the latent Z has dimension of 2. The encoder is a neural net with 2 hidden layers, each of which has 128 neurons with ReLU activation. The last layer has linear activation and 4 output neurons; the first two parameterize the mean of a Gaussian and the last two parameterize the log variance. The decoder is a neural net with 1 hidden layer with 128 neurons and ReLU activation. Its last layer has linear activation and outputs the logit for the class labels. It uses a mixture of Gaussian prior with 500 components (for the experiment with class overlap, 256 components), each of which is a 2D Gaussian with learnable mean and log variance, and the weights for the components are also learnable. For the MNIST experiment, the architecture is mostly the same, except the following: (1) for Z , we let it have dimension of 256. For the prior, we use standard Gaussian with diagonal covariance matrix.

For all experiments, we use Adam ([KB14]) optimizer with default parameters. We do not add any explicit regularization. We use learning rate of 10^{-4} and have a learning rate decay of $\frac{1}{1+0.01\times\text{epoch}}$. We train in total 2000 epochs with mini-batch size of 500.

For estimation of the observed β_0 in Fig. 3-3, in the $I(X; Z)$ vs. β_i curve (β_i denotes the i^{th} β), we take the mean and standard deviation of $I(X; Z)$ for the lowest 5 β_i values, denoting as μ_β, σ_β ($I(Y; Z)$ has similar behavior, but since we are minimizing $I(X; Z) - \beta \cdot I(Y; Z)$, the onset of nonzero $I(X; Z)$ is less prone to noise). When $I(X; Z)$ is greater than $\mu_\beta +$

$3\sigma_\beta$, we regard it as learning a non-trivial representation, and take the average of β_i and β_{i-1} as the experimentally estimated onset of learning. We also inspect manually and confirm that it is consistent with human intuition.

For estimating β_0 using Alg. 1, at step 6 we use the following discrete search algorithm. We fix $i_{\text{left}} = 1$ and gradually narrow down the range $[a, b]$ of i_{right} , starting from $[1, N]$. At each iteration, we set a tentative new range $[a', b']$, where $a' = 0.8a + 0.2b$, $b' = 0.2a + 0.8b$, and calculate $\tilde{\beta}_{0,a'} = \mathbf{Get}\beta(P_{y|x}, p_y, \Omega_{a'})$, $\tilde{\beta}_{0,b'} = \mathbf{Get}\beta(P_{y|x}, p_y, \Omega_{b'})$ where $\Omega_{a'} = \{1, 2, \dots, a'\}$ and $\Omega_{b'} = \{1, 2, \dots, b'\}$. If $\tilde{\beta}_{0,a'} < \tilde{\beta}_{0,a}$, let $a \leftarrow a'$. If $\tilde{\beta}_{0,b'} < \tilde{\beta}_{0,b}$, let $b \leftarrow b'$. In other words, we narrow down the range of i_{right} if we find that the Ω given by the left or right boundary gives a lower $\tilde{\beta}_0$ value. The process stops when both $\tilde{\beta}_{0,a'}$ and $\tilde{\beta}_{0,b'}$ stop improving (which we find always happens when $b' = a' + 1$), and we return the smaller of the final $\tilde{\beta}_{0,a'}$ and $\tilde{\beta}_{0,b'}$ as $\tilde{\beta}_0$.

For estimation of $p(y|x)$ for (2') Alg. 1 and (3') $\hat{\eta}_{\text{KL}}$ for both synthetic and MNIST experiments, we use a 3-layer neuron net where each hidden layer has 128 neurons and ReLU activation. The last layer has linear activation. The objective is cross-entropy loss. We use Adam [KB14] optimizer with a learning rate of 10^{-4} , and train for 100 epochs (after which the validation loss does not go down).

For estimating β_0 via (3') $\hat{\eta}_{\text{KL}}$ by the algorithm in [KGK⁺17], we use the code from the GitHub repository provided by the paper², using the same $p(y|x)$ employed for (2') Alg. 1. Since our datasets are classification tasks, we use $A_{ij} = p(y_j|x_i)/p(y_j)$ instead of the kernel density for estimating matrix A ; we take the maximum of 10 runs as estimation of μ .

CIFAR10 Details

We trained a deterministic 28x10 wide resnet [HZRS16b, ZK16], using the open source implementation from [CZM⁺18]. However, we extended the final 10 dimensional logits of that model through another 3 layer MLP classifier, in order to keep the inference network architecture identical between this model and the VIB models we describe below. During training, we dynamically added label noise according to the class confusion matrix in Tab.

²At <https://github.com/wgao9/hypercontractivity>.

Table A.4: Class confusion matrix used in CIFAR10 experiments. The value in row i , column j means for class i , the probability of labeling it as class j . The mean confusion across the classes is 20%.

	Plane	Auto.	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck
Plane	0.82232	0.00238	0.021	0.00069	0.00108	0	0.00017	0.00019	0.1473	0.00489
Auto.	0.00233	0.83419	0.00009	0.00011	0	0.00001	0.00002	0	0.00946	0.15379
Bird	0.03139	0.00026	0.76082	0.0095	0.07764	0.01389	0.1031	0.00309	0.00031	0
Cat	0.00096	0.0001	0.00273	0.69325	0.00557	0.28067	0.01471	0.00191	0.00002	0.0001
Deer	0.00199	0	0.03866	0.00542	0.83435	0.01273	0.02567	0.08066	0.00052	0.00001
Dog	0	0.00004	0.00391	0.2498	0.00531	0.73191	0.00477	0.00423	0.00001	0
Frog	0.00067	0.00008	0.06303	0.05025	0.0337	0.00842	0.8433	0	0.00054	0
Horse	0.00157	0.00006	0.00649	0.00295	0.13058	0.02287	0	0.83328	0.00023	0.00196
Ship	0.1288	0.01668	0.00029	0.00002	0.00164	0.00006	0.00027	0.00017	0.83385	0.01822
Truck	0.01007	0.15107	0	0.00015	0.00001	0.00001	0	0.00048	0.02549	0.81273

A.4. The mean label noise averaged across the 10 classes is 20%. After that model had converged, we used it to estimate β_0 with Alg. 1. Even with 20% label noise, β_0 was estimated to be 1.0483.

We then trained 73 different VIB models using the same 28x10 wide resnet architecture for the encoder, parameterizing the mean of a 10-dimensional unit variance Gaussian. Samples from the encoder distribution were fed to the same 3 layer MLP classifier architecture used in the deterministic model. The marginal distributions were mixtures of 500 fully covariate 10-dimensional Gaussians, all parameters of which are trained. The VIB models had β ranging from 1.02 to 2.0 by steps of 0.02, plus an extra set ranging from 1.04 to 1.06 by steps of 0.001 to ensure we captured the empirical β_0 with high precision.

However, this particular VIB architecture does not start learning until $\beta > 2.5$, so none of these models would train as described.³ Instead, we started them all at $\beta = 100$, and annealed β down to the corresponding target over 10,000 training gradient steps. The models continued to train for another 200,000 gradient steps after that. In all cases, the models converged to essentially their final accuracy within 20,000 additional gradient steps after annealing was completed. They were stable over the remaining $\sim 180,000$ gradient steps.

³A given architecture trained using maximum likelihood and with no stochastic layers will tend to have higher effective capacity than the same architecture with a stochastic layer that has a fixed but non-trivial variance, even though those two architectures have exactly the same number of learnable parameters.

A.3 Appendix for Chapter 4

A.3.1 Calculus of variations at any order of $\text{IB}_\beta[p(z|x)]$

Here we prove the Lemma 13.2, which will be crucial in the lemmas and theorems in this paper that follows.

Lemma 13.2. *For a relative perturbation function $r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ for a $p(z|x)$, where $r(z|x)$ satisfies $\mathbb{E}_{z \sim p(z|x)}[r(z|x)] = 0$, we have that the IB objective can be expanded as*

$$\begin{aligned}
& \text{IB}_\beta[p(z|x)(1 + \epsilon \cdot r(z|x))] \\
= & \text{IB}_\beta[p(z|x)] + \epsilon \cdot \left(\mathbb{E}_{x,z \sim p(x,z)} \left[r(z|x) \log \frac{p(z|x)}{p(z)} \right] - \beta \cdot \mathbb{E}_{y,z \sim p(y,z)} \left[r(z|y) \log \frac{p(z|y)}{p(z)} \right] \right) \\
& + \sum_{n=2}^{\infty} \frac{(-1)^n \epsilon^n}{n(n-1)} \{ (\mathbb{E}[r^n(z|x)] - \mathbb{E}[r^n(z)]) - \beta \cdot (\mathbb{E}[r^n(z|y)] - \mathbb{E}[r^n(z)]) \} \\
= & \text{IB}_\beta[p(z|x)] + \epsilon \cdot \left(\mathbb{E}_{x,z \sim p(x,z)} \left[r(z|x) \log \frac{p(z|x)}{p(z)} \right] - \beta \cdot \mathbb{E}_{y,z \sim p(y,z)} \left[r(z|y) \log \frac{p(z|y)}{p(z)} \right] \right) \\
& + \frac{\epsilon^2}{1 \cdot 2} \{ (\mathbb{E}[r^2(z|x)] - \mathbb{E}[r^2(z)]) - \beta \cdot (\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)]) \} \\
& - \frac{\epsilon^3}{2 \cdot 3} \{ (\mathbb{E}[r^3(z|x)] - \mathbb{E}[r^3(z)]) - \beta \cdot (\mathbb{E}[r^3(z|y)] - \mathbb{E}[r^3(z)]) \} \\
& + \frac{\epsilon^4}{3 \cdot 4} \{ (\mathbb{E}[r^4(z|x)] - \mathbb{E}[r^4(z)]) - \beta \cdot (\mathbb{E}[r^4(z|y)] - \mathbb{E}[r^4(z)]) \} \\
& - \dots
\end{aligned} \tag{A.31}$$

where $r(z|y) = \mathbb{E}_{x \sim p(x|y,z)}[r(z|x)]$ and $r(z) = \mathbb{E}_{x \sim p(x|z)}[r(z|x)]$. The expectations in the equations are all w.r.t. all variables. For example $\mathbb{E}[r^2(z|x)] = \mathbb{E}_{x,z \sim p(x,z)}[r^2(z|x)]$.

Proof. Suppose that we perform a relative perturbation $r(z|x)$ on $p(z|x)$ such that the perturbed conditional probability is $p'(z|x) = p(z|x)(1 + \epsilon \cdot r(z|x))$, then we have

$$p'(z) = \int p(x)p'(z|x)dx = \int dx p(x)p(z|x)(1 + \epsilon \cdot r(z|x)) = p(z) + \epsilon \cdot \int dx p(x)p(z|x)r(z|x)$$

Therefore, we can denote the corresponding relative perturbation $r(z)$ on $p(z)$ as

$$r(z) \equiv \frac{1}{\epsilon} \frac{p'(z) - p(z)}{p(z)} = \frac{1}{p(z)} \int dx p(x) p(z|x) r(z|x) = \mathbb{E}_{x \sim p(x|z)} [r(z|x)]$$

Similarly, we have

$$p'(z|y) = \frac{p'(y, z)}{p(y)} = \frac{1}{p(y)} \int dx p(x, y) p(z|x) (1 + \epsilon \cdot r(z|x)) = p(z|y) + \epsilon \cdot \frac{1}{p(y)} \int dx p(x, y) p(z|x) r(z|x)$$

And we can denote the corresponding relative perturbation $r(z|y)$ on $p(z|y)$ as

$$r(z|y) \equiv \frac{1}{\epsilon} \frac{p'(z|y) - p(z|y)}{p(z|y)} = \frac{1}{p(z|y)p(y)} \int dx p(x, y) p(z|x) r(z|x) = \mathbb{E}_{x \sim p(x|y, z)} [r(z|x)]$$

Since

$$\text{IB}_\beta[p(z|x)] = I(X; Z) - \beta \cdot I(Y; Z) = \int dx dz p(x, z) \log \frac{p(z|x)}{p(z)} - \beta \cdot \int dy dz p(y, z) \log \frac{p(z|y)}{p(z)}$$

We have

$$\begin{aligned} \text{IB}_\beta[p'(z|x)] &= \text{IB}_\beta[p(z|x)(1 + \epsilon \cdot r(z|x))] \\ &= \int dx dz p(x) p'(z|x) \log \frac{p'(z|x)}{p'(z)} - \beta \cdot \int dy dz p(y) p'(z|y) \log \frac{p'(z|y)}{p'(z)} \\ &= \int dx dz p(x) p(z|x) (1 + \epsilon \cdot r(z|x)) \log \frac{p(z|x)(1 + \epsilon \cdot r(z|x))}{p(z)(1 + \epsilon \cdot r(z))} \\ &\quad - \beta \cdot \int dy dz p(y) p(z|y) (1 + \epsilon \cdot r(z|y)) \log \frac{p(z|y)(1 + \epsilon \cdot r(z|y))}{p(z)(1 + \epsilon \cdot r(z))} \\ &= \int dx dz p(x) p(z|x) (1 + \epsilon \cdot r(z|x)) \left[\log \frac{p(z|x)}{p(z)} + \log (1 + \epsilon \cdot r(z|x)) - \log (1 + \epsilon \cdot r(z)) \right] \\ &\quad - \beta \cdot \int dy dz p(y) p(z|y) (1 + \epsilon \cdot r(z|y)) \left[\log \frac{p(z|y)}{p(z)} + \log (1 + \epsilon \cdot r(z|y)) - \log (1 + \epsilon \cdot r(z)) \right] \\ &= \int dx dz p(x) p(z|x) (1 + \epsilon \cdot r(z|x)) \left[\log \frac{p(z|x)}{p(z)} + \sum_{n=1}^{\infty} (-1)^{n-1} \frac{\epsilon^n}{n} (r(z|x) - r(z)) \right] \\ &\quad - \beta \cdot \int dy dz p(y) p(z|y) (1 + \epsilon \cdot r(z|y)) \left[\log \frac{p(z|y)}{p(z)} + \sum_{n=1}^{\infty} (-1)^{n-1} \frac{\epsilon^n}{n} (r(z|y) - r(z)) \right] \end{aligned}$$

The 0th-order term is simply $\text{IB}_\beta[p(z|x)]$. The first order term is

$$\delta \text{IB}_\beta[p(z|x)] = \epsilon \cdot \left(\mathbb{E}_{x,z \sim p(x,z)} \left[r(z|x) \log \frac{p(z|x)}{p(z)} \right] - \beta \cdot \mathbb{E}_{y,z \sim p(y,z)} \left[r(z|y) \log \frac{p(z|y)}{p(z)} \right] \right)$$

The n^{th} -order term for $n \geq 2$ is

$$\begin{aligned} & \delta^n \text{IB}_\beta[p(z|x)] \\ &= (-1)^n \epsilon^n \int dx dz p(x)p(z|x) \left(-\frac{1}{n} [r^n(z|x) - r^n(z)] + r(z|x) \frac{1}{n-1} [r^{n-1}(z|x) - r^n(z)] \right) \\ & \quad - \beta \cdot (-1)^n \epsilon^n \int dy dz p(y)p(z|y) \left(-\frac{1}{n} [r^n(z|y) - r^n(z)] + r(z|y) \frac{1}{n-1} [r^{n-1}(z|y) - r^n(z)] \right) \\ &= \frac{(-1)^n \epsilon^n}{n(n-1)} (\mathbb{E}_{x,z \sim p(x,z)} [r^n(z|x)] - n \mathbb{E}_{x,z \sim p(x,z)} [r(z|x)r^{n-1}(z)] - (n-1) \mathbb{E}_{z \sim p(z)} [r^n(z)]) \\ & \quad - \beta \cdot \frac{(-1)^n \epsilon^n}{n(n-1)} (\mathbb{E}_{y,z \sim p(y,z)} [r^n(z|y)] - n \mathbb{E}_{y,z \sim p(y,z)} [r(z|y)r^{n-1}(z)] - (n-1) \mathbb{E}_{z \sim p(z)} [r^n(z)]) \\ &= \frac{(-1)^n \epsilon^n}{n(n-1)} \{(\mathbb{E}[r^n(z|x)] - \mathbb{E}[r^n(z)]) - \beta \cdot (\mathbb{E}[r^n(z|y)] - \mathbb{E}[r^n(z)])\} \end{aligned}$$

In the last equality we have used

$$\mathbb{E}_{x,z \sim p(x,z)} [r(z|x)r^{n-1}(z)] = \mathbb{E}_{z \sim p(z)} [r^{n-1}(z) \mathbb{E}_{x \sim p(x|z)} [r(z|x)]] = \mathbb{E}_{z \sim p(z)} [r^{n-1}(z)r(z)] = \mathbb{E}_{z \sim p(z)} [r^n(z)]$$

Combining the terms with all orders, we have

$$\begin{aligned} & \text{IB}_\beta[p(z|x)(1 + \epsilon \cdot r(z|x))] \\ &= \text{IB}_\beta[p(z|x)] + \epsilon \cdot \left(\mathbb{E}_{x,z \sim p(x,z)} \left[r(z|x) \log \frac{p(z|x)}{p(z)} \right] - \beta \cdot \mathbb{E}_{y,z \sim p(y,z)} \left[r(z|y) \log \frac{p(z|y)}{p(z)} \right] \right) \\ & \quad + \sum_{n=2}^{\infty} \frac{(-1)^n \epsilon^n}{n(n-1)} \{(\mathbb{E}[r^n(z|x)] - \mathbb{E}[r^n(z)]) - \beta \cdot (\mathbb{E}[r^n(z|y)] - \mathbb{E}[r^n(z)])\} \end{aligned}$$

■

As a side note, the KL-divergence between $p'(z|x) = p(z|x)(1 + \epsilon \cdot r(z|x))$ and $p(z|x)$ is

$$\begin{aligned}
\text{KL}(p'(z|x) \parallel p(z|x)) &= \int dz p(z|x)(1 + \epsilon \cdot r(z|x)) \log \frac{p(z|x)(1 + \epsilon \cdot r(z|x))}{p(z|x)} \\
&= \int dz p(z|x)(1 + \epsilon \cdot r(z|x)) \left(\epsilon \cdot r(z|x) - \frac{\epsilon^2}{2} \cdot r^2(z|x) + O(\epsilon^3) \right) \\
&= \epsilon \cdot \int dz p(z|x)r(z|x) + \frac{\epsilon^2}{2} \int dz p(z|x)r^2(z|x) + O(\epsilon^3) \\
&= \frac{\epsilon^2}{2} \mathbb{E}_{z \sim p(z|x)} [r^2(z|x)] + O(\epsilon^3)
\end{aligned}$$

Therefore, to the second order, we have

$$\mathbb{E}_{x \sim p(x)} [\text{KL}(p'(z|x) \parallel p(z|x))] = \frac{\epsilon^2}{2} \mathbb{E}[r^2(z|x)] \quad (\text{A.32})$$

Similarly, we have $\mathbb{E}_{x \sim p(x)} [\text{KL}(p(z|x) \parallel p'(z|x))] = \frac{\epsilon^2}{2} \mathbb{E}[r^2(z|x)]$ up to the second order.

Using similar procedure, we have up to the second-order,

$$\begin{aligned}
\mathbb{E}_{y \sim p(y)} [\text{KL}(p'(z|y) \parallel p(z|y))] &= \mathbb{E}_{y \sim p(y)} [\text{KL}(p(z|y) \parallel p'(z|y))] = \frac{\epsilon^2}{2} \mathbb{E}[r^2(z|y)] \\
\text{KL}(p'(z) \parallel p(z)) &= \text{KL}(p(z) \parallel p'(z)) = \frac{\epsilon^2}{2} \mathbb{E}[r^2(z)]
\end{aligned}$$

A.3.2 Proof of Lemma 5.1

Proof. From Lemma 13.2, we have

$$\delta^2 \text{IB}_\beta[p(z|x)] = \frac{\epsilon^2}{2} \{ (\mathbb{E}[r^2(z|x)] - \mathbb{E}[r^2(z)]) - \beta \cdot (\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)]) \} \quad (\text{A.33})$$

The condition of

$$\forall r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}, \delta^2 \text{IB}_\beta[p(z|x)] \geq 0 \quad (\text{A.34})$$

is equivalent to

$$\forall r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}, \beta \cdot (\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)]) \leq \mathbb{E}[r^2(z|x)] - \mathbb{E}[r^2(z)] \quad (\text{A.35})$$

Using Jensen's inequality and the convexity of the square function, we have

$$\begin{aligned}
\mathbb{E}[r^2(z|y)] &= \mathbb{E}_{y,z \sim p(y,z)} \left[(\mathbb{E}_{x \sim p(x|y,z)} [r(z|x)])^2 \right] \\
&= \mathbb{E}_{z \sim p(z)} \left[\mathbb{E}_{y \sim p(y|z)} \left[(\mathbb{E}_{x \sim p(x|y,z)} [r(z|x)])^2 \right] \right] \\
&\geq \mathbb{E}_{z \sim p(z)} \left[(\mathbb{E}_{y \sim p(y|z)} [\mathbb{E}_{x \sim p(x|y,z)} [r(z|x)]])^2 \right] \\
&= \mathbb{E}_{z \sim p(z)} \left[(\mathbb{E}_{x \sim p(x|z)} [r(z|x)])^2 \right] \\
&= \mathbb{E}[r^2(z)]
\end{aligned}$$

The equality holds iff $r(z|y) = \mathbb{E}_{x \sim p(x|y,z)} [r(z|x)]$ is constant w.r.t. y , for any z .

Using Jensen's inequality on $\mathbb{E}[r^2(z)]$, we have $\mathbb{E}[r^2(z)] = \mathbb{E}_{z \sim p(z)} \left[(\mathbb{E}_{x \sim p(x|z)} [r(z|x)])^2 \right] \leq \mathbb{E}_{z \sim p(z)} [\mathbb{E}_{x \sim p(x|z)} [r^2(z|x)]] = \mathbb{E}[r^2(z|x)]$, where the equality holds iff $r(z|x)$ is constant w.r.t. x for any z .

When $\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)] > 0$, we have that the condition Eq. (A.35) is equivalent to $\forall r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$, $\beta \leq \frac{\mathbb{E}[r^2(z|x)] - \mathbb{E}[r^2(z)]}{\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)]}$, i.e.

$$\beta \leq G[p(z|x)] \equiv \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} \frac{\mathbb{E}[r^2(z|x)] - \mathbb{E}[r^2(z)]}{\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)]} \quad (\text{A.36})$$

where $r(z|y) = \mathbb{E}_{x \sim p(x|y,z)} [r(z|x)]$ and $r(z) = \mathbb{E}_{x \sim p(x|z)} [r(z|x)]$.

If $\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)] = 0$, substituting into Eq. (A.35), we have

$$\beta \cdot 0 \leq \mathbb{E}[r^2(z|x)] - \mathbb{E}[r^2(z)] \quad (\text{A.37})$$

which is always true due to that $\mathbb{E}[r^2(z|x)] \geq \mathbb{E}[r^2(z)]$, and will be a looser condition than Eq. (A.36) above. Above all, we have Eq. (A.36).

■

Empirical estimate of $G[p(z|x)]$ To empirically estimate $G[p(z|x)]$ from a minibatch of $\{(x_i, y_i)\}, i = 1, 2, \dots, N$ and the encoder $p(z|x)$, we can make the following Monte Carlo

importance sampling estimation, where we use the samples $\{x_j\} \sim p(x)$ and also get samples of $\{z_i\} \sim p(z) = p(x)p(z|x)$, and have:

$$\begin{aligned}
\mathbb{E}_{x,z \sim p(x,z)}[r^2(z|x)] &= \int dx dz p(x)p(z) \frac{p(x,z)}{p(x)p(z)} r^2(z|x) \\
&\simeq \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N \frac{p(x_j, z_i)}{p(x_j)p(z_i)} r^2(z_i|x_j) \\
\mathbb{E}_{z \sim p(z)}[r^2(z)] &= \mathbb{E}_{z \sim p(z)} \left[(\mathbb{E}_{x \sim p(x|z)}[r(z|x)])^2 \right] \\
&\simeq \frac{1}{N} \sum_{i=1}^N \left(\int dx p(x|z_i) r(z_i|x) \right)^2 \\
&= \frac{1}{N} \sum_{i=1}^N \left(\int dx p(x) \frac{p(z_i|x)}{p(z_i)} r(z_i|x) \right)^2 \\
&\simeq \frac{1}{N} \sum_{i=1}^N \left(\frac{1}{N} \sum_{j=1}^N \frac{p(z_i|x_j)}{p(z_i)} r(z_i|x_j) \right)^2 \\
&\simeq \frac{1}{N} \sum_{i=1}^N \left(\frac{1}{N} \sum_{j=1}^N \frac{p(z_i|x_j)}{\frac{1}{N} \sum_{k=1}^N p(z_i|x_k)} r(z_i|x_j) \right)^2 \\
&= \frac{1}{N} \sum_{i=1}^N \left(\frac{\sum_{j=1}^N p(z_i|x_j) r(z_i|x_j)}{\sum_{j=1}^N p(z_i|x_j)} \right)^2
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{y,z \sim p(y,z)}[r^2(z|y)] &= \mathbb{E}_{y,z \sim p(y,z)} \left[(\mathbb{E}_{x \sim p(x|y,z)}[r(z|x)])^2 \right] \\
&\simeq \frac{1}{N} \sum_{i=1}^N \left(\int dx p(x|y_i, z_i) r(z_i|x) \right)^2 \\
&= \frac{1}{N} \sum_{i=1}^N \left(\frac{1}{p(y_i, z_i)} \int dx p(y_i) p(x|y_i) p(z_i|x) r(z_i|x) \right)^2 \\
&= \frac{1}{N} \sum_{i=1}^N \left(\frac{\int dx p(y_i) p(x|y_i) p(z_i|x) r(z_i|x)}{\int dx p(y_i) p(x|y_i) p(z_i|x)} \right)^2 \\
&\simeq \frac{1}{N} \sum_{i=1}^N \left(\frac{\sum_{x_j \in \Omega_x(y_i)} p(z_i|x_j) r(z_i|x_j)}{\sum_{x_j \in \Omega_x(y_i)} p(z_i|x_j)} \right)^2 \\
&= \frac{1}{N} \sum_{i=1}^N \left(\frac{\sum_{j=1}^N p(z_i|x_j) r(z_i|x_j) \mathbb{1}[y_i = y_j]}{\sum_{j=1}^N p(z_i|x_j) \mathbb{1}[y_i = y_j]} \right)^2
\end{aligned}$$

Here $\Omega_x(y_i)$ denotes the set of x examples that has label of y_i , and $\mathbb{1}[\cdot]$ is an indicator

function that takes value 1 if its argument is true, 0 otherwise.

The requirement of $\mathbb{E}_{z \sim p(z|x)}[r(z|x)] = 0$ yields

$$0 = \mathbb{E}_{z \sim p(z|x)}[r(z|x)] = \int dz p(z) \frac{p(z|x)}{p(z)} r(z|x) \simeq \frac{1}{N} \sum_{i=1}^N \frac{p(z_i|x_j)}{p(z_i)} r(z_i|x_j) \quad (\text{A.38})$$

for any x_j .

Combining all terms, we have that the empirical $\hat{G}[p(z|x)]$ is given by

$$\hat{G}[p(z|x)] = \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} \frac{\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^N \frac{p(x_j, z_i)}{p(x_j)p(z_i)} r^2(z_i|x_j) - \sum_{i=1}^N \left(\frac{\sum_{j=1}^N p(z_i|x_j)r(z_i|x_j)}{\sum_{j=1}^N p(z_i|x_j)} \right)^2}{\sum_{i=1}^N \left(\frac{\sum_{j=1}^N p(z_i|x_j)r(z_i|x_j)\mathbb{1}[y_i=y_j]}{\sum_{j=1}^N p(z_i|x_j)\mathbb{1}[y_i=y_j]} \right)^2 - \sum_{i=1}^N \left(\frac{\sum_{j=1}^N p(z_i|x_j)r(z_i|x_j)}{\sum_{j=1}^N p(z_i|x_j)} \right)^2} \quad (\text{A.39})$$

where $\{z_i\} \sim p(z)$ and $\{x_i\} \sim p(x)$. It is also possible to use different distributions for importance sampling, which will result in different formulas for empirical estimation of $G[p(z|x)]$.

A.3.3 $G_\Theta[p_\theta(z|x)]$ for parameterized distribution $p_\theta(z|x)$

Proof. For the parameterized⁴ $p_\theta(z|x)$ with $\theta \in \Theta$, after $\theta' \leftarrow \theta + \Delta\theta$, where⁵ $\Delta\theta \in \Theta$ is an infinitesimal perturbation on θ , we have that the distribution changes from $p_\theta(z|x)$ to

⁴In this paper, $\theta = (\theta_1, \theta_2, \dots, \theta_k)^T$ and $\frac{\partial p_\theta(z|x)}{\partial \theta} = \left(\frac{\partial p_\theta(z|x)}{\partial \theta_1}, \frac{\partial p_\theta(z|x)}{\partial \theta_2}, \dots, \frac{\partial p_\theta(z|x)}{\partial \theta_k} \right)^T$ are all column vectors. $\frac{\partial^2 p_\theta(z|x)}{\partial \theta^2}$ is a $k \times k$ matrix with (i, j) element of $\frac{\partial^2 p_\theta(z|x)}{\partial \theta_i \partial \theta_j}$.

⁵Note that since Θ is a field, it is closed under subtraction, we have $\Delta\theta \in \Theta$.

$p_{\theta+\Delta\theta}(z|x)$, and thus the relative perturbation on $p_{\theta}(z|x)$ is

$$\begin{aligned}\epsilon \cdot r(z|x) &= \frac{p_{\theta+\Delta\theta}(z|x) - p_{\theta}(z|x)}{p_{\theta}(z|x)} \\ &= \frac{1}{p_{\theta}(z|x)} \left(p_{\theta}(z|x) + \Delta\theta^T \frac{\partial p_{\theta}(z|x)}{\partial\theta} + \frac{1}{2} \Delta\theta^T \frac{\partial^2 p_{\theta}(z|x)}{\partial\theta^2} \Delta\theta + O(\|\Delta\theta\|^3) - p_{\theta}(z|x) \right) \\ &\simeq \Delta\theta^T \frac{\partial}{\partial\theta} \log p_{\theta}(z|x) + \frac{1}{2} \Delta\theta^T \frac{1}{p_{\theta}(z|x)} \frac{\partial^2 p_{\theta}(z|x)}{\partial\theta^2} \Delta\theta + O(\|\Delta\theta\|^3)\end{aligned}$$

where $\|\Delta\theta\|$ is the norm of $\Delta\theta$ in the parameter field Θ .

Similarly, we have

$$\begin{aligned}\epsilon \cdot r(z|y) &= \Delta\theta^T \frac{\partial}{\partial\theta} \log p_{\theta}(z|y) + \frac{1}{2} \Delta\theta^T \frac{1}{p_{\theta}(z|y)} \frac{\partial^2 p_{\theta}(z|y)}{\partial\theta^2} \Delta\theta + O(\|\Delta\theta\|^3) \\ \epsilon \cdot r(z) &= \Delta\theta^T \frac{\partial}{\partial\theta} \log p_{\theta}(z) + \frac{1}{2} \Delta\theta^T \frac{1}{p_{\theta}(z)} \frac{\partial^2 p_{\theta}(z)}{\partial\theta^2} \Delta\theta + O(\|\Delta\theta\|^3)\end{aligned}$$

Substituting the above expressions into the expansion of $\text{IB}_{\beta}[p(z|x)]$ in Eq. (A.31), and preserving to the second order $\|\Delta\theta\|^2$, we have

$$\begin{aligned}
& \text{IB}_\beta[p_\theta(z|x)(1 + \epsilon \cdot r(z|x))] \\
&= \text{IB}_\beta[p_\theta(z|x)] + \epsilon \cdot \left(\mathbb{E}_{x,z \sim p_\theta(x,z)} \left[r(z|x) \log \frac{p_\theta(z|x)}{p_\theta(z)} \right] - \beta \cdot \mathbb{E}_{y,z \sim p_\theta(y,z)} \left[r(z|y) \log \frac{p_\theta(z|y)}{p_\theta(z)} \right] \right) \\
&\quad + \frac{\epsilon^2}{1+2} \left\{ (\mathbb{E}_{x,z \sim p_\theta(x,z)} [r^2(z|x)] - \mathbb{E}_{z \sim p_\theta(z)} [r^2(z)]) - \beta \cdot (\mathbb{E}_{y,z \sim p_\theta(y,z)} [r^2(z|y)] - \mathbb{E}_{z \sim p_\theta(z)} [r^2(z)]) \right\} \\
&= \text{IB}_\beta[p_\theta(z|x)] + \mathbb{E}_{x,z \sim p_\theta(x,z)} \left[\left(\Delta\theta^T \frac{\partial}{\partial\theta} \log p_\theta(z|x) + \frac{1}{2} \Delta\theta^T \frac{1}{p_\theta(z|x)} \frac{\partial^2 p_\theta(z|x)}{\partial\theta^2} \Delta\theta \right) \log \frac{p_\theta(z|x)}{p_\theta(z)} \right] \\
&\quad - \beta \cdot \mathbb{E}_{y,z \sim p_\theta(y,z)} \left[\left(\Delta\theta^T \frac{\partial}{\partial\theta} \log p_\theta(z|y) + \frac{1}{2} \Delta\theta^T \frac{1}{p_\theta(z|y)} \frac{\partial^2 p_\theta(z|y)}{\partial\theta^2} \Delta\theta \right) \log \frac{p_\theta(z|y)}{p_\theta(z)} \right] \\
&\quad + \frac{1}{2} \left(\mathbb{E}_{x,z \sim p_\theta(x,z)} \left[\left(\Delta\theta^T \frac{\partial}{\partial\theta} \log p_\theta(z|x) \right)^2 \right] - \mathbb{E}_{z \sim p_\theta(z)} \left[\left(\Delta\theta^T \frac{\partial}{\partial\theta} \log p_\theta(z) \right)^2 \right] \right) \\
&\quad - \frac{\beta}{2} \left(\mathbb{E}_{y,z \sim p_\theta(y,z)} \left[\left(\Delta\theta^T \frac{\partial}{\partial\theta} \log p_\theta(z|y) \right)^2 \right] - \mathbb{E}_{z \sim p_\theta(z)} \left[\left(\Delta\theta^T \frac{\partial}{\partial\theta} \log p_\theta(z) \right)^2 \right] \right) \\
&= \text{IB}_\beta[p_\theta(z|x)] + \Delta\theta^T \left\{ \mathbb{E}_{x,z \sim p_\theta(x,z)} \left[\log \frac{p_\theta(z|x)}{p_\theta(z)} \frac{\partial}{\partial\theta} \log p_\theta(z|x) \right] \right. \\
&\quad \left. - \beta \cdot \mathbb{E}_{x,z \sim p_\theta(x,z)} \left[\log \frac{p_\theta(z|x)}{p_\theta(z)} \frac{\partial}{\partial\theta} \log p_\theta(z|x) \right] \right\} \\
&\quad + \frac{1}{2} \Delta\theta^T \left\{ (\mathcal{I}_{Z|X}(\theta) - \mathcal{I}_Z(\theta)) - \beta (\mathcal{I}_{Z|X}(\theta) - \mathcal{I}_Z(\theta)) \right\} \Delta\theta
\end{aligned}$$

In the last equality we have used $\mathbb{E}_{x,z \sim p_\theta(x,z)} \left[\frac{1}{p_\theta(z|x)} \frac{\partial^2 p_\theta(z|x)}{\partial\theta^2} \right] = \int dx p(x) \frac{\partial^2}{\partial\theta^2} \int dz p_\theta(z|x) = \int dx p(x) \frac{\partial^2}{\partial\theta^2} 1 = \mathbf{0}$, and similarly $\mathbb{E}_{y,z \sim p_\theta(y,z)} \left[\frac{1}{p_\theta(z|y)} \frac{\partial^2 p_\theta(z|y)}{\partial\theta^2} \right] = \mathbf{0}$. In other words, the $\|\Delta\theta\|^2$ terms in the first-order variation $\delta \text{IB}_\beta[p_\theta(z|x)]$ vanish, and the remaining $\|\Delta\theta\|^2$ are all in $\delta^2 \text{IB}_\beta[p_\theta(z|x)]$. Also in the last expression, $\mathcal{I}_Z(\theta) \equiv \int dz p_\theta(z) \left(\frac{\partial \log p_\theta(z)}{\partial\theta} \right) \left(\frac{\partial \log p_\theta(z)}{\partial\theta} \right)^T$ is the Fisher information matrix of θ for Z , $\mathcal{I}_{Z|X}(\theta) \equiv \int dx dz p(x) p_\theta(z|x) \left(\frac{\partial \log p_\theta(z|x)}{\partial\theta} \right) \left(\frac{\partial \log p_\theta(z|x)}{\partial\theta} \right)^T$, $\mathcal{I}_{Z|Y}(\theta) \equiv \int dy dz p(y) p_\theta(z|y) \left(\frac{\partial \log p_\theta(z|y)}{\partial\theta} \right) \left(\frac{\partial \log p_\theta(z|y)}{\partial\theta} \right)^T$ are the conditional Fisher information matrix [Zeg15] of θ for Z conditioned on X and Y , respectively.

Let us look at

$$\delta^2 \text{IB}_\beta[p_\theta(z|x)] = \frac{1}{2} \Delta\theta^T \left\{ (\mathcal{I}_{Z|X}(\theta) - \mathcal{I}_Z(\theta)) - \beta (\mathcal{I}_{Z|X}(\theta) - \mathcal{I}_Z(\theta)) \right\} \Delta\theta \quad (\text{A.40})$$

Firstly, note that $\delta^2 \text{IB}_\beta[p_\theta(z|x)]$ is a quadratic function of $\Delta\theta$, and the scale of $\Delta\theta$ does

not change the sign of $\delta^2 \text{IB}_\beta[p_\theta(z|x)]$, so the condition of $\forall \Delta\boldsymbol{\theta} \in \Theta, \delta^2 \text{IB}_\beta[p_\theta(z|x)] \geq 0$ is invariant to the scale of $\Delta\boldsymbol{\theta}$, and is describing the “curvature” in the infinitesimal neighborhood of $\boldsymbol{\theta}$. Therefore, $\Delta\boldsymbol{\theta}$ can explore any value in Θ . Secondly, we see that Eq. (A.40) is a special case of Eq. (A.33) with $\epsilon \cdot r(z|x) = \Delta\boldsymbol{\theta}^T \frac{\partial}{\partial \boldsymbol{\theta}} \log p_\theta(z|x)$. Therefore, The inequalities due to Jensen still hold: $\epsilon^2 (\mathbb{E}[r^2(z|x)] - \mathbb{E}[r^2(z)]) = \Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|X}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta} \geq 0$, $\epsilon^2 (\mathbb{E}[r^2(z|y)] - \mathbb{E}[r^2(z)]) = \Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|Y}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta} \geq 0$. If $\Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|Y}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta} > 0$, then the condition of $\forall \Delta\boldsymbol{\theta} \in \Theta, \delta^2 \text{IB}_\beta[p_\theta(z|x)] \geq 0$ is equivalent to $\forall \Delta\boldsymbol{\theta} \in \Theta$,

$$\beta \leq \frac{\Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|X}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta}}{\Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|Y}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta}}$$

i.e.

$$\beta \leq G_\Theta[p_\theta(z|x)] \equiv \inf_{\Delta\boldsymbol{\theta} \in \Theta} \frac{\Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|X}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta}}{\Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|Y}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta}} \quad (\text{A.41})$$

If $\Delta\boldsymbol{\theta}^T (\mathcal{I}_{Z|Y}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})) \Delta\boldsymbol{\theta} = 0$, we have that Eq. (A.40) always holds, which is a looser condition than Eq. (A.41). Above all, we have that the condition of $\forall \Delta\boldsymbol{\theta} \in \Theta, \delta^2 \text{IB}_\beta[p_\theta(z|x)]$ is equivalent to $\beta \leq G_\Theta[p_\theta(z|x)]$.

Moreover, $(G_\Theta[p_\theta(z|x)])^{-1}$ given by Eq. (A.41) has the format of a generalized Rayleigh quotient $R(A, B; x) \equiv \frac{\Delta\boldsymbol{\theta}^T A \Delta\boldsymbol{\theta}}{\Delta\boldsymbol{\theta}^T B \Delta\boldsymbol{\theta}}$ where $A = \mathcal{I}_{Z|Y}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})$ and $B = \mathcal{I}_{Z|X}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})$ are both Hermitian matrices⁶, which can be reduced to Rayleigh quotient $R(D, C^T \Delta\boldsymbol{\theta}) = \frac{(C^T \Delta\boldsymbol{\theta})^T D (C^T \Delta\boldsymbol{\theta})}{(C^T \Delta\boldsymbol{\theta})^T (C^T \Delta\boldsymbol{\theta})}$, with the transformation $D = C^{-1} A (C^T)^{-1}$ where CC^T is the Cholesky decomposition of $B = \mathcal{I}_{Z|X}(\boldsymbol{\theta}) - \mathcal{I}_Z(\boldsymbol{\theta})$. Moreover, we have that when $G_\Theta[p_\theta(z|x)]$ attains its minimum value, the Reyleigh quotient $R(D, C^T \Delta\boldsymbol{\theta})$ attains its maximum value of λ_{\max} with $C^T \Delta\boldsymbol{\theta} = v_{\max}$, i.e. $\Delta\boldsymbol{\theta} = (C^T)^{-1} v_{\max}$, where λ_{\max} is the largest eigenvalue of D and v_{\max} the corresponding eigenvector.

■

⁶Here all the Fisher information matrices are real symmetric, thus Hermitian.

A.3.4 Proof of Theorem 6

Proof. Define

$$T_\beta(\beta') := \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} [(\mathbb{E}_\beta[r^2(z|x)] - \mathbb{E}_\beta[r^2(z)]) - \beta' \cdot (\mathbb{E}_\beta[r^2(z|y)] - \mathbb{E}_\beta[r^2(z)])] \quad (\text{A.42})$$

where $\mathbb{E}_\beta[\cdot]$ denotes taking expectation w.r.t. the optimal solution $p_\beta^*(x, y, z) = p(x, y)p_\beta^*(z|x)$ at β . Using Lemma 13.2, we have that the IB phase transition as defined in Definition 4 corresponds to satisfying the following two equations:

$$T_\beta(\beta')|_{\beta'=\beta} \geq 0 \quad (\text{A.43})$$

$$\lim_{\beta' \rightarrow \beta^+} T_\beta(\beta') = 0^- \quad (\text{A.44})$$

Now we prove that $T_\beta(\beta')$ is continuous at $\beta' = \beta$, i.e. $\forall \varepsilon > 0$, $\exists \delta > 0$ s.t. $\forall \beta \in (\beta - \delta, \beta + \delta)$, we have $|T_\beta(\beta') - T_\beta(\beta)| < \epsilon$.

From Eq. (A.42), we have $T_\beta(\beta') - T_\beta(\beta) = -(\beta' - \beta) \cdot (\mathbb{E}_\beta[r^2(z|y)] - \mathbb{E}_\beta[r^2(z)])$. Since $r(z|x)$ is bounded, i.e. $\exists M > 0$ s.t. $\forall z \in \mathcal{Z}, x \in \mathcal{X}, |r(z|x)| \leq M$, we have

$$|\mathbb{E}_\beta[r^2(z|y)]| = \left| \mathbb{E}_\beta \left[(\mathbb{E}_{x \sim p(x|y,z)} [r(z|x)])^2 \right] \right| \leq \left| \mathbb{E}_\beta \left[(\mathbb{E}_{x \sim p(x|y,z)} [M])^2 \right] \right| = M^2$$

Similarly, we have

$$|\mathbb{E}_\beta[r^2(z)]| = \left| \mathbb{E}_\beta \left[(\mathbb{E}_{x \sim p(x|z)} [r(z|x)])^2 \right] \right| \leq \left| \mathbb{E}_\beta \left[(\mathbb{E}_{x \sim p(x|z)} [M])^2 \right] \right| = M^2$$

Hence, $|T_\beta(\beta') - T_\beta(\beta)| = |\beta' - \beta| |\mathbb{E}_\beta[r^2(z|y)] - \mathbb{E}_\beta[r^2(z)]| \leq 2|\beta' - \beta|M^2$.

To prove that $T_\beta(\beta')$ is continuous at $\beta' = \beta$, we have $\forall \varepsilon > 0$, $\exists \delta = \frac{\varepsilon}{2M^2} > 0$, s.t.

$\forall \beta' \in (\beta - \delta, \beta + \delta)$, we have

$$|T_\beta(\beta') - T_\beta(\beta)| \leq 2|\beta' - \beta|M^2 < 2\delta M^2 = 2\frac{\varepsilon}{2M^2}M^2 = \varepsilon$$

Hence $T_\beta(\beta')$ is continuous at $\beta' = \beta$.

Combining the continuity of $T_\beta(\beta')$ at $\beta' = \beta$, and Eq. (A.43) and (A.44), we have $T_\beta(\beta) = 0$, which is equivalent to $G[p_\beta^*(z|x)] = \beta$ after simple manipulation.

■

A.3.5 Invariance of $\mathcal{G}[r(z|x); p(z|x)]$ to addition of a global representation

Here we prove the following lemma:

Lemma 13.3. $\mathcal{G}[r(z|x); p(z|x)]$ defined in Lemma 5.1 is invariant to the transformation $r'(z|x) \leftarrow r(z|x) + s(z)$.

Proof. When we $r(z|x)$ is shifted by a global transformation $r'(z|x) \leftarrow r(z|x) + s(z)$, we have $r'(z) \leftarrow \mathbb{E}_{x \sim p(x|z)}[r(z|x) + s(z)] = \mathbb{E}_{x \sim p(x|z)}[r(z|x)] + s(z)\mathbb{E}_{x \sim p(x|z)}[1] = r(z) + s(z)$, and similarly $r'(z|y) \leftarrow r(z|y) + s(z)$.

The numerator of $\mathcal{G}[r(z|x); p(z|x)]$ is then

$$\begin{aligned}
& \mathbb{E}_{x,z \sim p(x,z)} \left[(r'(z|x))^2 \right] - \mathbb{E}_{z \sim p(z)} \left[(r'(z))^2 \right] \\
&= \mathbb{E}_{x,z \sim p(x,z)} \left[(r(z|x) + s(z))^2 \right] - \mathbb{E}_{z \sim p(z)} \left[(r(z) + s(z))^2 \right] \\
&= (\mathbb{E}_{x,z \sim p(x,z)} [r^2(z|x)] + 2\mathbb{E}_{x,z \sim p(x,z)} [r(z|x)s(z)] + \mathbb{E}_{x,z \sim p(x,z)} [s^2(z)]) \\
&\quad - (\mathbb{E}_{z \sim p(z)} [r^2(z)] + 2\mathbb{E}_{z \sim p(z)} [r(z)s(z)] + \mathbb{E}_{z \sim p(z)} [s^2(z)]) \\
&= (\mathbb{E}_{x,z \sim p(x,z)} [r^2(z|x)] + 2\mathbb{E}_{z \sim p(z)} [s(z)\mathbb{E}_{x \sim p(x|z)} [r(z|x)]] + \mathbb{E}_{z \sim p(z)} [s^2(z)]) \\
&\quad - (\mathbb{E}_{z \sim p(z)} [r^2(z)] + 2\mathbb{E}_{z \sim p(z)} [r(z)s(z)] + \mathbb{E}_{z \sim p(z)} [s^2(z)]) \\
&= (\mathbb{E}_{x,z \sim p(x,z)} [r^2(z|x)] + 2\mathbb{E}_{z \sim p(z)} [r(z)s(z)] + \mathbb{E}_{z \sim p(z)} [s^2(z)]) \\
&\quad - (\mathbb{E}_{z \sim p(z)} [r^2(z)] + 2\mathbb{E}_{z \sim p(z)} [r(z)s(z)] + \mathbb{E}_{z \sim p(z)} [s^2(z)]) \\
&= \mathbb{E}_{x,z \sim p(x,z)} [r^2(z|x)] - \mathbb{E}_{z \sim p(z)} [r^2(z)]
\end{aligned}$$

Symmetrically, we have

$$\mathbb{E}_{y,z \sim p(y,z)} \left[(r'(z|y))^2 \right] - \mathbb{E}_{z \sim p(z)} \left[(r'(z))^2 \right] = \mathbb{E}_{y,z \sim p(y,z)} [r^2(z|y)] - \mathbb{E}_{z \sim p(z)} [r^2(z)]$$

Therefore, $\mathcal{G}[r(z|x); p(z|x)] = \frac{\mathbb{E}_{x,z \sim p(x,z)} [r^2(z|x)] - \mathbb{E}_{z \sim p(z)} [r^2(z)]}{\mathbb{E}_{y,z \sim p(y,z)} [r^2(z|y)] - \mathbb{E}_{z \sim p(z)} [r^2(z)]}$ is invariant to $r'(z|x) \leftarrow r(z|x) + s(z)$. ■

A.3.6 Proof of Theorem 7

Proof. Using the condition of the theorem, we have that $\forall r(z|x) \in \mathbf{Q}_{\mathcal{Z}|\mathcal{X}}^0$, there exists $r_1(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ and $s(z) \in \{s : \mathcal{Z} \rightarrow \mathbf{R} | s \text{ bounded}\}$ s.t. $r(z|x) = r_1(z|x) + s(z)$. Note that the only difference between $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ and $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)}$ is that $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}$ requires $\mathbb{E}_{p(z|x)} [r_1(z|x)] = 0$.

Using Lemma 13.3, we have

$$\inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(0)}} \mathcal{G}[r(z|x); p(z|x)] = \inf_{r_1(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}} \mathcal{G}[r_1(z|x); p(z|x)] = G[p(z|x)]$$

where $r(z|x)$ doesn't have the constraint of $\mathbb{E}_{p(z|x)} [\cdot] = 0$.

After dropping the constraint of $\mathbb{E}_{z \sim p(z|x)} [r(z|x)] = 0$, again using Lemma 13.3, we can let $r(z) = \mathbb{E}_{x \sim p(x|z)} [r(z|x)] = 0$ (since we can perform the transformation $r'(z|x) \leftarrow$

$r(z|x) - r(z)$, so that the new $r'(z) \equiv 0$). Now we get a simpler formula for $G[p(z|x)]$, as follows:

$$G[p(z|x)] = \inf_{r(z|x) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(1)}} \frac{\mathbb{E}_{x,z \sim p(x,z)}[r^2(z|x)]}{\mathbb{E}_{y,z \sim p(y,z)} \left[(\mathbb{E}_{x \sim p(x|y,z)}[r(z|x)])^2 \right]} \quad (\text{A.45})$$

where $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(1)} := \{r : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbf{R} \mid \mathbb{E}_{x \sim p(x|z)}[r(z|x)] = 0, r \text{ bounded}\}$.

From Eq. (A.45), we can further require that $\mathbb{E}_{x,z \sim p(x,z)}[r^2(z|x)] = 1$. Define

$$\rho_s^2(X, Y; Z) := \sup_{f(X,Z) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(2)}} \mathbb{E}[(\mathbb{E}[f(X, Z)|Y, Z])^2] = \sup_{f(x,z) \in \mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(2)}} \mathbb{E}_{y,z \sim p(y,z)} \left[(\mathbb{E}_{x \sim p(x|y,z)}[f(x, z)])^2 \right] \quad (\text{A.46})$$

where⁷ $\mathcal{Q}_{\mathcal{Z}|\mathcal{X}}^{(2)} := \{r : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbf{R} \mid \mathbb{E}_{x \sim p(x|z)}[r(z|x)] = 0, \mathbb{E}_{x,z \sim p(x,z)}[r^2(z|x)] = 1, r \text{ bounded}\}$.

Comparing with Eq. (A.45), it immediately follows that

$$G[p(z|x)] = \frac{1}{\rho_s^2(X, Y; Z)}$$

(i) We only have to prove that $\rho_s(X, Y; Z) = \rho_r(X, Y; Z)$, where $\rho_r(X, Y; Z)$ is defined in Definition 5.

We have

$$\begin{aligned} & \mathbb{E}[f(X, Z)g(Y, Z)] \\ &= \int dxdydz p(x, y, z) f(x, z) g(y, z) \\ &= \int dydz p(y, z) g(y, z) \int dx p(x|y, z) f(x, z) \\ &\equiv \int dydz p(y, z) g(y, z) F(y, z) \\ &\leq \sqrt{\int dydz p(y, z) g^2(y, z)} \cdot \sqrt{\int dydz p(y, z) F^2(y, z)} \end{aligned}$$

where $F(y, z) := \int dx p(x|y, z) f(x, z)$. We have used Cauchy-Schwarz inequality, where the equality holds when $g(y, z) = \alpha F(y, z)$ for some α . Since $\mathbb{E}[g^2(y, z)] = 1$, we have

⁷In the definition of $\rho_r(X, Y; Z)$, we have used an equivalent format $f(x, z)$ instead of $r(z|x)$.

$\alpha^2 \mathbb{E}[F^2(y, z)] = 1$. Taking the supremum of $(\mathbb{E}[f(X, Z)g(Y, Z)])^2$ w.r.t. f and g , we have

$$\begin{aligned}
\rho_r^2(X, Y; Z) &= \sup_{(f(X, Z), g(Y, Z)) \in \mathbf{S}_1} (\mathbb{E}[f(X, Z)g(Y, Z)])^2 \\
&= \sup_{(f(x, z), g(y, z)) \in \mathbf{S}_1} \int dy dz p(y, z) g^2(y, z) \cdot \int dy dz p(y, z) F^2(y, z) \\
&= \sup_{f(x, z) \in \mathcal{Q}_{Z|X}^{(2)}} \int dy dz p(y, z) F^2(y, z) \\
&= \sup_{f(x, z) \in \mathcal{Q}_{Z|X}^{(2)}} \int dy dz p(y, z) \left(\int dx p(x|y, z) f(x, z) \right)^2 \\
&= \sup_{f(X, Z) \in \mathcal{Q}_{Z|X}^{(2)}} \mathbb{E}[(\mathbb{E}[f(X, Z)|Y, Z])^2] \\
&\equiv \rho_s^2(X, Y; Z)
\end{aligned}$$

Here \mathbf{S}_1 is defined in Definition 5. By definition both $\rho_r(X, Y; Z)$ and $\rho_s(X, Y; Z)$ take non-negative values. Therefore,

$$\rho_s(X, Y; Z) = \rho_r(X, Y; Z) \quad (\text{A.47})$$

(ii) Using the definition of $\rho_r(X, Y; Z)$, we have

$$\begin{aligned}
\rho_r^2(X, Y; Z) &\equiv \sup_{f(x, z) \in \mathcal{Q}_{Z|X}^{(2)}} \int dy dz p(y, z) \left(\int dx p(x|y, z) f(x, z) \right)^2 \\
&= \sup_{f(x, z) \in \mathcal{Q}_{Z|X}^{(2)}} \int dz p(z) \int dy p(y|z) \left(\int dx p(x|y, z) f(x, z) \right)^2 \\
&\equiv \sup_{f(x, z) \in \mathcal{Q}_{Z|X}^{(2)}} \int dz p(z) W[f(x, z)]
\end{aligned}$$

where $W[f(x, z)] := \int dy p(y|z) \left(\int dx p(x|y, z) f(x, z) \right)^2$.

Denote $c(z) := p(z) \mathbb{E}_{x \sim p(x|z)}[f^2(x, z)]$, we have $\int c(z) dz = \mathbb{E}_{x, z \sim p(x, z)}[f^2(x, z)] = 1$.

Then the supremum $\rho_r^2(X, Y; Z) = \sup_{f(x, z) \in \mathcal{Q}_{Z|X}^{(2)}} \int dz p(z) W[f(x, z)]$ is equivalent to the

following two-stage supremum:

$$\rho_r^2(X, Y; Z) = \sup_{c(z): \int c(z) dz = 1} \int dz p(z) \sup_{f(x,z) \in \mathcal{Q}_{Z|X}^{(3)}} W[f(x, z)] \quad (\text{A.48})$$

where $\mathcal{Q}_{Z|X}^{(3)} := \{\mathcal{X} \times \mathcal{Z} \rightarrow \mathbf{R} \mid \mathbb{E}_{x \sim p(x|z)}[f^2(x, z)] = \frac{c(z)}{p(z)}, \mathbb{E}_{x \sim p(x|z)}[f(x, z)] = 0, f \text{ bounded}\}$

We can think of the inner supremum $\sup_{f(x,z) \in \mathcal{Q}_{Z|X}^{(3)}} W[f(x, z)]$ as only w.r.t. x , for some given z .

Now let's consider another supremum:

$$\sup_{h(x) \in \mathbf{Q}_X^{(h)}} \int dy p(y|z) \left(\int dx p(x|y, z) h(x) \right)^2 \quad (\text{A.49})$$

where $\mathbf{Q}_X^{(h)} := \{h : \mathcal{X} \rightarrow \mathbf{R} \mid \mathbb{E}_{p(x|z)}[h(x)] = 0, \mathbb{E}_{p(x|z)}[h^2(x)] = 1, h \text{ bounded}\}$. Using similar technique in (ii), it is easy to prove that it equals $\rho_m^2(X, Y|Z)$ as defined in Definition 5.

Comparing Eq. (A.49) and the supremum:

$$\sup_{f(x,z) \in \mathcal{Q}_{Z|X}^{(3)}} W[f(x, z)]$$

we see that the only difference is that in the latter $\mathbb{E}_{x \sim p(x|z)}[f^2(x, z)]$ equals $\frac{c(z)}{p(z)}$ instead of 1.

Since $W[f(x, z)]$ is a quadratic functional of $f(x, z)$, we have

$$\sup_{f(x,z) \in \mathcal{Q}_{Z|X}^{(3)}} W[f(x, z)] = \frac{c(z)}{p(z)} \rho_m^2(X, Y|Z)$$

Therefore,

$$\begin{aligned}
\rho_r(X, Y; Z) &= \sup_{c(z): \int c(z) dz = 1} \int dz p(z) \sup_{f(x, z) \in \mathcal{Q}_{Z|\mathcal{X}}^{(3)}} W[f(x, z)] \\
&= \sup_{c(z): \int c(z) dz = 1} \int dz p(z) \frac{c(z)}{p(z)} \rho_m^2(X, Y|Z) \\
&= \sup_{c(z): \int c(z) dz = 1} \int dz c(z) \rho_m^2(X, Y|Z = z) \\
&= \sup_{Z \in \mathcal{Z}} \rho_m^2(X, Y|Z)
\end{aligned}$$

where in the last equality we have let $c(z)$ have ‘‘mass’’ only on the place where $\rho_m^2(X, Y|Z = z)$ attains supremum w.r.t. z .

(iii) When Z is a continuous variable, let $f(x, z) = f_X(x) \sqrt{\frac{\delta(z - z_0)}{p(z)}}$, where $\delta(\cdot)$ is the Dirac-delta function, z_0 is a parameter, $f_X(x) \in \mathbf{Q}_{\mathcal{X}|Z}^{(f)}$, with $\mathbf{Q}_{\mathcal{X}|Z}^{(f)} := \{f_X : \mathcal{X} \rightarrow \mathbf{R} \mid f_X \text{ bounded}; \forall Z \in \mathcal{Z} : \mathbb{E}_{X \sim p(X|Z)}[f_X(x)] = 0, \mathbb{E}_{X \sim p(X|Z)}[f_X^2(x)] = 1\}$. We have

$$\begin{aligned}
\mathbb{E}_{x \sim p(x|z)}[f(x, z)] &= \int p(x|z) f(x, z) dx \\
&= \sqrt{\frac{\delta(z - z_0)}{p(z)}} \int p(x|z) f_X(x) dx \\
&= \sqrt{\frac{\delta(z - z_0)}{p(z)}} \cdot 0 \\
&= 0
\end{aligned}$$

And

$$\begin{aligned}
\mathbb{E}[f^2(X, Z)] &= \int p(x, z) f^2(x, z) dx dz \\
&= \int p(x, z) f_X^2(x) \frac{\delta(z - z_0)}{p(z)} dx dz \\
&= \int dz \delta(z - z_0) \int dx p(x|z) f_X^2(x) dx \\
&= \int dz \delta(z - z_0) \cdot 1 \\
&= 1
\end{aligned}$$

Therefore, such constructed $f(x, z) = f_X(x) \sqrt{\frac{\delta(z - z_0)}{p(z)}}$ $\in \mathcal{Q}_{Z|\mathcal{X}}^{(2)}$, satisfying the requirement for $\rho_s(X, Y; Z)$ (which equals $\rho_r(X, Y; Z)$ by Eq. A.47).

Substituting in the special form of $f(x, z)$ into the expression of $\rho_s(X, Y; Z)$ in Eq. (A.46), we have

$$\begin{aligned}
& \sup_{f(x,z): f(x,z) = f_X(x)\sqrt{\frac{\delta(z-z_0)}{p(z)}}, f_X(x) \in \mathcal{Q}_{\mathcal{X}|Z}^{(f)}} \int dz p(z) \int dy p(y|z) \left(\int dx p(x|y, z) f(x, z) \right)^2 \\
&= \sup_{f_X(x) \in \mathcal{Q}_{\mathcal{X}|Z}^{(f)}, z_0 \in \mathcal{Z}} \int dz p(z) \int dy p(y|z) \left(\int dx p(x|y, z) f_X(x) \sqrt{\frac{\delta(z-z_0)}{p(z)}} \right)^2 \\
&= \sup_{z_0 \in \mathcal{Z}} \int dz p(z) \frac{\delta(z-z_0)}{p(z)} \sup_{f_X(x) \in \mathcal{Q}_{\mathcal{X}|Z}^{(f)}} \int dy p(y|z) \left(\int dx p(x|y, z) f_X(x) \right)^2 \\
&= \sup_{z_0 \in \mathcal{Z}} \int dz \delta(z-z_0) \sup_{f_X(x) \in \mathcal{Q}_{\mathcal{X}|Z}^{(f)}} \mathbb{E}[(\mathbb{E}[f_X(X)|Y, Z=z])^2 | Z=z] \\
&= \sup_{z_0 \in \mathcal{Z}} \int dz \delta(z-z_0) \rho_m^2(X, Y | Z=z) \\
&= \sup_{z_0 \in \mathcal{Z}} \rho_m^2(X, Y | Z=z_0) \\
&= \sup_{Z \in \mathcal{Z}} \rho_m^2(X, Y | Z)
\end{aligned}$$

We can identify $\sup_{f_X(X) \in \mathcal{Q}_{\mathcal{X}|Z}^{(f)}} \mathbb{E}[(\mathbb{E}[f_X(X)|Y, Z=z])^2 | Z=z]$ with $\rho_m^2(X, Y | Z=z)$ because $f_X(x)$ satisfies the requirement for conditional maximum correlation that $\mathbb{E}_{p(x|z)}[f_X(x)] = 0$ and $\mathbb{E}_{p(x|z)}[f_X^2(x)] = 1$, for any z , and using the same technique in (i), it is straightforward to prove that $\sup_{f_X(X) \in \mathcal{Q}_{\mathcal{X}|Z}^{(f)}} \mathbb{E}[(\mathbb{E}[f_X(X)|Y, Z=z])^2 | Z=z]$ equals the conditional maximum correlation as defined in Definition 5.

Since the conditional maximum correlation can be viewed as the maximum correlation between X and Y , where $X, Y \sim p(X, Y | Z)$, using the equality of $(\beta_0[h(x)])^{-1} = \rho_m^2(X, Y)$ (Eq. 7 in [WFCT19a]), we can identify the $h(x)$ in $\beta_0[h(x)]$ with the $f_X(X)$ here, and an optimal $f_X^*(X)$ that maximizes $\rho_m^2(X, Y | Z)$ is also an optimal $h^*(x)$ that minimizes $\beta_0[h(x)]$.

(iv) For discrete X, Y and Z and a given $Z = z$, let $Q_{X,Y|Z} := \left(\frac{p(x,y|z)}{\sqrt{p(x|z)p(y|z)}} \right)_{x,y} = \left(\frac{p(x,y)}{\sqrt{p(x)p(y)}} \sqrt{\frac{p(z|x)}{p(z|y)}} \right)_{x,y}$, we first prove that its second largest singular value is $\rho_m^2(X, Y | Z) =$

$$\sup_{(f,g) \in \mathbf{S}_2} \mathbb{E}_{x,y \sim p(x,y|z)}[f(x)g(y)] \quad (\mathbf{S}_2 \text{ is defined in Definition 5}).$$

Let column vectors $u_1 = \sqrt{p(x|z)}$ and $v_1 = \sqrt{p(y|z)}$ (note that z is given and fixed). Also let $u_2 = f(x)\sqrt{p(x|z)}$ and $v_2 = g(y)\sqrt{p(y|z)}$. Denote inner product $\langle u, v \rangle \equiv \sum_i u_i v_i$, and the length of a vector as $\|u\| = \sqrt{\langle u, u \rangle}$. We have $\|u_1\| = \|v_1\| = 1$ due to the normalization of probability, $\|u_2\| = \|v_2\| = 1$ due to $\mathbb{E}_{x \sim p(x|z)}[f^2(x)] = \mathbb{E}_{y \sim p(y|z)}[g^2(y)] = 1$, and $\langle u_1, u_2 \rangle = \langle v_1, v_2 \rangle = 0$ due to $\mathbb{E}_{x \sim p(x|z)}[f(x)] = \mathbb{E}_{y \sim p(y|z)}[g(y)] = 0$. Furthermore, we have

$$\sup_{(f,g) \in \mathbf{S}_2} \mathbb{E}_{x,y \sim p(x,y|z)}[f(x)g(y)] = \max_{u,v} u^T Q_{X,Y|Z} v$$

which is exactly the second largest singular value $\sigma_2(Z)$ of the matrix $Q_{X,Y|Z}$. Using the result in (ii), we have that $\rho_r(X, Y; Z) = \max_{Z \in \mathcal{Z}} \sigma_2(Z)$.

■

A.3.7 Subset separation at phase transitions

In this section we study the behavior of $p(z|x)$ on the phase transitions. We use the same categorical dataset (where $|X| = |Y| = |Z| = 3$ and $p(x)$ is uniform, and $p(y|x)$ is given in Fig. A-3). In Fig. A-4 we show the $p(z|x)$ on the simplex before and after each phase transition. We see that the first phase transition corresponds to the separation of $x = 2$ (belonging to $y = 2$) w.r.t. $x \in \{0, 1\}$ (belonging to classes $y \in \{0, 1\}$), on the $p(z|x)$

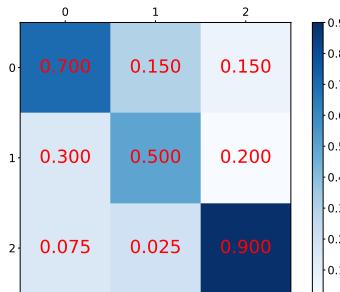


Figure A-3: $p(y|x)$ for the categorical dataset in Fig. 4-2 and Fig. A-4. The value in i^{th} row and j^{th} column denotes $p(y=j|x=i)$. $p(x)$ is uniform.

simplex. The second phase transition corresponds to the separation of $x = 0$ with $x = 1$. Therefore, each phase transition corresponds to the ability to distinguish subset of examples, and learning of new classes.

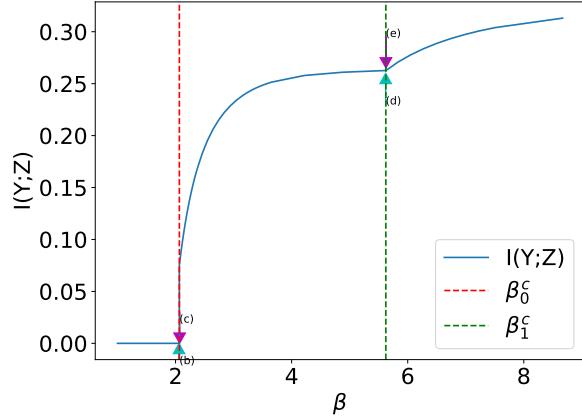
A.3.8 MNIST Experiment Details

We use the MNIST training examples with class 0, 1, 2, 3, with a hidden label-noise matrix as given in Fig. A-5, based on which at each minibatch we dynamically sample the observed label. We use conditional entropy bottleneck (CEB) [Fis18] as the variational IB objective, and run multiple independent instances with different the target β . We jump start learning by started training at $\beta = 100$ for 100 epochs, annealing β from 100 down to the target β over 600 epochs, and continue to train at the target epoch for another 800 epochs. The encoder is a three-layer neural net, where each hidden layer has 512 neurons and leakyReLU activation, and the last layer has linear activation. The classifier $p(y|z)$ is a 2-layer neural net with a 128-neuron ReLU hidden layer. The backward encoder $p(z|y)$ is also a 2-layer neural net with a 128-neuron ReLU hidden layer. We trained with Adam [KW13] at learning rate of 10^{-3} , and anneal down with factor $1/(1 + 0.01 \cdot \text{epoch})$. For Alg. 1, for the f_θ we use the same architecture as the encoder of CEB, and use $|Z| = 50$ in Alg. 1.

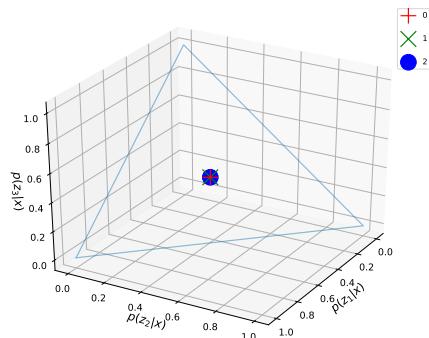
A.3.9 CIFAR10 Experiment Details

We use the same CIFAR10 class confusion matrix provided in [WFCT19a] to generate noisy labels with about 20% label noise on average (reproduced in Table A.5). We trained 28×1 Wide ResNet [HZRS16b, ZK16] models using the open source implementation from [CZM⁺18] as encoders for the Variational Information Bottleneck (VIB) [AFDM16]. The 10 dimensional output of the encoder parameterized a mean-field Gaussian with unit covariance. Samples from the encoder were passed to the classifier, a 2 layer MLP. The marginal distributions were mixtures of 500 fully covariate 10-dimensional Gaussians, all parameters of which are trained.

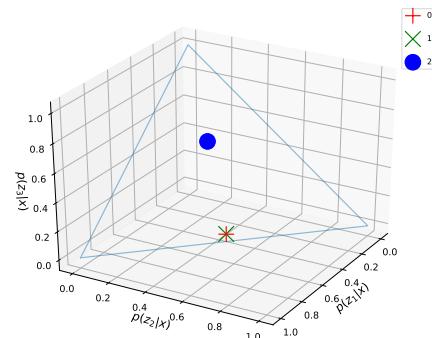
With this standard model, we trained 251 different models at β from 1.0 to 6.0 with step



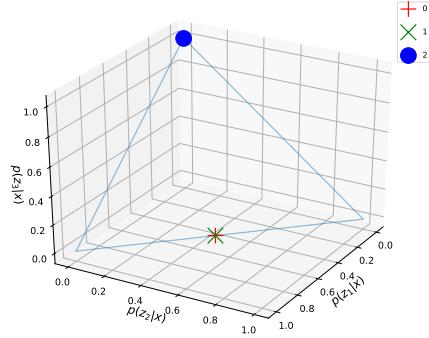
(a)



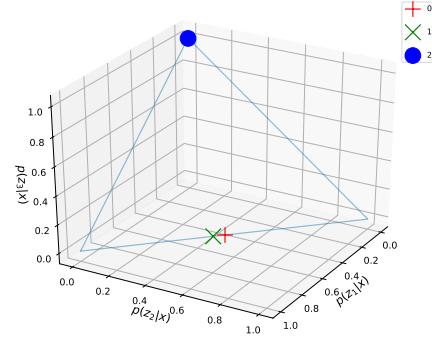
(b)



(c)



(d)



(e)

Figure A-4: (a) $I(Y; Z)$ vs. β for the dataset given in Fig. A-3. The phase transitions are marked with vertical dashed line, with $\beta_0^c = 2.065571$ and $\beta_1^c = 5.623333$. (b)-(e) Optimal $p_\beta^*(z|x)$ for four values of β , i.e. (b) $\beta = 2.060$, (c) $\beta = 2.070$, (d) $\beta = 5.620$ (e) $\beta = 5.625$ (their β values are also marked in (a)), where each marker denotes $p(z|x = i)$ for a given $i \in \{0, 1, 2\}$.

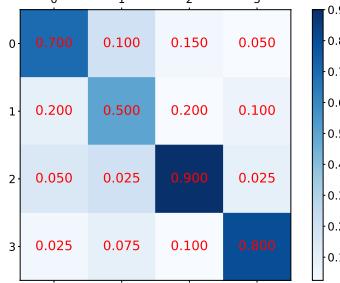


Figure A-5: Confusion matrix for MNIST experiment. The value in i^{th} row and j^{th} column denotes $p(\tilde{y} = j|y = i)$ for the label noise.

size of 0.02. As in [WFCT19a], we jump-start learning by annealing β from 100 down to the target β . We do this over the first 4000 steps of training. The models continued to train for another 56,000 gradient steps after that, a total of 600 epochs. We trained with Adam [KB15] at a base learning rate of 10^{-3} , and reduced the learning rate by a factor of 0.5 at 300, 400, and 500 epochs. The models converged to essentially their final accuracy within 40,000 gradient steps, and then remained stable. The accuracies reported in Figure 4-4 are averaged across five passes over the training set. We use $|Z| = 50$ in Alg. 1.

Table A.5: Class confusion matrix used in CIFAR10 experiments, reproduced from [WFCT19a]. The value in row i , column j means for class i , the probability of labeling it as class j . The mean confusion across the classes is 20%.

	Plane	Auto.	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck
Plane	0.82232	0.00238	0.021	0.00069	0.00108	0	0.00017	0.00019	0.1473	0.00489
Auto.	0.00233	0.83419	0.00009	0.00011	0	0.00001	0.00002	0	0.00946	0.15379
Bird	0.03139	0.00026	0.76082	0.0095	0.07764	0.01389	0.1031	0.00309	0.00031	0
Cat	0.00096	0.0001	0.00273	0.69325	0.00557	0.28067	0.01471	0.00191	0.00002	0.0001
Deer	0.00199	0	0.03866	0.00542	0.83435	0.01273	0.02567	0.08066	0.00052	0.00001
Dog	0	0.00004	0.00391	0.2498	0.00531	0.73191	0.00477	0.00423	0.00001	0
Frog	0.00067	0.00008	0.06303	0.05025	0.0337	0.00842	0.8433	0	0.00054	0
Horse	0.00157	0.00006	0.00649	0.00295	0.13058	0.02287	0	0.83328	0.00023	0.00196
Ship	0.1288	0.01668	0.00029	0.00002	0.00164	0.00006	0.00027	0.00017	0.83385	0.01822
Truck	0.01007	0.15107	0	0.00015	0.00001	0.00001	0	0.00048	0.02549	0.81273

A.4 Appendix for Chapter 5

A.4.1 Binning can be practically lossless

If the conditional probability distribution $p_1(w) \equiv P(Y=1|W=w)$ is a slowly varying function and the range of W is divided into tiny bins, then $p_1(w)$ will be almost constant within each bin and so binning W (discarding information about the exact position of W within a bin) should destroy almost no information about Y .

This intuition is formalized by the following theorem, which says that a random variable W can be binned into a finite number of bins at the cost of losing arbitrarily little information about Y .

Theorem 14. *Binning can be practically lossless: Given a random variable $Y \in \{1, 2\}$ and a uniformly distributed random variable $W \in [0, 1]$ such that the conditional probability distribution $p_1(w) \equiv P(Y=1|W=w)$ is monotonic, there exists for any real number $\epsilon > 0$ a vector $\mathbf{b} \in \mathbb{R}^{N-1}$ of bin boundaries such that the information reduction*

$$\Delta I \equiv I[W, Y] - I[B(W, \mathbf{b}), Y] < \epsilon,$$

where B is the binning function defined by equation (5.17).

Proof. The binned bivariate probability distribution is

$$P_{ij} \equiv P(Z=j, Y = i) = \int_{b_{j-1}}^{b_j} p_i(w)dw \quad (\text{A.50})$$

with marginal distribution

$$P_j^Z \equiv P(Z=j) = b_j - b_{j-1}. \quad (\text{A.51})$$

Let $\bar{p}_i(w)$ denote the piecewise constant function that in the j^{th} bin $b_{j-1} < w \leq b_j$ takes the average value of $p_i(w)$ in that bin, *i.e.*,

$$\bar{p}_i(w) \equiv \frac{1}{b_j - b_{j-1}} \int_{b_{j-1}}^{b_j} p_i(w)dw = \frac{P_{ij}}{P_j^Z}. \quad (\text{A.52})$$

These definitions imply that

$$-\sum_{j=1}^N P_{ij} \log \frac{P_{ij}}{P_j^Z} = \int_0^1 h[\bar{p}_i(w)] dw, \quad (\text{A.53})$$

where $h(x) \equiv -x \log x$. Since $h(x)$ vanishes at $x = 0$ and $x = 1$ and takes its intermediate maximum value at $x = 1/e$, the function

$$h_*(x) \equiv \begin{cases} h(x) & \text{if } x < e^{-1}, \\ 2h(e^{-1}) - h(x) & \text{if } x \geq e^{-1} \end{cases} \quad (\text{A.54})$$

is continuous and increases monotonically for $x \in [0, 1]$, with $h'_* = |h'(x)|$. This means that if we define the non-negative monotonic function

$$h_+(w) \equiv h_*[p_1(w)] - h_*[p_2(w)],$$

it changes at least as fast as either of its terms, so that for any $w_1, w_2 \in [0, 1]$, we have

$$\begin{aligned} |h[p_i(w_2)] - h[p_i(w_1)]| &\leq |h_*[p_i(w_2)] - h_*[p_i(w_1)]| \\ &\leq |h_+(w_2) - h_+(w_1)|. \end{aligned} \quad (\text{A.55})$$

We will exploit this bound to limit how much $h[p_i(w)]$ can vary within a bin. Since $h_+(0) \geq 0$ and $h_+(1) \leq 2h_*(1) = 4/e \ln 2 \approx 2.12 < 3$, we pick $N - 1$ bins boundaries b_k implicitly defined by

$$h_+(b_j) = h_+(0) + [h_+(1) - h_+(0)] \frac{j}{N} \quad (\text{A.56})$$

for some integer $N \gg 1$. Using equation (A.55), this implies that

$$|h[\bar{p}_i(w)] - h[p_i(w)]| \leq \frac{h_+(1) - h_+(0)}{N} < \frac{3}{N}. \quad (\text{A.57})$$

The mutual information between two variables is given by $I(Y, U) = H(Y) - H(Y|U)$, where the second term (the conditional entropy) is given by the following expressions in the

cases that we need:

$$H(Y|Z) = - \sum_{i=1}^N \sum_{j=1}^2 P_{ij} \log \frac{P_{ij}}{P_i}, \quad (\text{A.58})$$

$$H(Y|W) = - \sum_{i=1}^2 \int_0^1 p_i(w) \log p_i(w) dw. \quad (\text{A.59})$$

The information loss caused by our binning is therefore

$$\begin{aligned} \Delta I &= I(W, Y) - I(Z, Y) = H(Y|Z) - H(Y|W) \\ &= - \sum_{i=1}^2 \left(\sum_{j=1}^N P_{ij} \log \frac{P_{ij}}{P_j^Z} + \int_0^1 h[p_i(w)] dw \right) \\ &= \sum_{i=1}^2 \int_0^1 (h[\bar{p}_i(w)] - h[p_i(w)]) dw \\ &\leq \sum_{i=1}^2 \int_0^1 |h[\bar{p}_i(w)] - h[p_i(w)]| dw \\ &< \sum_{i=1}^2 \int_0^1 \frac{3}{N} = \frac{6}{N}, \end{aligned} \quad (\text{A.60})$$

where we used equation (A.53) to obtain the 3rd row and equation (A.57) to obtain the last row. This means that however small an information loss tolerance ϵ we want, we can guarantee $\Delta I < \epsilon$ by choosing $N > 6/\epsilon$ bins placed according to equation (A.56), which completes the proof. \blacksquare

Note that the proof still holds if the function $p_i(w)$ is not monotonic, as long as the number of times M that it changes direction is finite: in that case, we can simply repeat the above-mentioned binning procedure separately in the $M + 1$ intervals where $p_i(w)$ is monotonic, using $N > 6/\epsilon$ bins in each interval, *i.e.*, a total of $N > 6M/\epsilon$ bins.

A.4.2 More varying conditional probability boosts mutual information

Mutual information is loosely speaking a measure of how far a probability distribution P_{ij} is from being separable, *i.e.*, a product of its two marginal distributions.⁸ If all conditional probabilities for one variable Y given the other variable Z are identical, then the distribution is separable and the mutual information $I(Z, Y)$ vanishes, so one may intuitively expect that making conditional probabilities more different from each other will increase $I(Z, Y)$. The following theorem formalizes this intuition in a way that enables Theorem 9.

Theorem 15. *Consider two discrete random variables $Z \in \{1, \dots, n\}$ and $Y \in \{1, 2\}$ and define $P_i \equiv P(Z = i)$,*

$p_i \equiv P(Y = 1|Z = i)$, so that the joint probability distribution $P_{ij} \equiv P(Z = i, Y = j)$ is given by

$P_{i1} = P_i p_i$, $P_{i2} = P_i(1 - p_i)$. If two conditional probabilities p_k and p_l differ, then we increase the mutual information $I(Y, Z)$ if we bring them further apart by adjusting P_{kj} and P_{lj} in such a way that both marginal distributions remain unchanged.

Proof. The only such change that keep the marginal distributions for both Z and Y unchanged takes the form

$$\begin{pmatrix} P_1 p_1 & \cdots & P_k p_k - \epsilon & \cdots & P_l p_l + \epsilon & \cdots \\ P_1(1 - p_1) & \cdots & P_k(1 - p_k) + \epsilon & \cdots & P_l(1 - p_l) - \epsilon & \cdots \end{pmatrix}$$

where the parameter ϵ that must be kept small enough for all probabilities to remain non-negative. Without loss of generality, we can assume that $p_k < p_l$, so that we make the conditional probabilities

$$P(Y = 1|Z = k) = \frac{P_{k1}}{P_k} = p_k - \epsilon/P_k, \quad (\text{A.61})$$

$$P(Y = 1|Z = l) = \frac{P_{l1}}{P_l} = p_l + \epsilon/P_l \quad (\text{A.62})$$

⁸Specifically, the mutual information is the Kullback–Leibler divergence between the bivariate probability distribution and the product of its marginals.

more different when increasing ϵ from zero. Computing and differentiating the mutual information with respect to ϵ , most terms cancel and we find that

$$\frac{\partial I(Z, Y)}{\partial \epsilon} \Big|_{\epsilon=0} = \log \left[\frac{1/p_k - 1}{1/p_l - 1} \right] > 0 \quad (\text{A.63})$$

which means that adjusting the probabilities with a sufficiently tiny $\epsilon > 0$ will increase the mutual information, completing the proof. \blacksquare

A.5 Appendix for Chapter 6

A.5.1 Hyperparameter λ selection

For selecting an appropriate hyperparameter λ , we run our experiments for the synthetic dataset with $\lambda = 0.001, 0.002, 0.005, 0.01, 0.02, 0.05$. For each experiment involving N time series, we append $\lceil N/2 \rceil$ independent time series $v_{t-1}^{(s)}$ ($s = 1, 2, \dots, \lceil N/2 \rceil$) to \mathcal{X}_{t-1} , generated by randomly sampling $\lceil N/2 \rceil$ time series from \mathcal{X}_{t-1} and performing random permutation across the examples. We also append $\lceil N/2 \rceil$ time series $w_t^{(i)}$, $i = 1, 2, \dots, \lceil N/2 \rceil$ to $x_t^{(i)}$, such that $w_t^{(i)} = X_{t-1}^{(i)} \cdot Q$, where Q is a fixed random $K \times 1$ matrix, so that we know $X_{t-1}^{(i)}$ causes $w_t^{(i)}$, and $v_{t-1}^{(s)}$ does not cause $w_t^{(i)}$ for any i, s . We apply Alg. 3 to the augmented dataset, and produce the estimated predictive strength W_{ji} from $[\mathcal{X}_{t-1}, \mathbf{v}_{t-1}]$ to $[\mathbf{x}_t, \mathbf{w}_t]$. For each hyperparameter λ , we then fit a Gaussian distribution $G_{v \rightarrow w}$ to the estimated predictive strengths from $v_{t-1}^{(s)}$ to $w_t^{(i)}$ ($s = 1, 2, \dots, \lceil N/2 \rceil; j = 1, 2, \dots, \lceil N/2 \rceil$), and fit another Gaussian distribution $G_{x \rightarrow w}$ to the estimated predictive strengths from $X_{t-1}^{(i)}$ to $w_t^{(i)}$, $i = 1, 2, \dots, \lceil N/2 \rceil$, and select the λ such that the upper 4σ value of $G_{v \rightarrow w}$ is smaller than the lower 4σ value of $G_{x \rightarrow w}$. In this way, for the known causal and non-causal relations, they are sufficiently apart. We find that $\lambda = 0.001$ and $\lambda = 0.002$ satisfy this criterion, while larger λ fails to satisfy. We then set $\lambda = 0.002$ for all our experiments.

A.5.2 Proof and analysis of the Minimum Predictive Information regularized risk

In this section we prove the three properties of W_{ji} in Section 6.2.2, and analyze why it is likely to select variables that directly causes the variable of interest.

Firstly we state the assumption that will be used throughout this section:

Assumption 1. Assume that $f_\theta \in \mathcal{F}$ is a continuous function and has enough capacity so that it can approximate any $\int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)}$. Let $j \neq i$ and assume that $P(X_{t-1}^{(j)})$ has support with intrinsic dimension of KM .

Also we emphasize that in this paper, the expected risks (with symbol R) are w.r.t. the distributions, and the empirical risks (with symbol \hat{R}) are w.r.t. a dataset drawn from the distribution, with finite number of examples. The theorems in this paper are all proved w.r.t. distributions (assuming infinite number of examples). Sample complexity results will be left for future work.

Before going forward with the main proof, we first prove the following lemma.

Proving a lemma

Lemma 15.1. *Suppose that Assumption 1 holds. Denote*

$$R_{\mathbf{X},x^{(i)}}^{\text{MSE}}[f_\theta] = \mathbb{E}_{\mathbf{X}_{t-1},x_t^{(i)}} \left[\left(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \right)^2 \right]$$

as the standard MSE loss, we have

$$\operatorname{argmin}_{f_\theta} R_{\mathbf{X},x^{(i)}}^{\text{MSE}}[f_\theta] = \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)} \quad (\text{A.64})$$

and

$$\min_{f_\theta} R_{\mathbf{X},x^{(i)}}^{\text{MSE}}[f_\theta] = \mathbb{E}_{\mathbf{X}_{t-1},x_t^{(i)}} \left[\left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)} \right)^2 \right] \quad (\text{A.65})$$

In other words, for the MSE loss, its minimum is attained when $f_\theta(\mathbf{X}_{t-1})$ is the expectation of $x_t^{(i)}$ conditioned on \mathbf{X}_{t-1} .

Proof. The proof of the lemma is adapted from [Pap85]. The risk

$$\begin{aligned} R_{\mathbf{X},x^{(i)}}^{\text{MSE}}[f_\theta] &= \mathbb{E}_{\mathbf{X}_{t-1},x_t^{(i)}} \left[\left(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \right)^2 \right] \\ &= \int d\mathbf{X}_{t-1} dx_t^{(i)} \cdot P(\mathbf{X}_{t-1}, x_t^{(i)}) \left(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \right)^2 \\ &= \int d\mathbf{X}_{t-1} P(\mathbf{X}_{t-1}) \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) \left(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \right)^2 \end{aligned}$$

Note that here $(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}))^2 \equiv \langle x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}), x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \rangle$ is an inner product in \mathbf{R}^M .

For any \mathbf{X}_{t-1} , treating $f_\theta(\mathbf{X}_{t-1}) \in \mathbf{R}^M$ as a vector, let's calculate its value such that the integral $F(f_\theta(\mathbf{X}_{t-1})) := \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) \left(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \right)^2$ attains its minimum.

Let

$$\begin{aligned} 0 &= \frac{\partial}{\partial f_\theta(\mathbf{X}_{t-1})} F(f_\theta(\mathbf{X}_{t-1})) \\ &= \frac{\partial}{\partial f_\theta(\mathbf{X}_{t-1})} \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) \left(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \right)^2 \\ &= -2 \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) \left(x_t^{(i)} - f_\theta(\mathbf{X}_{t-1}) \right) \end{aligned}$$

we have

$$\begin{aligned} \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)} &= \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) f_\theta(\mathbf{X}_{t-1}) \\ &= f_\theta(\mathbf{X}_{t-1}) \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) \\ &= f_\theta(\mathbf{X}_{t-1}) \end{aligned}$$

Therefore, for any \mathbf{X}_{t-1} , $f_\theta(\mathbf{X}_{t-1}) = \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)}$ is the only stationary point for $F(f_\theta(\mathbf{X}_{t-1}))$.

Taking the second derivative, we have

$$\frac{\partial^2}{(\partial f_\theta(\mathbf{X}_{t-1}))^2} F(f_\theta(\mathbf{X}_{t-1})) = 2 \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) \mathbf{I} = 2\mathbf{I}$$

where \mathbf{I} is an $M \times M$ identity matrix, which is always positive definite.

Therefore, for any \mathbf{X}_{t-1} , $f_\theta(\mathbf{X}_{t-1}) = \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)}$ is the only global minimum of $F(f_\theta(\mathbf{X}_{t-1}))$ w.r.t. $f_\theta(\mathbf{X}_{t-1})$.

Since

$$R_{\mathbf{X}, x^{(i)}}[f_\theta] = \int d\mathbf{X}_{t-1} P(\mathbf{X}_{t-1}) F(f_\theta(\mathbf{X}_{t-1}))$$

The minimum of the risk $R_{\mathbf{X}, x^{(i)}}[f_\theta]$ is attained iff $F(f_\theta(\mathbf{X}_{t-1}))$ attains minimum at every \mathbf{X}_{t-1} , i.e.,

$$f_\theta(\mathbf{X}_{t-1}) = \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)}$$

is true for any \mathbf{X}_{t-1} . Given Assumption 1, we know that $f_\theta \in \mathcal{F}$ has enough capacity such that it can approximate any $\int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)}$. Therefore,

$$\operatorname{argmin}_{f_\theta} R_{\mathbf{X}, x^{(i)}}^{\text{MSE}}[f_\theta] = \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)}$$

and

$$\min_{f_\theta} R_{\mathbf{X}, x^{(i)}}^{\text{MSE}}[f_\theta] = \mathbb{E}_{\mathbf{X}_{t-1}, x_t^{(i)}} \left[\left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | \mathbf{X}_{t-1}) x_t^{(i)} \right)^2 \right]$$

■

Proof of the three properties of W_{ji}

The three properties are

- (1) If $x^{(j)} \perp\!\!\!\perp x^{(i)}$, then $W_{ji} = 0$.
- (2) W_{ji} is invariant to affine transformation of each individual $X_{t-1}^{(k)}$, $k = 1, 2, \dots, N$.
- (3) W_{ji} is invariant to reparameterization of θ in f_θ (the mapping remains the same).

Proof. (1) If $x^{(j)} \perp\!\!\!\perp x^{(i)}$, then $X_{t-1}^{(j)} \perp\!\!\!\perp x_t^{(i)}$. Since $\tilde{X}_{t-1}^{(j)(\eta_j)} = X_{t-1}^{(j)} + \eta_j \cdot \epsilon_j$ where $\epsilon_j \sim N(\mathbf{0}, \mathbf{I})$, we have $\tilde{X}_{t-1}^{(j)(\eta_j)} \perp\!\!\!\perp x_t^{(i)}$. Recall Eq. (6.2):

$$R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\eta}] = \mathbb{E}_{\mathbf{X}_{t-1}, x_t^{(i)}, \boldsymbol{\epsilon}} \left[\left(x_t^{(i)} - f_\theta(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) \right)^2 \right] + \lambda \cdot \sum_{k=1}^N I(\tilde{X}_{t-1}^{(k)(\eta_k)}; X_{t-1}^{(k)})$$

let $f_{\theta_\eta^*} = \operatorname{argmin}_{f_\theta} R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\eta}]$ given a certain $\boldsymbol{\eta}$, we have

$$\begin{aligned} f_{\theta_\eta^*}(\tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})}) &= \operatorname{argmin}_{f_\theta} R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\eta}] \\ &= \operatorname{argmin}_{f_\theta} \mathbb{E}_{\tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})}, x_t^{(i)}} \left[\left(x_t^{(i)} - f_\theta(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) \right)^2 \right] \\ &= \int dx_t^{(i)} P(x_t^{(i)} | \tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})}) x_t^{(i)} \end{aligned}$$

where the second equality is due to that the mutual information term in $R_{\mathbf{X},x^{(i)}}[f_\theta, \boldsymbol{\eta}]$ does not depend on f_θ , and the last equality is due to Lemma 15.1. Let $\tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})(\hat{j})} = \tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})} \setminus \tilde{X}_{t-1}^{(j)(\eta_j)}$, since $\tilde{X}_{t-1}^{(j)(\eta_j)} \perp\!\!\!\perp x_t^{(i)}$, we have

$$\begin{aligned} P(x_t^{(i)} | \tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})}) &\equiv P(x_t^{(i)} | \tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})(\hat{j})}, \tilde{X}_{t-1}^{(j)(\eta_j)}) \\ &= P(x_t^{(i)} | \tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})(\hat{j})}) \end{aligned}$$

Therefore,

$$f_{\theta^*}(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) = \int dx_t^{(i)} P(x_t^{(i)} | \tilde{\mathcal{X}}_{t-1}^{(\boldsymbol{\eta})(\hat{j})}) x_t^{(i)}$$

which *does not* depend on $\tilde{X}_{t-1}^{(j)(\eta_j)}$. Finally, we have

$$\begin{aligned} &\min_{(f_\theta, \boldsymbol{\eta})} R_{\mathbf{X},x^{(i)}}[f_\theta, \boldsymbol{\eta}] \\ &= \min_{\boldsymbol{\eta}} [R_{\mathbf{X},x^{(i)}}[f_{\theta^*}, \boldsymbol{\eta}]] \\ &= \min_{\boldsymbol{\eta}} \left[\mathbb{E}_{\mathbf{X}_{t-1}, x_t^{(i)}, \boldsymbol{\epsilon}} \left[\left(x_t^{(i)} - f_{\theta^*}(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) \right)^2 \right] + \lambda \cdot \sum_{k=1}^N I(\tilde{X}_{t-1}^{(k)(\eta_k)}; X_{t-1}^{(k)}) \right] \\ &= \min_{\boldsymbol{\eta}} \left[\left(\mathbb{E}_{\mathbf{X}_{t-1}, x_t^{(i)}, \boldsymbol{\epsilon}} \left[\left(x_t^{(i)} - f_{\theta^*}(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})(\hat{j})}) \right)^2 \right] + \lambda \cdot \sum_{k \neq j} I(\tilde{X}_{t-1}^{(k)(\eta_k)}; X_{t-1}^{(k)}) \right) + I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) \right] \end{aligned}$$

For the last equality, the elements in the parenthesis (\cdot) does not depend on $\tilde{X}_{t-1}^{(j)(\eta_j)}$, and only the $I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)})$ term depends on $\tilde{X}_{t-1}^{(j)(\eta_j)}$. Therefore, at the minimization of the whole objective $R_{\mathbf{X},x^{(i)}}[f_\theta, \boldsymbol{\eta}]$, we have $I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)})$ attains its minimum of 0, at which $\eta_j^* \rightarrow \infty$. By the definition of W_{ji} , we have $W_{ji} = I(\tilde{X}_{t-1}^{(j)(\eta_j^*)}; X_{t-1}^{(j)}) = 0$. Proof completes.

In essence, the proof states that if $x^{(j)} \perp\!\!\!\perp x^{(i)}$, then at the minimization of the whole objective, the MSE term does not depend on $X_{t-1}^{(j)}$ or $\tilde{X}_{t-1}^{(j)(\eta_j)}$, and the mutual information term $I(\tilde{X}_{t-1}^{(j)(\eta_j^*)}; X_{t-1}^{(j)})$ w.r.t. time series j can be independently minimized and approach 0.

(2) Suppose that we replace $X_{t-1}^{(j)}$ by $X'_{t-1}^{(j)} = a \cdot X_{t-1}^{(j)} + b$ where $a, b \in \mathbb{R}$. Let $\eta'_j = a \cdot \eta_j$. We have $\tilde{X}'_{t-1}^{(j)(\eta'_j)} = X'_{t-1}^{(j)} + \eta'_j \cdot \epsilon_j = a(X_{t-1}^{(j)} + \eta_j \cdot \epsilon_j) + b = a \cdot \tilde{X}_{t-1}^{(j)(\eta_j)} + b$, and therefore

$I\left(\tilde{X}_{t-1}^{'(j)(\eta'_j)}; X_{t-1}^{'(j)}\right) = I\left(a \cdot \tilde{X}_{t-1}^{(j)(\eta_j)} + b; a \cdot X_{t-1}^{(j)} + b\right) = I\left(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}\right)$, where the last equality is due to that mutual information is invariant to invertible transformations. Furthermore, due to Assumption 1, we can find another $f_{\theta'}$ which undoes this affine transformation on $\tilde{X}_{t-1}^{(j)(\eta_j)}$, so the MSE term can be kept the same. Therefore, we have a one-to-one mapping between the original $X_{t-1}^{(j)}, \eta_j, f_\theta$ and the new $X_{t-1}^{'(j)}, \eta'_j, f_{\theta'}$ such that value of the MSE term and the mutual information term remain unchanged. Thus at the minimization of the objective, W_{ji} remains the same.

(3) This is trivial to prove. We see that in $R_{\mathbf{X},x^{(i)}}[f_\theta, \eta]$, the MSE term remains the same if the mapping f remains the same, regardless of how we parameterize f in terms of parameter θ . The second term does not depend on f_θ . Therefore, at the minimization of $R_{\mathbf{X},x^{(i)}}[f_\theta, \eta]$, the $W_{ji} = I(\tilde{X}_{t-1}^{(j)(\eta_j^*)}; X_{t-1}^{(j)})$ is invariant to the reparameterization of the same f in terms of parameter θ . As a direct corollary, W_{ji} is insensitive to the network architecture, as long as the capacity is enough (provided with sufficient number of examples). This is confirmed in Table SA.2 in Appendix A.5.7.

Note that L1 and group L1 regularization do not have this property, since they explicitly regularize on the parameter θ . ■

Analysis of the minimum predictive information-regularized risk

After proving the three properties of W_{ji} , now we analyze why the minimum predictive information-regularized risk is likely to select the variables that directly cause $x_t^{(i)}$, under some additional assumptions. We first state the additional assumption needed to perform the analysis, then we restate the definitions of direct causality to make our statements more rigorous. We then prove two lemmas in Appendix A.5.2, and finally perform the analysis in Appendix A.5.2.

Assumption 2. Assume that causal sufficiency [PJS17] is satisfied, i.e. the observed time series $x^{(i)}, i = 1, 2, \dots, N$ are all the variables that take part in the dynamics (no hidden confounding variables). Also assume that in the response function Eq. (6.1), the noise variable $u_i, i = 1, 2, \dots, N$ are effective variables, so each h_i is not a deterministic mapping. Assume that by saying “causality”, we mean “causality in mean”.

To make our statement of causality more rigorous, here we restate the definition of direct (structural) causality [WCL11] using our notations of the system Eq. (6.1). This definition is a natural extension to Pearl causality [Pea09] in canonical settable systems [WC09, WCL11], which formalizes time series in its full generality.

Direct (structural) causality [WCL11] *We say $X_{t-1}^{(j)}, j \neq i$ does not directly (structurally) cause $x_t^{(i)}$, if for all possible values of $\mathbf{X}_{t-1}^{(j)}$ and $u_l, l \in 1, 2, \dots, N$, the function $X_{t-1}^{(j)} \rightarrow h_i(\mathbf{X}_{t-1}, u_i)$ is constant in $X_{t-1}^{(j)}$. Otherwise, we say $X_{t-1}^{(j)}$ directly (structurally) causes $x_t^{(i)}$.*

The relationship between direct causality and Granger causality in Section 6.2.1 is the following Lemma, which states that for our system, Granger causality is a sufficient condition for direct (structural) causality.

Lemma 15.2. *Assuming causal sufficiency, for system Eq. 6.1, for any $i, j \in \{1, 2, \dots, N\}, i \neq j$, if $X_{t-1}^{(j)}$ Granger-causes $x_t^{(i)}$, then $X_{t-1}^{(j)}$ directly structurally causes $x_t^{(i)}$.*

Proof. We base the proof on the Theorem 5.6 in [WCL11]. Firstly, by definition, the system Eq. (6.1) belongs to the canonical settable system (Def. 3.3 in [WCL11]), on which their Theorem 5.6 is based. To prove that in our system Granger causality can deduce direct structural causality, we only have to prove that the assumption A.1 and assumption A.2 in [WCL11] are satisfied by our system. If we identify our $x_t^{(i)}$ with their $Y_{1,t}$, our \mathbf{X}_{t-1} with their \mathbf{Y}_{t-1} , our $x_t^{(j)}$ with their $Y_{2,t}$, our $u_{i,t}$ (our u_i at time t) with their $U_{1,t}$, our $u_{j,t}$ with their $U_{2,t}$, their $\mathbf{Z}_t = \emptyset$, $\mathbf{W}_t = \emptyset$, then our system Eq. (6.1) satisfies their Assumption A.1. Additionally, by definition, our $u_i \in R^M, i = 1, 2, \dots, N$ are random variables that are mutually independent, and also independent of any $X_{t-1}^{(i)}, x_t^{(i)}, i \in \{1, 2, \dots, N\}$. Therefore, our system satisfies their strict exogeneity $(\mathbf{Y}_{t-1}, \mathbf{Z}_t) \perp\!\!\!\perp U_{1,t}$ (in our representation $(\mathbf{X}_{t-1}, \emptyset) \perp\!\!\!\perp u_{i,t}$), which is a sufficient condition for Assumption A.2. Therefore, both their Assumption A.1 and Assumption A.2 are satisfied by our system Eq. (6.1). Applying their Theorem 5.6, we prove Lemma 15.2.

■

Therefore, for our system Eq. (6.1), applying the results by [WCL11], we have that Granger causality is a sufficient condition for direct structural causality. The reason that here Granger

causality can deduce direct structural causality is in part due to the fact that for system Eq. (6.1), conditional exogeneity [WCL11] is automatically satisfied.

Note that the reverse of the statement is not true, i.e. a failed Granger causality test does not necessarily imply that there is no direct structural causality (White & Lu [WL10] give several examples, and also note that these instances are exceptional).

After stating Assumption 2 and clarifying the definition of causalities, now we prove two lemmas, which are important for the analysis of our objective.

Minimum MSE with different variables

Lemma 15.3. *Suppose that Assumption 1 and 2 holds, and $X_{t-1}^{(U)}, X_{t-1}^{(V)}, X_{t-1}^{(W)} \subset \mathcal{X}_{t-1}$ are mutually exclusive sets of variables satisfying*

$$X_{t-1}^{(W)} \perp\!\!\!\perp x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}, \quad X_{t-1}^{(V)} \not\perp\!\!\!\perp x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(W)}$$

Then

$$\min_{f_\theta} \mathbb{E}_{X_{t-1}^{(U)}, X_{t-1}^{(V)}, x_t^{(i)}} \left[\left(x_t^{(i)} - f_\theta(X_{t-1}^{(U)}, X_{t-1}^{(V)}) \right)^2 \right] < \min_{f_\theta} \mathbb{E}_{X_{t-1}^{(U)}, X_{t-1}^{(V)}, x_t^{(i)}} \left[\left(x_t^{(i)} - f_\theta(X_{t-1}^{(U)}, X_{t-1}^{(W)}) \right)^2 \right]$$

Fig. SA.1 below shows the relations between the variables, where the dashed arrows denote the potential existence of causal relations between variables. We see that *conditioned* on $(X_{t-1}^{(U)}, X_{t-1}^{(V)})$, we have $x_t^{(i)}$ and $X_{t-1}^{(W)}$ are independent, while *conditioned* on $(X_{t-1}^{(U)}, X_{t-1}^{(W)})$, we have $x_t^{(i)}$ and $X_{t-1}^{(V)}$ are *not* independent. Lemma 15.3 states that under the above scenario and under Assumptions 1 and 2, using $X_{t-1}^{(U)}$ and $X_{t-1}^{(V)}$ to predict $x_t^{(i)}$ can achieve a lower MSE than using $X_{t-1}^{(U)}$ and $X_{t-1}^{(W)}$ to predict $x_t^{(i)}$.

Proof. Since Assumption 1 holds, according to Lemma 15.1, Lemma 15.3 is equivalent to

$$\begin{aligned} & \mathbb{E}_{X_{t-1}^{(U)}, X_{t-1}^{(V)}, x_t^{(i)}} \left[\left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)} \right)^2 \right] \\ & < \mathbb{E}_{X_{t-1}^{(U)}, X_{t-1}^{(W)}, x_t^{(i)}} \left[\left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(W)}) x_t^{(i)} \right)^2 \right] \end{aligned}$$

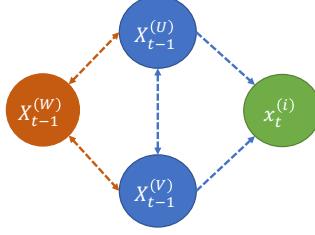


Figure S A.1: Diagram of variables for Lemma 15.3. The dashed arrows denote the possible existence of causal relations between variables.

We have

$$\begin{aligned}
& \mathbb{E}_{X_{t-1}^{(U)}, X_{t-1}^{(W)}, x_t^{(i)}} \left[\left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(W)}) x_t^{(i)} \right)^2 \right] \\
&= \int dX_{t-1}^{(U)} dX_{t-1}^{(W)} dx_t^{(i)} P(X_{t-1}^{(U)}, X_{t-1}^{(W)}, x_t^{(i)}) \left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(W)}) x_t^{(i)} \right)^2 \\
&= \int dX_{t-1}^{(U)} dX_{t-1}^{(V)} dX_{t-1}^{(W)} dx_t^{(i)} P(X_{t-1}^{(U)}, X_{t-1}^{(V)}, X_{t-1}^{(W)}, x_t^{(i)}) \left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(W)}) x_t^{(i)} \right)^2 \\
&= \int dX_{t-1}^{(U)} dX_{t-1}^{(V)} dX_{t-1}^{(W)} P(X_{t-1}^{(U)}, X_{t-1}^{(V)}) P(X_{t-1}^{(W)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) \\
&\quad \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) \left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)} \right)^2 \\
&> \int dX_{t-1}^{(U)} dX_{t-1}^{(V)} dX_{t-1}^{(W)} P(X_{t-1}^{(U)}, X_{t-1}^{(V)}) P(X_{t-1}^{(W)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) \\
&\quad \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) \left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)} \right)^2 \\
&= \int dX_{t-1}^{(U)} dX_{t-1}^{(V)} P(X_{t-1}^{(U)}, X_{t-1}^{(V)}) \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) \left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)} \right)^2 \\
&= \mathbb{E}_{X_{t-1}^{(U)}, X_{t-1}^{(V)}, x_t^{(i)}} \left[\left(x_t^{(i)} - \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)} \right)^2 \right]
\end{aligned}$$

The third equality (the one before the inequality) is due to that $X_{t-1}^{(W)} \perp\!\!\!\perp x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}$, leading to $P(X_{t-1}^{(U)}, X_{t-1}^{(V)}, X_{t-1}^{(W)}, x_t^{(i)}) = P(X_{t-1}^{(U)}, X_{t-1}^{(V)}) P(X_{t-1}^{(W)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)})$.

The inequality step first uses the Assumption 2 that the noise variables u_i are effective arguments of the response functions h_i , and that each h_i is “causality in mean”. Therefore, $\int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)} \neq \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)}$. Using Lemma 15.1, we

have $f_\theta(X_{t-1}^{(U)}, X_{t-1}^{(V)}) = \int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) x_t^{(i)}$ minimizes

$$\int dx_t^{(i)} P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) \left(x_t^{(i)} - f_\theta(X_{t-1}^{(U)}, X_{t-1}^{(V)}) \right)^2$$

hence the inequality. ■

Using Lemma 15.3 recursively, we see that using variables that directly causes $x^{(i)}$ to predict $x^{(i)}$ can achieve the lowest MSE. Formalizing the above intuition, we have

Lemma 15.4. *Suppose that Assumption 1 and 2 holds, and $X_{t-1}^{(D)} \subseteq \mathcal{X}_{t-1}$ are the set of variables that directly causes $x_t^{(i)}$. Then $\forall X_{t-1}^{(S)} \subseteq \mathcal{X}_{t-1}$ with $X_{t-1}^{(S)} \neq X_{t-1}^{(D)}$, we have*

$$\min_{f_\theta} \mathbb{E}_{X_{t-1}^{(D)}} \left[\left(x_t^{(i)} - f_\theta(X_{t-1}^{(D)}) \right)^2 \right] < \min_{f_\theta} \mathbb{E}_{X_{t-1}^{(S)}} \left[\left(x_t^{(i)} - f_\theta(X_{t-1}^{(S)}) \right)^2 \right]$$

Specifically, we have

$$\min_{f_\theta} \mathbb{E}_{X_{t-1}^{(D)}} \left[\left(x_t^{(i)} - f_\theta(X_{t-1}^{(D)}) \right)^2 \right] < \min_{f_\theta} \mathbb{E}_{X_{t-1}^{(\hat{D})}} \left[\left(x_t^{(i)} - f_\theta(X_{t-1}^{(\hat{D})}) \right)^2 \right]$$

where $X_{t-1}^{(\hat{D})} = \mathcal{X}_{t-1} \setminus X_{t-1}^{(D)}$.

Proof. For any $X_{t-1}^{(S)}$, let $X_{t-1}^{(U)} = X_{t-1}^{(D)} \cap X_{t-1}^{(S)}$, $X_{t-1}^{(V)} = X_{t-1}^{(D)} \setminus X_{t-1}^{(S)}$, $X_{t-1}^{(W)} = X_{t-1}^{(S)} \setminus X_{t-1}^{(D)}$. Then $X_{t-1}^{(U)}$, $X_{t-1}^{(V)}$, $X_{t-1}^{(W)}$ are mutually exclusive, and $X_{t-1}^{(D)} = X_{t-1}^{(U)} \cup X_{t-1}^{(V)}$, $X_{t-1}^{(S)} = X_{t-1}^{(U)} \cup X_{t-1}^{(W)}$. Now we prove that $\forall X_{t-1}^{(S)} \subseteq \mathcal{X}_{t-1}$ with $X_{t-1}^{(S)} \neq X_{t-1}^{(D)}$, the corresponding $X_{t-1}^{(U)}$, $X_{t-1}^{(V)}$, $X_{t-1}^{(W)}$, $x_t^{(i)}$ satisfy the condition for Lemma 15.3. Since $X_{t-1}^{(D)}$ are the set of variables that directly causes $x_t^{(i)}$, there does not exist a $X_{t-1}^{(S)}$ such that the corresponding $X_{t-1}^{(V)} \perp\!\!\!\perp x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(W)}$ (otherwise it violates the direct causality). Thus $X_{t-1}^{(V)} \not\perp\!\!\!\perp x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(W)}$. To prove $X_{t-1}^{(W)} \perp\!\!\!\perp x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}$, note that $X_{t-1}^{(W)}$ does not directly cause $x_t^{(i)}$, then $X_{t-1}^{(W)}$ does not Granger-cause $x_t^{(i)}$, i.e. $P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}) = P(x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}, X_{t-1}^{(W)})$, which is equivalent to $X_{t-1}^{(W)} \perp\!\!\!\perp x_t^{(i)} | X_{t-1}^{(U)}, X_{t-1}^{(V)}$. The special case of $X_{t-1}^{(\hat{D})}$ follows directly that $X_{t-1}^{(\hat{D})} = \mathcal{X}_{t-1} \setminus X_{t-1}^{(D)} \neq X_{t-1}^{(D)}$ and letting $X_{t-1}^{(S)} = X_{t-1}^{(\hat{D})}$. ■

Qualitative and quantitative behaviors of the mutual information-regularized risk

In this section, we analyze the qualitative and quantitative behaviors of the mutual information-regularized risk (Eq. 6.2), with varying noise levels η_j . For each variable $X_{t-1}^{(j)} \in \mathcal{X}_{t-1}$, $j = 1, 2, \dots, N$, define $\rho_j = \tanh(I(X_{t-1}^{(j)}; \tilde{X}_{t-1}^{(j)(\eta_j)})) \in [0, 1]$ as a “rescaled” mutual information between $X_{t-1}^{(j)}$ and $\tilde{X}_{t-1}^{(j)(\eta_j)}$. When $\eta_j = \mathbf{0}$ so that $\tilde{X}_{t-1}^{(j)(\eta_j)} = X_{t-1}^{(j)}$, $\rho_j = 1$, at which the input $X_{t-1}^{(j)}$ is fully preserved. When all elements of $\eta_j \rightarrow \infty$, $\rho_j = 0$, at which $X_{t-1}^{(j)}$ is fully corrupted. Denoting $\boldsymbol{\rho} = (\rho_1, \rho_2, \dots, \rho_N)$, we can then rewrite the mutual information-regularized risk (Eq. 6.2) as

$$R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\rho}] = \text{MMSE}^{(i)}(\boldsymbol{\rho}) + \lambda \cdot \sum_{j=1}^N \operatorname{arctanh}(\rho_j) \quad (\text{A.66})$$

where $\text{MMSE}^{(i)}(\boldsymbol{\rho}) = \min_{\boldsymbol{\eta}, f_\theta} \mathbb{E}_{\mathbf{X}_{t-1}, x_t^{(i)}, \boldsymbol{\epsilon}} \left[\left(x_t^{(i)} - f_\theta(\tilde{\mathbf{X}}_{t-1}^{(\boldsymbol{\eta})}) \right)^2 \right]$ subject to

$$\rho_j = \tanh(I(X_{t-1}^{(j)}; \tilde{X}_{t-1}^{(j)(\eta_j)})), j = 1, 2, \dots, N$$

Let $X_{t-1}^{(D)} \subseteq \mathcal{X}_{t-1}$ be the set of variables that directly causes $x_t^{(i)}$, and denote the corresponding set of ρ_j as $\boldsymbol{\rho}^{(D)}$. Denote $X_{t-1}^{(\hat{D})} = \mathcal{X}_{t-1} \setminus X_{t-1}^{(D)}$ and the corresponding set of ρ_j as $\boldsymbol{\rho}^{(\hat{D})}$. For any $i = 1, 2, \dots, N$, it is easy to see that $\text{MMSE}^{(i)}(\boldsymbol{\rho})$ has the following properties:

1. $\text{MMSE}^{(i)}(\boldsymbol{\rho})$ attains maximum at $\boldsymbol{\rho} = \mathbf{0}$.
2. $\text{MMSE}^{(i)}(\boldsymbol{\rho})$ is monotonically decreasing w.r.t. each ρ_j .
3. $\text{MMSE}^{(i)}(\boldsymbol{\rho})|_{\boldsymbol{\rho}^{(D)}=\mathbf{1}, \boldsymbol{\rho}^{(\hat{D})}=\mathbf{0}} < \text{MMSE}^{(i)}(\boldsymbol{\rho})|_{\boldsymbol{\rho}^{(D)}=\mathbf{0}, \boldsymbol{\rho}^{(\hat{D})}=\mathbf{1}}$ (using Lemma 15.4).
4. $\text{MMSE}^{(i)}(\boldsymbol{\rho})$ attains minimum at $\boldsymbol{\rho}^{(D)} = \mathbf{1}$. $\text{MMSE}^{(i)}(\boldsymbol{\rho})|_{\boldsymbol{\rho}^{(D)}=\mathbf{1}}$ is constant w.r.t. $\boldsymbol{\rho}^{(\hat{D})}$.

To get a better intuition of the landscape of $R_{\mathbf{X}, x^{(i)}}[f_\theta, \boldsymbol{\rho}]$, let’s investigate a simple example.

Let the response function be:

$$\begin{cases} x_t^{(1)} := h_1(u_1) = \sqrt{\Sigma_x} \cdot u_1 \\ x_t^{(2)} := h_2(x_{t-1}^{(1)}, u_2) = x_{t-1}^{(1)} + \sqrt{\Omega_x} \cdot u_2 \\ x_t^{(3)} := h_3(x_{t-1}^{(2)}, u_3) = x_{t-1}^{(2)} + \sqrt{\Omega_y} \cdot u_3 \end{cases} \quad (\text{A.67})$$

where u_1, u_2, u_3 are independent unit Gaussian variables, and $\mathcal{X}_{t-1} = (X_{t-1}^{(1)}, X_{t-1}^{(2)}, X_{t-1}^{(3)}) = ((x_{t-2}^{(1)}, x_{t-1}^{(1)}), (x_{t-2}^{(2)}, x_{t-1}^{(2)}), (x_{t-2}^{(3)}, x_{t-1}^{(3)}))$. For $R_{\mathbf{X}, x^{(3)}}[f_\theta, \rho] = \text{MMSE}^{(3)}(\rho) + \lambda \cdot \sum_{j=1}^3 \operatorname{arctanh}(\rho_j)$, since only $x_{t-2}^{(1)}$ and $x_{t-1}^{(2)}$ are d-connected to $x_t^{(3)}$, at the minimization of $R_{\mathbf{X}, x^{(3)}}[f_\theta, \rho]$, only $x_{t-2}^{(1)}$ and $x_{t-1}^{(2)}$ may have a finite $\eta_{j,l}^*$ (the other $\eta_{j,l}^*$ are all infinite). Therefore, setting the $\eta_{j,l}$ not corresponding to $x_{t-2}^{(1)}$ and $x_{t-1}^{(2)}$ as infinity, and let $\tilde{x}_{t-2}^{(1)} = x_{t-2}^{(1)} + \eta_x \cdot \epsilon_x$, $\tilde{x}_{t-1}^{(2)} = x_{t-1}^{(2)} + \eta_y \cdot \epsilon_y$, ϵ_x and ϵ_y being independent unit Gaussian variables. Let $f_\theta(x_{t-2}^{(1)}, x_{t-1}^{(2)}) = a \cdot x_{t-2}^{(1)} + b \cdot x_{t-1}^{(2)}$, then we can get an analytic expression for $R_{\mathbf{X}, x^{(3)}}[f_\theta, \eta_x, \eta_y]$:

$$\begin{aligned} R_{\mathbf{X}, x^{(3)}}[f_\theta, \eta_x, \eta_y] &= a^2 \Sigma_x + (b-1)^2 (\Sigma_x + \Omega_x) + a^2 \eta_x^2 + b^2 \eta_y^2 + 2a(b-1) \Sigma_x + \Omega_y \\ &\quad + \frac{\lambda}{2} \log \left(1 + \frac{\Sigma_x}{\eta_x^2} \right) + \frac{\lambda}{2} \log \left(1 + \frac{\Sigma_x + \Omega_x}{\eta_y^2} \right) \end{aligned}$$

Minimizing $R_{\mathbf{X}, x^{(3)}}[f_\theta, \eta_x, \eta_y]$ w.r.t. a and b , we get

$$\begin{aligned} a^* &= \frac{\eta_y^2 \Sigma_x}{\eta_x^2 \eta_y^2 + \eta_x^2 \Sigma_x + \eta_y^2 \Sigma_x + \eta_x^2 \Omega_x + \Omega_x \Sigma_x} \\ b^* &= \frac{\eta_x^2 (\Sigma_x + \Omega_x) + \Sigma_x \Omega_x}{\eta_x^2 \eta_y^2 + \eta_x^2 \Sigma_x + \eta_y^2 \Sigma_x + \eta_x^2 \Omega_x + \Omega_x \Sigma_x} \end{aligned}$$

Substituting into $R_{\mathbf{X}, x^{(3)}}[f_\theta, \eta_x, \eta_y]$, we have

$$\begin{aligned} R_{\mathbf{X}, x^{(3)}}[\eta_x, \eta_y] &= \min_{f_\theta} R_{\mathbf{X}, x^{(3)}}[f_\theta, \eta_x, \eta_y] \\ &= \frac{\eta_y^2 (\Sigma_x \Omega_x + \eta_x^2 (\Sigma_x + \Omega_x))}{\eta_x^2 \eta_y^2 + \eta_x^2 \Sigma_x + \eta_y^2 \Sigma_x + \eta_x^2 \Omega_x + \Omega_x \Sigma_x} + \frac{\lambda}{2} \log \left(1 + \frac{\Sigma_x}{\eta_x^2} \right) + \frac{\lambda}{2} \log \left(1 + \frac{\Sigma_x + \Omega_x}{\eta_y^2} \right) \end{aligned}$$

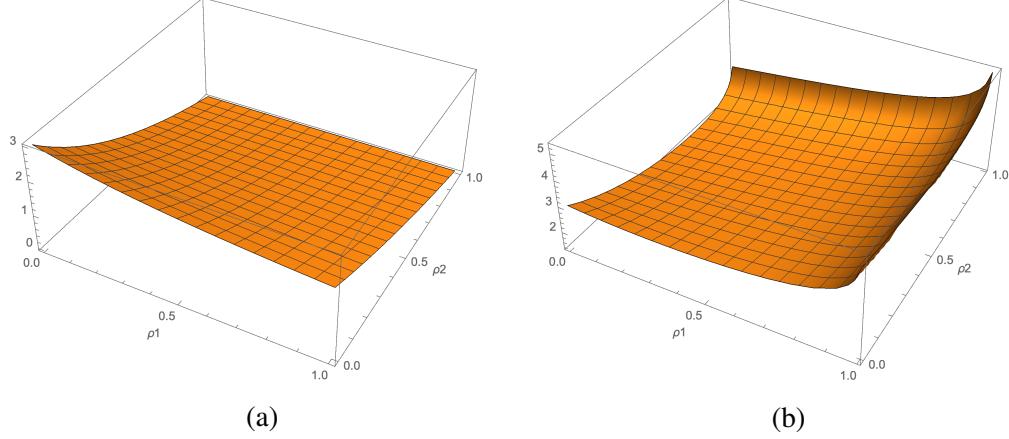


Figure S A.2: (a) $\text{MMSE}^{(3)}(\boldsymbol{\rho})$ and (b) $R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}]$ in section A.5.2, for $\Sigma_x = 1$, $\Omega_x = 2$, $\lambda = 1$.

Here we have neglected the constant Ω_y . To obtain $R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}]$, let $\rho_1 = \tanh\left(\frac{1}{2}\log\left(1 + \frac{\Sigma_x}{\eta_x^2}\right)\right)$, $\rho_2 = \tanh\left(\frac{1}{2}\log\left(1 + \frac{\Sigma_x + \Omega_x}{\eta_x^2}\right)\right)$, we have $\eta_x^2 = \frac{1-\rho_1}{2\rho_1}\Sigma_x$, $\eta_y^2 = \frac{1-\rho_2}{2\rho_2}(\Sigma_x + \Omega_x)$. Substituting, we have

$$\begin{aligned} R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}] &= \text{MMSE}^{(3)}(\boldsymbol{\rho}) + \lambda \cdot \sum_{j=1}^2 \operatorname{arctanh}(\rho_j) \\ &= \frac{(\rho_2 - 1)(\Sigma_x + \Omega_x)((\rho_1 - 1)\Sigma_x - (\rho_1 + 1)\Omega_x)}{(1 + \rho_1 + \rho_2 - 3\rho_1\rho_2)\Sigma_x + (1 + \rho_1)(1 + \rho_2)\Omega_x} + \lambda \cdot \operatorname{arctanh}(\rho_1) + \lambda \cdot \operatorname{arctanh}(\rho_2) \end{aligned}$$

Fig. SA.2 shows the landscape of $\text{MMSE}^{(3)}(\boldsymbol{\rho})$ and $R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}]$, for $\Sigma_x = 1$, $\Omega_x = 2$, $\lambda = 1$. We see that $\text{MMSE}^{(3)}(\boldsymbol{\rho})$ satisfies the above mentioned four properties. Particularly, $\text{MMSE}^{(3)}(\boldsymbol{\rho})|_{\rho_1=1,\rho_2=0} > \text{MMSE}^{(3)}(\boldsymbol{\rho})|_{\rho_1=0,\rho_2=1}$. After adding $\lambda \cdot \operatorname{arctanh}(\rho_1) + \lambda \cdot \operatorname{arctanh}(\rho_2)$, the $R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}]$ has global minimum along $\rho_1 = 0$ largely due to this property. Therefore, for this particular example, when $R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}]$ is minimized, $\rho_1 = 0$, i.e. $I(x_{t-2}^{(1)}, \tilde{x}_{t-2}^{(1)(\eta_1^*)}) = 0$.

By varying the value of λ , we can tune the relative influence of the two terms $\text{MMSE}^{(3)}(\boldsymbol{\rho})$ and $\sum_{j=1}^2 \operatorname{arctanh}(\rho_j)$. The landscape corresponding to $\lambda = 0.01, 0.5, 2, 10$ are plotted in Fig. SA.3. We see that when $\lambda \ll 1$, the MMSE term dominates, and it is possible that the global minimum of $R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}]$ is not at $\rho_1 = 0$. This is similar to the effect of a L1 regularization, where if the coefficient λ for the L1 is vanishingly small, the L1 regularization

will barely influence the loss landscape. When λ is not vanishingly small, as in Fig. SA.3 (b), we see that the global minimum of $R_{\mathbf{X},x^{(3)}}[\boldsymbol{\rho}]$ lies on $\rho_1 = 0$. When $\lambda \rightarrow +\infty$, the $\sum_{j=1}^2 \operatorname{arctanh}(\rho_j)$ term dominates and the global minimum is at $\rho_1 = 0, \rho_2 = 0$.

In general, we expect $R_{\mathbf{X},x^{(i)}}[\boldsymbol{\rho}]$ behave qualitatively similar. When $\lambda \rightarrow +\infty$, the global minimum for $R_{\mathbf{X},x^{(i)}}[\boldsymbol{\rho}]$ is at $\boldsymbol{\rho}^* = 0$. As we ramp down λ , the dimension that has largest influence on MMSE will first host the global minimum with nonzero ρ_j^* , which is most likely the variable that directly causes $x_i^{(i)}$. When λ is further ramping down, we expect that the variables that host the global minimum with nonzero ρ_j will more likely be those that directly causes $x_i^{(i)}$, due to the landscape influenced by the four properties of MMSE. This can justify the mutual information-regularized risk as a good objective for causal discovery/variable selection. The experiments in the paper will empirically test the performance of the mutual information-regularized risk.

A.5.3 Upper bound for the mutual information-regularized risk

In this section, we prove that $I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) \leq \frac{1}{2} \sum_{l=1}^{KM} \log \left(1 + \frac{\operatorname{Var}(X_{t-1,l}^{(j)})}{\eta_{j,l}^2} \right)$. We formally state the theorem as follows:

Theorem 16. *Let $\tilde{X}_{t-1}^{(j)(\eta_j)} := X_{t-1}^{(j)} + \eta_j \cdot \epsilon_j$, $j = 1, 2, \dots, N$ be the noise-corrupted inputs with learnable noise amplitudes $\eta_j \in \mathbf{R}^{KM}$, and $\epsilon_j \sim N(\mathbf{0}, \mathbf{I})$. We have*

$$I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) \leq \frac{1}{2} \sum_{l=1}^{KM} \log \left(1 + \frac{\operatorname{Var}(X_{t-1,l}^{(j)})}{\eta_{j,l}^2} \right) \quad (\text{A.68})$$

where l is the l^{th} element of a vector, $\operatorname{std}(X_{t-1,l}^{(j)})$ is the standard deviation of $X_{t-1,l}^{(j)}$ across t . The equality is reached when $X_{t-1}^{(j)}$ obeys a multivariate Gaussian distribution with diagonal covariance matrix Σ satisfying $\Sigma_{l,l} = \operatorname{Var}(X_{t-1,l}^{(j)}) + \eta_{j,l}^2$.

Proof. We have

$$\begin{aligned} I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) &= H(\tilde{X}_{t-1}^{(j)(\eta_j)}) - H(\eta_j \cdot \epsilon_j) \\ &= H(\tilde{X}_{t-1}^{(j)(\eta_j)}) - \left(\frac{KM}{2} \log(2\pi e) + \sum_{l=1}^{KM} \frac{1}{2} \log(\eta_{j,l}^2) \right) \end{aligned}$$

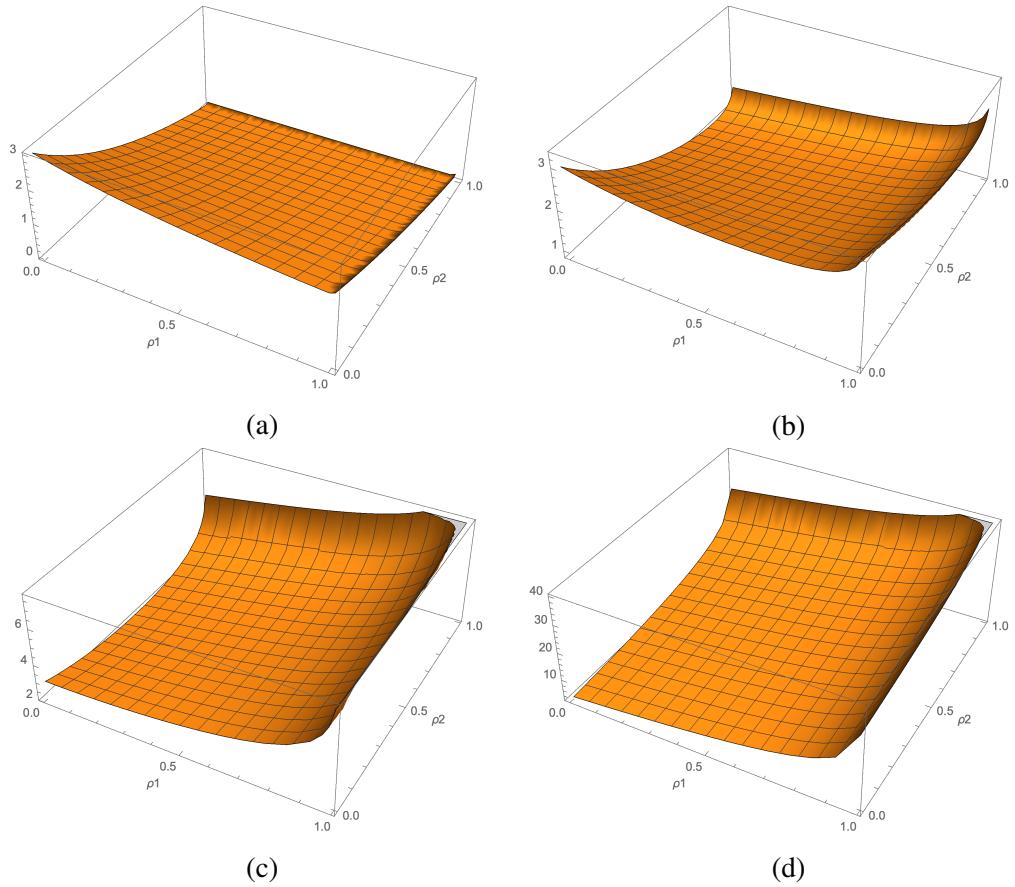


Figure S A.3: (a) $R_{X,x^{(3)}}[\rho]$ for (a) $\lambda = 0.01$, (b) $\lambda = 0.5$, (c) $\lambda = 2$ and (d) $\lambda = 10$ in section A.5.2, for $\Sigma_x = 1$, $\Omega_x = 2$.

Here $H(\cdot)$ is differential entropy. For $\tilde{X}_{t-1}^{(j)(\eta_j)}$, its variance at the l^{th} dimension is

$$\begin{aligned}\text{Var}(\tilde{X}_{t-1,l}^{(j)(\eta_j)}) &= \text{Var}(X_{t-1,l}^{(j)} + \eta_j \cdot \epsilon_j) \\ &= \text{Var}(X_{t-1,l}^{(j)}) + \text{Var}(\eta_j \cdot \epsilon_j) \\ &= \text{Var}(X_{t-1,l}^{(j)}) + \eta_{j,l}^2\end{aligned}$$

The second equality is due to that $X_{t-1}^{(j)}$ is independent of ϵ_j . Using the principle of maximum entropy, the distribution that maximizes $H(\tilde{X}_{t-1}^{(j)(\eta_j)})$ subject to the constraint of $\text{Var}(\tilde{X}_{t-1,l}^{(j)}) = \text{Var}(X_{t-1,l}^{(j)}) + \eta_{j,l}^2, l = 1, 2, \dots, KM$ is a Gaussian distribution whose diagonal covariance matrix Σ satisfies $\Sigma_{l,l} = \text{Var}(X_{t-1,l}^{(j)}) + \eta_{j,l}^2$. Its entropy is $H(\tilde{X}_{t-1}^{(j)(\eta_j)}) = \frac{KM}{2} \log(2\pi e) + \sum_{l=1}^{KM} \frac{1}{2} \log(\eta_{j,l}^2 + \text{Var}(X_{t-1,l}^{(j)}))$. Therefore,

$$\begin{aligned}I(\tilde{X}_{t-1}^{(j)(\eta_j)}; X_{t-1}^{(j)}) &\leq \left(\frac{KM}{2} \log(2\pi e) + \sum_{l=1}^{KM} \frac{1}{2} \log(\eta_{j,l}^2 + \text{Var}(X_{t-1,l}^{(j)})) \right) - \left(\frac{KM}{2} \log(2\pi e) + \sum_{l=1}^{KM} \frac{1}{2} \log(\eta_{j,l}^2) \right) \\ &= \frac{1}{2} \sum_{l=1}^{KM} \log \left(1 + \frac{\text{Var}(X_{t-1,l}^{(j)})}{\eta_{j,l}^2} \right)\end{aligned}$$

The equality is reached when $X_{t-1}^{(j)}$ obeys a multivariate Gaussian distribution with diagonal covariance matrix Σ satisfying $\Sigma_{l,l} = \text{Var}(X_{t-1,l}^{(j)}) + \eta_{j,l}^2$. ■

A.5.4 Implementation details for the methods

Here we state the implementation details for our method, as well as other methods being compared. Throughout this paper, unless otherwise specified, we use the standard k-nearest neighbor technique in [KSG04] to estimate the KL-divergence and mutual information (with number of neighbors $k = 5$) and conditional mutual information (with number of neighbors $k = 3$), which is used in our implementations of Mutual information, Transfer Entropy and Causal Influence.

Our method

Without stating otherwise, our method (Algorithm 3) as a default uses a three layer neural net, with two hidden layers having 8 neurons and leakyReLU ($\max(0.3x, x)$) activation, and the last layer having linear activation. We set the number of fake time series $S = \max(2, \lceil N/2 \rceil)$, and significance level $\alpha = 0.05$. Adam [KB14] optimizer with learning rate $= 10^{-4}$ is used as default throughout this paper. We set $\eta_0 = 0.01$ and $\lambda = 0.002$. We use 30000 epochs. It also has a 400 epoch warm-up period where the mutual information term is turned off, to allow f_θ to find a good initial model as a start. We use the upper bound (Eq. 6.4) as the risk and also in estimating W_{ji} , as discussed in the main text in Section 6.2.2. In this work, the relative noise amplitude $\chi_{j,l} = \frac{\eta_{j,l}}{\text{std}(X_{t-1,l}^{(j)})}$ is shared across the dimension l for each time series j . This simplifies the risk calculation, and is invariant to the rescaling of each time series $X_{t-1}^{(j)}$. We also tested fully parameterizing $\chi_{j,l}$ with a similar performance.

Transfer Entropy

We use the definition of transfer entropy as defined in [Sch00]. In that work the transfer entropy is defined for two time series. To deal with multiple time series, we let $X_{t-1}^{(j)}$ also include other time series, similar to the extension of transfer entropy as in [LPZ08].

Causal Influence

For causal influence [JBGW⁺13], we use the same network architecture as in our method, to learn a prediction model. Then the KL divergence is estimated via the technique in [KSG04].

Linear Granger

We follow the definition of linear Granger causality (Eq. (7) and (8) in [DCB06]) to calculate linear Granger causality. Specifically, we calculate the residual squared error of a linear predictor of $x_{t-1}^{(i)}$ with and without $X_{t-1}^{(j)}$ (both with $\mathbf{X}_{t-1}^{(j)}$). Then the linear Granger causality equals the log of the ratio of the two residual squared errors.

Kernel Granger

We use the implementation⁹ for [MPS08b, MPS08a] for estimating kernel Granger causality. We use their default settings, with inhomogeneous polynomial (IP) kernel of degree $p = 2$. We follow the normalization requirement of the algorithm to normalize the data for each experiment.

Elastic Net

We use elastic net [ZH05] with 5-fold time-series-split cross-validation, along the following regularization path: L1-ratio: 0.5, 0.8, 0.9, 0.95, 0.99, and strength of penalization α being a 200-step geometric series from 10^{-4} to $10^{-0.5}$. The score function used for cross-validation is the coefficient of determination (R^2). The elastic net is implemented with scikit-learn's ElasticNetCV module¹⁰, with optimization tolerance of 10^{-10} .

Gaussian Random

For Gaussian Random, we draw 10,000 random matrices, each element of which is drawn from a standard Gaussian distribution.

A.5.5 Implementation details for synthetic experiments

For all experiments in this section, each metric is obtained by performing the experiments (including generation of the dataset and the training) ten times with seed = 0, 30, 60, 90, 120, 150, 180, 210, 240, 270 and averaging the resulting metrics (for Gaussian random matrices, for each true causal matrix A sample 10,000 random matrices \tilde{A}). For the ground-truth causal tensor A , each element A_{ji} is a $K \times M$ matrix, with 0.5 probability of being an all-zero matrix, and 0.5 probability of being a nonzero matrix. If A_{ji} is a nonzero matrix, its each element is sampled from a log-normal distribution with $\mu = 0$ and $\sigma = 1$. For B , each B_j is also a $K \times N$ matrix, with each element sampling from $U[-1, 1]$. We use

⁹At <https://github.com/danielemarinazzo/KernelGrangerCausality>.

¹⁰At https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNetCV.html.

Table S A.1: Mean and standard deviation of AUC-ROC (%) vs. N , over 10 random sampling of datasets. Bold font marks the top method for each N .

N method	3	4	5	8	10	15	20	30
MPIR (ours)	$95.3_{\pm 10.0}$	$97.6_{\pm 4.1}$	$97.3_{\pm 3.6}$	$96.0_{\pm 2.4}$	$94.2_{\pm 3.8}$	$91.0_{\pm 3.5}$	$85.5_{\pm 2.4}$	$76.8_{\pm 3.5}$
Mutual Information	$84.1_{\pm 18.9}$	$90.0_{\pm 7.6}$	$89.0_{\pm 1.8}$	$87.2_{\pm 3.8}$	$81.3_{\pm 5.3}$	$77.5_{\pm 3.9}$	$74.6_{\pm 3.0}$	$72.0_{\pm 2.0}$
Transfer Entropy	$88.3_{\pm 14.6}$	$95.6_{\pm 5.7}$	$89.9_{\pm 8.7}$	$84.4_{\pm 7.6}$	$80.8_{\pm 5.1}$	$69.6_{\pm 2.5}$	$64.7_{\pm 2.5}$	$59.2_{\pm 1.9}$
Linear Granger	$98.8_{\pm 4.0}$	$96.2_{\pm 5.5}$	$91.7_{\pm 8.9}$	$84.1_{\pm 9.0}$	$82.7_{\pm 7.2}$	$73.6_{\pm 6.9}$	$69.9_{\pm 4.1}$	$60.0_{\pm 2.6}$
Kernel Granger	$98.1_{\pm 5.9}$	$98.0_{\pm 4.4}$	$95.4_{\pm 3.9}$	$91.2_{\pm 2.6}$	$89.5_{\pm 3.3}$	$82.4_{\pm 2.2}$	$76.2_{\pm 2.2}$	$68.1_{\pm 1.3}$
Elastic Net	$97.5_{\pm 7.9}$	$97.4_{\pm 4.5}$	$95.3_{\pm 4.3}$	$90.4_{\pm 5.1}$	$87.7_{\pm 4.1}$	$81.8_{\pm 3.1}$	$77.8_{\pm 3.0}$	$72.7_{\pm 1.4}$
Causal Influence	$62.9_{\pm 28.3}$	$58.3_{\pm 13.8}$	$60.4_{\pm 11.7}$	$47.4_{\pm 7.5}$	$50.7_{\pm 5.6}$	$55.3_{\pm 3.3}$	$51.0_{\pm 3.2}$	$50.3_{\pm 1.6}$
Gaussian random	$49.9_{\pm 0.3}$	$50.0_{\pm 0.1}$	$50.0_{\pm 0.1}$	$50.0_{\pm 0.0}$	$50.0_{\pm 0.1}$	$50.0_{\pm 0.0}$	$50.0_{\pm 0.0}$	$50.0_{\pm 0.0}$

$H_1(x) = \text{softplus}(x) = \log(1 + e^x)$, and $H_2(x) = \tanh(x)$ in equation (6.5). As a default, 500 time series each with length of 22 are generated from Eq. (6.5), each of which is wrapped into 19 ($\mathbf{X}_{t-1}, x_t^{(i)}$) pairs (since $K = 3$), so there are in total $500 \times 19 = 9500$ examples for each dataset. The train-test-split is 9:1 for all experiments in this paper. See Fig. SA.4 for example snapshots of time series together with the corresponding A_{ji} matrices.

A.5.6 AUC-ROC table for synthetic experiment

Table S6.1 show the AUC-ROC table for the synthetic experiment, where for each N , 10 datasets are randomly sampled according to Eq. (6.5) using random seed 0, 30, 60, 90, 120, 150, 180, 210, 240, 270, over which each method is run and their metrics are accumulated. It has similar behavior as the AUC-PR table (Table 6.1) in the main text.

A.5.7 Additional experiment: testing with model capacity variations

Since in practice, we do not know the underlying causal structure *a priori*, it presents a greater challenge to select the model capacity for f_θ , as compared with supervised learning method where we can do cross-validation. To see how the capacity of the function approximator f_θ influences our method, we vary the number of layers and the number of neurons in each layer at $N = 10$, using the same 10 datasets as in Section 6.3.1. Table SA.2 summarizes

Table S A.2: Average and standard deviation of AUC-PR and AUC-ROC for different network structures for $N = 10$ with our method. Here for example, $(8, 8, 8)$ means that the f_θ has 3 hidden layers, each with 8 neurons.

Neurons in hidden layers	AUC-PR (%)	AUC-ROC (%)
(8)	90.0 ± 4.9	91.5 ± 4.3
(8, 8)	93.4 ± 3.6	94.1 ± 3.7
(8, 8, 8)	93.6 ± 3.6	94.4 ± 3.6
(8, 8, 8, 8)	93.8 ± 4.1	94.2 ± 4.3
(16, 16)	94.3 ± 3.3	94.4 ± 3.5
(16, 16, 16)	94.6 ± 3.0	95.1 ± 2.6
(16, 16, 16, 16)	92.8 ± 4.4	94.0 ± 3.2

the result. We see that our method’s performance here is hardly influenced by the model capacity, with only a slight degradation at very low capacity. This shows that our method is quite tolerant and stable with model capacity variations.

A.5.8 Details for the video game dataset

Here, we implement a custom Atari Breakout game in the OpenAI Gym [BCP⁺16] environment, mimicking the original game¹¹, where we can access the state of the ball, paddle and bricks, etc. This representation is also used in the OO-MDP [DCL08] paradigm for a more efficient representation of the environment state. We use the DQN algorithm, the same CNN architecture as in [MKS⁺15] to train an RL agent. Then we let it play the game for ~ 45000 steps, obtaining a dataset with time-length of 45000 steps (if the agent dies, we restart the game) and 6 time series: action, paddle’s x position, ball’s x position, ball’s y position, number of bricks and reward. We then feed the time series (each time series normalized to mean of 0 and variance of 1) to our method, the same procedure as performed in the synthetic experiment, to let it produce an inferred matrix W_{ji} , which is shown in Fig. 1 in main text. All the datasets used in this paper and code will be open-sourced upon publication of the paper.

¹¹A game playing video can be seen at <https://goo.gl/XGzppc>.

A.5.9 Implementation details for experiment with heart-rate vs. breath-rate

For the two real-world datasets, we obtain the data with the same procedure as in [AMS04] (See Fig.SA.5 for their plots). Then the data (each time series normalized to mean of 0 and variance of 1) are fed into our algorithm to infer the causal strength W_{ji} . For each $K = 1, 2, \dots, 20$, the experiments are run for 50 times with seed from 0 to 49, and Fig. 6-2 in the main text is obtained by averaging over the inferred W matrix.

A.5.10 Additional experiment: rat EEG dataset

As another real-world example, we apply our algorithm to estimate the directional relations of the EEG signals between the right and left cortical intracranial electrodes [Qui], before and after lesion (see Fig. SA.6 and SA.7 for the signals), also studied in [AMS04, QKG02, MPS08b]. Figure SA.8 (left) shows the inferred predictive strength W_{ji} for the EEG signals of a normal rat. We see that there is only a slight asymmetry, with the right channel having a slightly stronger influence on the left channel than the reverse direction. Fig. SA.8 (right) shows W_{ji} for the EEG signals with unilateral lesion in the rostral pole of the reticular thalamic nucleus. We see that there is stronger predictive strength from the left to the right channels. Compared with the result of previous works [AMS04, MPS08b] as also shown in Fig. SA.9, we see that all methods correctly infer the directional relations before and after brain lesion. In addition, our method shows only a slight decay of predictive strength with increasing history length, in contrast to the much more rapid decay of causality index in [AMS04], again demonstrating our method's insensitivity against history length, due to its flexibility in extracting the right amount of information in order to predict the future. This experiment and the breadth rate vs. heart rate experiment in Section 6.3.3 demonstrate our method's capability in inferring the directional relations from noisy, real-world data.

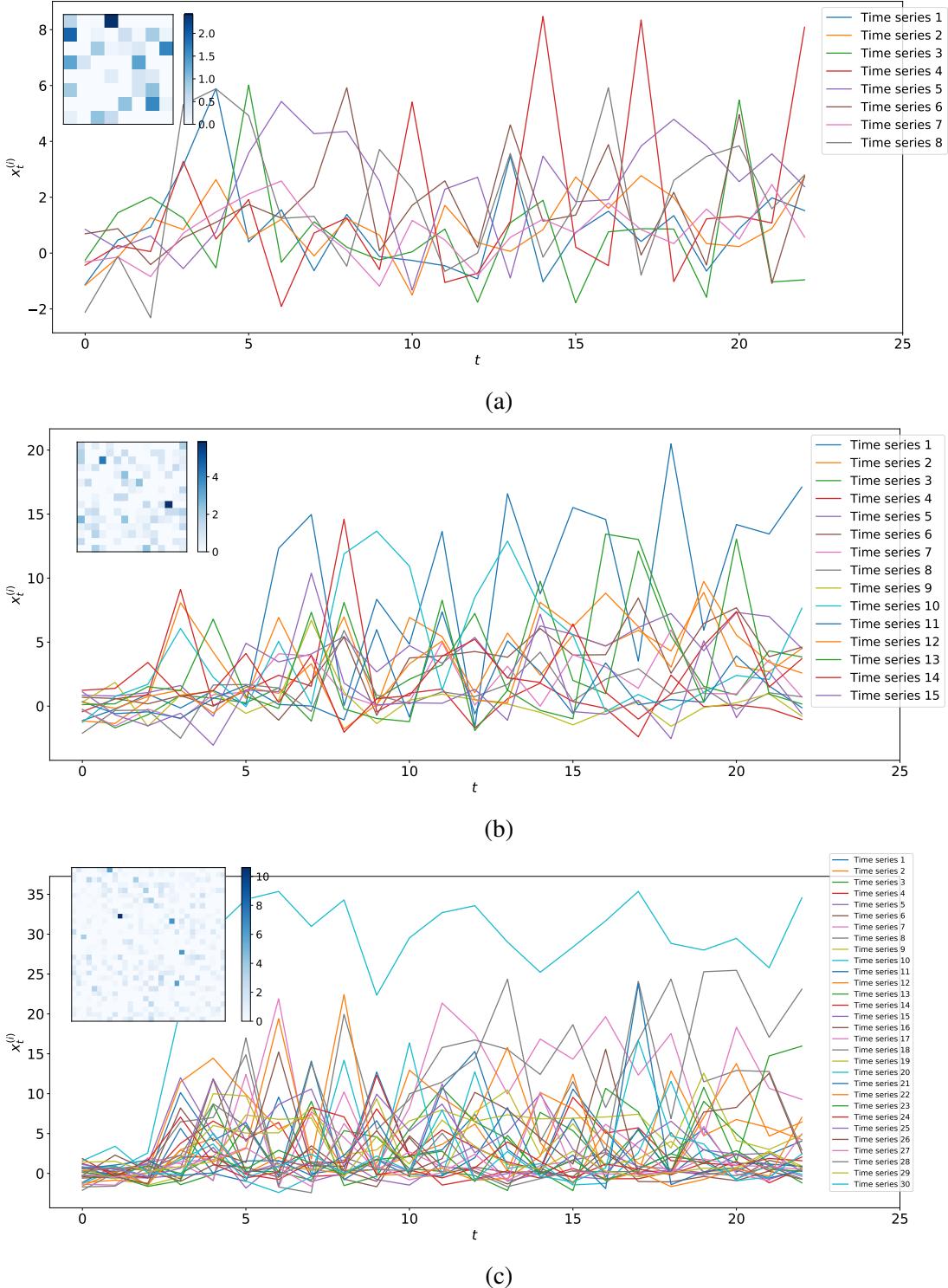


Figure S A.4: Example snapshots of the synthetic time series with (a) $N = 8$, (b) $N = 15$, and (c) $N = 30$. The inset is the hidden underlying $|A_{ji}|$ matrix, whose (j, i) element denotes the causal strength from time series j to i . We see that the causal strength varies in orders, making it very difficult to identify each edge correctly.

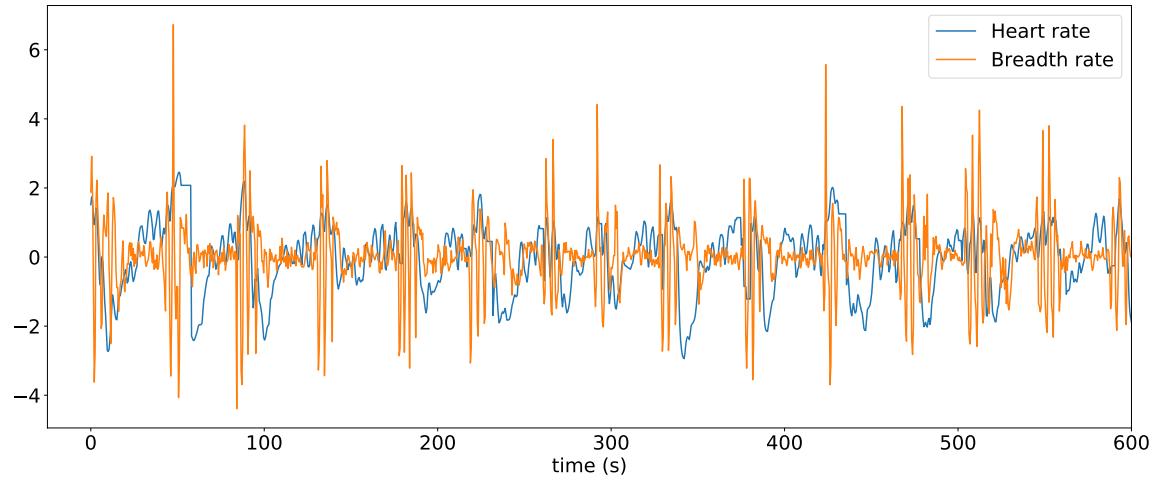


Figure S A.5: Time series of the heart rate and breath rate of a patient suffering sleep apnea. The data is normalized to have 0 mean and standard deviation of 1. Sample rate is 2Hz.

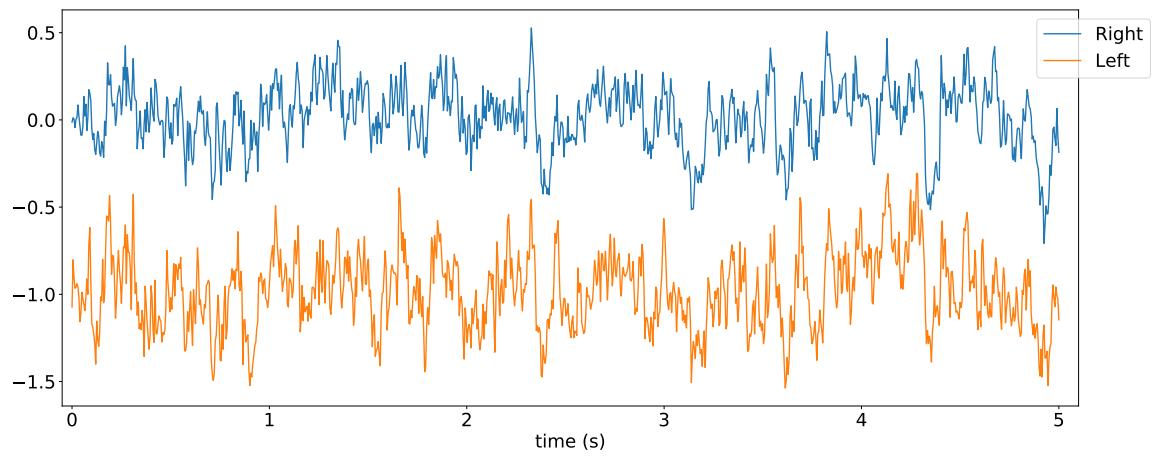


Figure S A.6: Time series of a normal rat EEG signals from right and left cortical intracranial electrodes. The data is normalized to have 0 mean and standard deviation of 1, and the left signal is plotted with offset for better visualization. Sample rate is 200Hz.

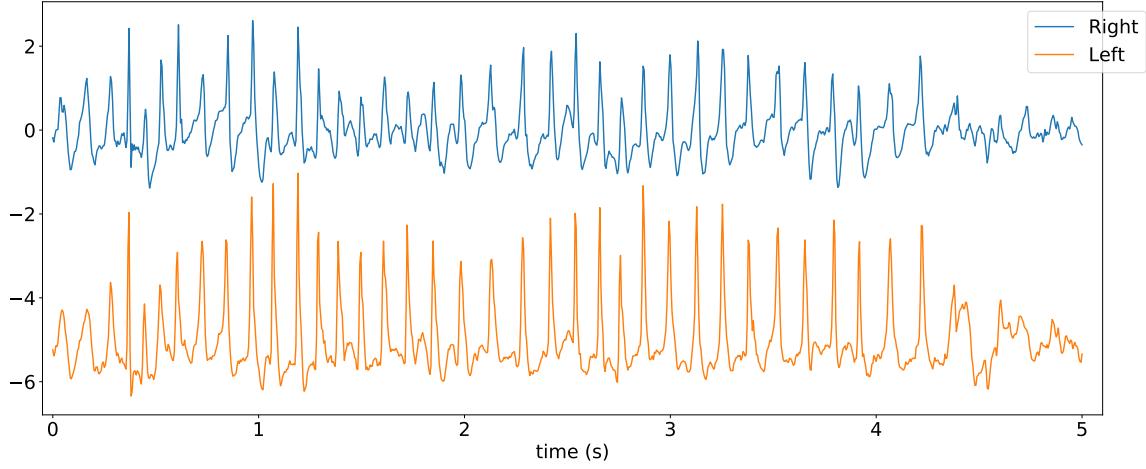


Figure S A.7: Time series of a rat EEG signals from right and left cortical intracranial electrodes, after lesion. The data is normalized to have 0 mean and standard deviation of 1, and the left signal is plotted with offset for better visualization. Sample rate is 200Hz.

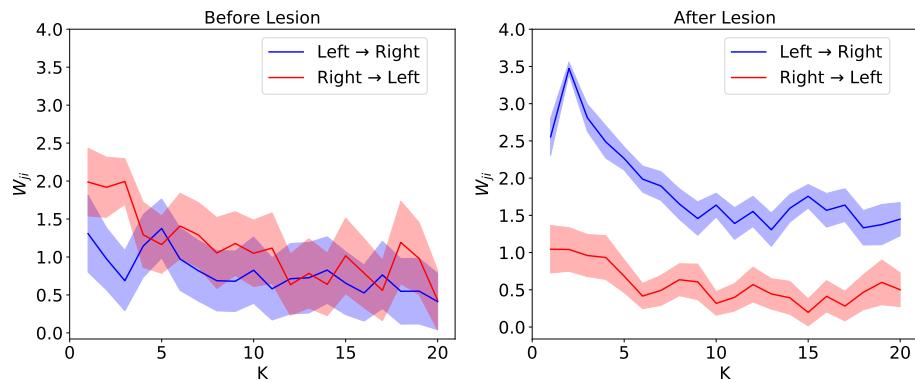


Figure S A.8: Predictive strength inferred by our method with the EEG datasets, for different maximum time horizon K , averaged over 50 initializations of f_θ , for a normal rat (left) and after brain lesion (right).

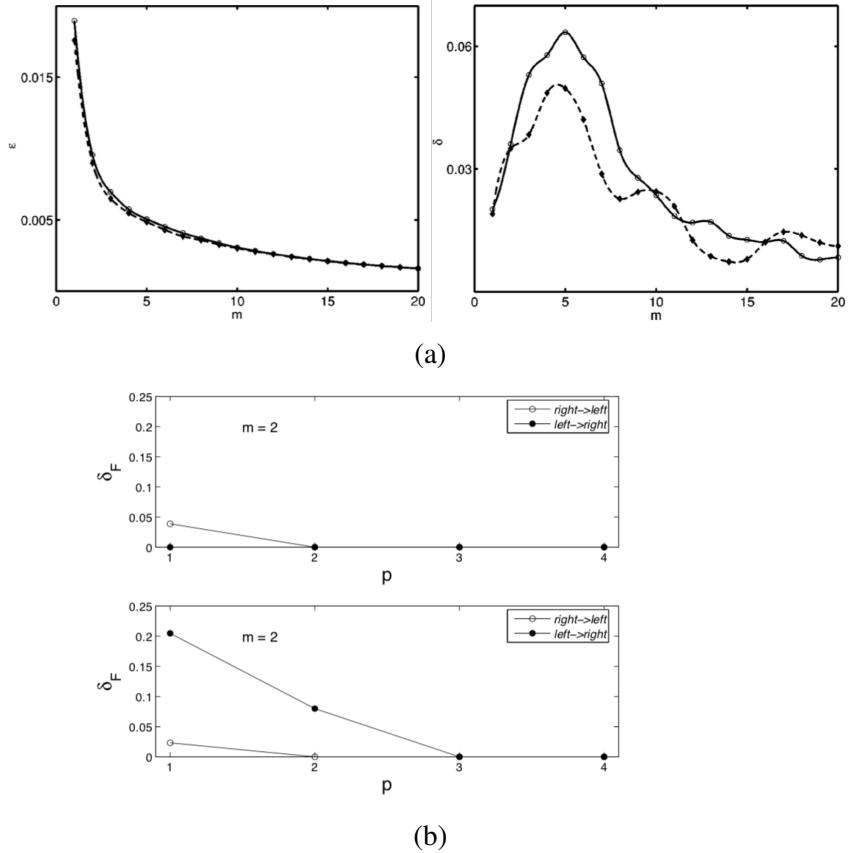


Figure S A.9: Causal indices for the rat EEG dataset with previous methods. (a) By [AMS04]. Left: the variance for the left EEG (open circles) and right EEG (diamonds) vs. time lag m before brain lesion. Right: the causality index after brain lesion. (b) By [MPS08a]. The filtered causality index vs. varying p , the order of the inhomogeneous polynomial kernel, before (upper) and after (lower) brain lesion.

A.6 Appendix for Chapter 7

MeLA architectural details

As described in section 7.2.2, MeLA consists of a meta-recognition model \mathbf{m}_μ and a meta-generative model \mathbf{g}_γ that generates the task-specific model \mathbf{f}_θ . The meta-recognition model consists of two blocks. The first block is a MLP with 3 hidden layers, each of which has 60 neurons with leakyReLU activation (unless otherwise specified, the leakyReLU activation in this paper all have a slope of 0.3 when the activation is below 0). The last layer has $s_{\text{pool}} = 200$ neurons and linear activation. Then a max-pooling is performed along the example dimension, collapsing the $N \times s_{\text{pool}}$ matrix to $1 \times s_{\text{pool}}$ matrix, which feeds into the second block. The second block is an MLP with two hidden layers, each of which has 60 neurons with leakyReLU activation, and the last layer has s_{code} neurons with linear activation. The output is the model code \mathbf{z} .

The meta-generative model \mathbf{g}_γ takes as input the model code \mathbf{z} , and for each layer in the main model \mathbf{f}_θ , it has two separate MLPs that map \mathbf{z} to all the weight and bias parameters of that layer. For all the experiments in this paper, the MLPs in the meta-generative model have 3 hidden layers, each of which has 60 neurons with leakyReLU activation. The last layer of the MLP has linear activation, and has an output size equal to the size of weight or bias in the main network \mathbf{f}_θ . The output of each MLP in the meta-generative model is then reshaped into the size of the corresponding weight or bias matrix, and directly used as the parameters of \mathbf{f}_θ .

The architecture of the main network \mathbf{f}_θ is dependent on the specific application, which MeLA’s architecture is agnostic to. For the simple regression problem in this paper, we implement \mathbf{f}_θ as an MLP with 2 hidden layers, each of which has 40 neurons with leakyReLU activation. The last layer has linear activation with output size of 1. For the ball bouncing with state representation experiment, \mathbf{f}_θ is an MLP with input size of 6 and 3 hidden layers, each of which has 40 neurons with leakyReLU activation. The last layer has linear activation with output size of 2. For the video prediction task, the latent dynamics network uses the same architecture. The convolutional autoencoder used in this experiment is as follows. For the encoder, it has 3 convolutional layers with $32 \times 3 \times 3$ kernels with stride 2 and leakyReLU

activation. After that, it is flattened into 512 neurons, which feeds into a dense layer with 2 neurons and linear activation. For the decoder, the first layer is a dense layer with 512 neurons and linear activation, then the output is reshaped to a $N \times 4 \times 4 \times 32$ tensor (32 is the number of channels). The tensor then goes into 3 layers of convolutional-transpose layers with 32 kernels, each with size of 3, stride of 2 and leakyReLU activation. For the leakyReLU activation in the convolutional autoencoder, we use a slope of 0.01 when the activation is below 0.

A.7 Appendix for Chapter 8

A.7.1 Proofs

In this section, we provide proofs for all the lemmas and theorems in the main paper. We always assume that a class-conditional extension of the Classification Noise Process (CNP) [AL88] maps true labels y to observed labels s such that each label in P is flipped independently with probability ρ_1 and each label in N is flipped independently with probability ρ_0 ($s \leftarrow CNP(y, \rho_1, \rho_0)$), so that $P(s = s|y = y, x) = P(s = s|y = y)$. Remember that $\rho_1 + \rho_0 < 1$ is a necessary condition of minimal information, otherwise we may learn opposite labels.

In Lemma 1, Theorem 2, Lemma 3 and Theorem 4, we assume that P and N have infinite number of examples so that they are the true, hidden distributions.

A fundamental equation we use in the proofs is the following lemma:

Lemma A1 *When g is ideal, i.e. $g(x) = g^*(x)$ and P and N have non-overlapping support, we have*

$$g(x) = (1 - \rho_1) \cdot \mathbb{1}[[y = 1]] + \rho_0 \cdot \mathbb{1}[[y = 0]] \quad (\text{A.69})$$

Proof: Since $g(x) = g^*(x)$ and P and N have non-overlapping support, we have

$$\begin{aligned} g(x) &= g^*(x) = P(s = 1|x) \\ &= P(s = 1|y = 1, x) \cdot P(y = 1|x) + P(s = 1|y = 0, x) \cdot P(y = 0|x) \\ &= P(s = 1|y = 1) \cdot P(y = 1|x) + P(s = 1|y = 0) \cdot P(y = 0|x) \\ &= (1 - \rho_1) \cdot \mathbb{1}[[y = 1]] + \rho_0 \cdot \mathbb{1}[[y = 0]] \end{aligned}$$

Proof of Lemma 1

Lemma 1 When g is ideal, i.e. $g(x) = g^*(x)$ and P and N have non-overlapping support, we have

$$\begin{cases} \tilde{P}_{y=1} = \{x \in P | s = 1\}, \tilde{N}_{y=1} = \{x \in P | s = 0\} \\ \tilde{P}_{y=0} = \{x \in N | s = 1\}, \tilde{N}_{y=0} = \{x \in N | s = 0\} \end{cases} \quad (\text{A.70})$$

Proof: Firstly, we compute the threshold $LB_{y=1}$ and $UB_{y=0}$ used by $\tilde{P}_{y=1}$, $\tilde{N}_{y=1}$, $\tilde{P}_{y=0}$ and $\tilde{N}_{y=0}$. Since P and N have non-overlapping support, we have $P(y = 1|x) = \mathbb{1}[[y = 1]]$. Also using $g(x) = g^*(x)$, we have

$$\begin{aligned} LB_{y=1} &= E_{x \in \tilde{P}}[g(x)] = E_{x \in \tilde{P}}[P(s = 1|x)] \\ &= E_{x \in \tilde{P}}[P(s = 1|x, y = 1)P(y = 1|x) + P(s = 1|x, y = 0)P(y = 0|x)] \\ &= E_{x \in \tilde{P}}[P(s = 1|y = 1)P(y = 1|x) + P(s = 1|y = 0)P(y = 0|x)] \\ &= (1 - \rho_1)(1 - \pi_1) + \rho_0\pi_1 \end{aligned} \quad (\text{A.71})$$

Similarly, we have

$$UB_{y=0} = (1 - \rho_1)\pi_0 + \rho_0(1 - \pi_0)$$

Since $\pi_1 = P(y = 0|s = 1)$, we have $\pi_1 \in [0, 1]$. Furthermore, we have the requirement that $\rho_1 + \rho_0 < 1$, then $\pi_1 = 1$ will lead to $\rho_1 = P(s = 0|y = 1) = 1 - P(s = 1|y = 1) = 1 - \frac{P(y=1|s=1)P(s=1)}{P(y=1)} = 1 - 0 = 1$ which violates the requirement of $\rho_1 + \rho_0 < 1$. Therefore, $\pi_1 \in [0, 1)$. Similarly, we can prove $\pi_0 \in [0, 1)$. Therefore, we see that both $LB_{y=1}$ and $UB_{y=0}$ are interpolations of $(1 - \rho_1)$ and ρ_0 :

$$\begin{aligned} \rho_0 &< LB_{y=1} \leq 1 - \rho_1 \\ \rho_0 &\leq UB_{y=0} < 1 - \rho_1 \end{aligned}$$

The first equality holds iff $\pi_1 = 0$ and the second equality holds iff $\pi_0 = 0$.

Using Lemma A1, we know that under the condition of $g(x) = g^*(x)$ and non-overlapping

support, $g(x) = (1 - \rho_1) \cdot \mathbb{1}[[y = 1]] + \rho_0 \cdot \mathbb{1}[[y = 0]]$. In other words,

$$g(x) \geq LB_{y=1} \Leftrightarrow x \in P$$

$$g(x) \leq UB_{y=0} \Leftrightarrow x \in N$$

Since

$$\begin{cases} \tilde{P}_{y=1} = \{x \in \tilde{P} | g(x) \geq LB_{y=1}\} \\ \tilde{N}_{y=1} = \{x \in \tilde{N} | g(x) \geq LB_{y=1}\} \\ \tilde{P}_{y=0} = \{x \in \tilde{P} | g(x) \leq UB_{y=0}\} \\ \tilde{N}_{y=0} = \{x \in \tilde{N} | g(x) \leq UB_{y=0}\} \end{cases}$$

where $\tilde{P} = \{x | s = 1\}$ and $\tilde{N} = \{x | s = 0\}$, we have

$$\begin{cases} \tilde{P}_{y=1} = \{x \in P | s = 1\}, \tilde{N}_{y=1} = \{x \in P | s = 0\} \\ \tilde{P}_{y=0} = \{x \in N | s = 1\}, \tilde{N}_{y=0} = \{x \in N | s = 0\} \end{cases}$$

Proof of Theorem 2

We restate Theorem 2 here:

Theorem 2 When g is ideal, i.e. $g(x) = g^*(x)$ and P and N have non-overlapping support, we have

$$\hat{\rho}_1^{conf} = \rho_1, \hat{\rho}_0^{conf} = \rho_0$$

Proof: Using the definition of $\hat{\rho}_1^{conf}$ in the main paper:

$$\hat{\rho}_1^{conf} = \frac{|\tilde{N}_{y=1}|}{|\tilde{N}_{y=1}| + |\tilde{P}_{y=1}|}, \hat{\rho}_0^{conf} = \frac{|\tilde{P}_{y=0}|}{|\tilde{P}_{y=0}| + |\tilde{N}_{y=0}|}$$

Since $g(x) = g^*(x)$ and P and N have non-overlapping support, using Lemma 1, we know

$$\begin{cases} \tilde{P}_{y=1} = \{x \in P | s = 1\}, \tilde{N}_{y=1} = \{x \in P | s = 0\} \\ \tilde{P}_{y=0} = \{x \in N | s = 1\}, \tilde{N}_{y=0} = \{x \in N | s = 0\} \end{cases}$$

Since $\rho_1 = P(s = 0 | y = 1)$ and $\rho_0 = P(s = 1 | y = 0)$, we immediately have

$$\hat{\rho}_1^{conf} = \frac{|\{x \in P | s = 0\}|}{|P|} = \rho_1, \quad \hat{\rho}_0^{conf} = \frac{|\{x \in N | s = 1\}|}{|N|} = \rho_0$$

Proof of Lemma 3

We rewrite Lemma 3 below:

Lemma 3 *When g is unasssuming, i.e., $\Delta g(x) := g(x) - g^*(x)$ can be nonzero, and P and N can have overlapping support, we have*

$$\begin{cases} LB_{y=1} = LB_{y=1}^* + E_{x \in \tilde{P}}[\Delta g(x)] - \frac{(1-\rho_1-\rho_0)^2}{p_{s1}} \Delta p_o \\ UB_{y=0} = UB_{y=0}^* + E_{x \in \tilde{N}}[\Delta g(x)] + \frac{(1-\rho_1-\rho_0)^2}{1-p_{s1}} \Delta p_o \\ \hat{\rho}_1^{conf} = \rho_1 + \frac{1-\rho_1-\rho_0}{|P|-|\Delta P_1|+|\Delta N_1|} |\Delta N_1| \\ \hat{\rho}_0^{conf} = \rho_0 + \frac{1-\rho_1-\rho_0}{|N|-|\Delta N_0|+|\Delta P_0|} |\Delta P_0| \end{cases} \quad (\text{A.72})$$

where

$$\begin{cases} LB_{y=1}^* = (1 - \rho_1)(1 - \pi_1) + \rho_0 \pi_1 \\ UB_{y=0}^* = (1 - \rho_1)\pi_0 + \rho_0(1 - \pi_0) \\ \Delta p_o := \frac{|P \cap N|}{|P \cup N|} \\ \Delta P_1 = \{x \in P | g(x) < LB_{y=1}\} \\ \Delta N_1 = \{x \in N | g(x) \geq LB_{y=1}\} \\ \Delta P_0 = \{x \in P | g(x) \leq UB_{y=0}\} \\ \Delta N_0 = \{x \in N | g(x) > UB_{y=0}\} \end{cases} \quad (\text{A.73})$$

Proof: We first calculate $LB_{y=1}$ and $UB_{y=0}$ under unasssuming conditions, then calculate

$\hat{\rho}_i^{conf}$, $i = 0, 1$ under unasssuming condition.

Note that Δp_o can also be expressed as

$$\Delta p_o := \frac{|P \cap N|}{|P \cup N|} = P(\hat{y} = 1, y = 0) = P(\hat{y} = 0, y = 1)$$

Here $P(\hat{y} = 1, y = 0) \equiv P(\hat{y} = 1|y = 0)P(y = 0)$, where $P(\hat{y} = 1|y = 0)$ means for a perfect classifier $f^*(x) = P(y = 1|x)$, the expected probability that it will label a $y = 0$ example as positive ($\hat{y} = 1$).

(1) $LB_{y=1}$ and $UB_{y=0}$ under unasssuming condition

Firstly, we calculate $LB_{y=1}$ and $UB_{y=0}$ with perfect probability estimation $g^*(x)$, but the support may overlap. Secondly, we allow the probability estimation to be imperfect, superimposed onto the overlapping support condition, and calculate $LB_{y=1}$ and $UB_{y=0}$.

I. Calculating $LB_{y=1}$ and $UB_{y=0}$ when $g(x) = g^*(x)$ and support may overlap

With overlapping support, we no longer have $P(y = 1|x) = \mathbb{1}[[y = 1]]$. Instead, we have

$$\begin{aligned} LB_{y=1} &= E_{x \in \tilde{P}}[g^*(x)] = E_{x \in \tilde{P}}[P(s = 1|x)] \\ &= E_{x \in \tilde{P}}[P(s = 1|x, y = 1)P(y = 1|x) + P(s = 1|x, y = 0)P(y = 0|x)] \\ &= E_{x \in \tilde{P}}[P(s = 1|y = 1)P(y = 1|x) + P(s = 1|y = 0)P(y = 0|x)] \\ &= (1 - \rho_1) \cdot E_{x \in \tilde{P}}[P(y = 1|x)] + \rho_0 \cdot E_{x \in \tilde{P}}[P(y = 0|x)] \\ &= (1 - \rho_1) \cdot P(\hat{y} = 1|s = 1) + \rho_0 \cdot P(\hat{y} = 0|s = 1) \end{aligned}$$

Here $P(\hat{y} = 1|s = 1)$ can be calculated using Δp_o :

$$\begin{aligned}
P(\hat{y} = 1|s = 1) &= \frac{P(\hat{y} = 1, s = 1)}{P(s = 1)} \\
&= \frac{P(\hat{y} = 1, y = 1, s = 1) + P(\hat{y} = 1, y = 0, s = 1)}{P(s = 1)} \\
&= \frac{P(s = 1|y = 1)P(\hat{y} = 1, y = 1) + P(s = 1|y = 0)P(\hat{y} = 1, y = 0)}{P(s = 1)} \\
&= \frac{(1 - \rho_1)(p_{y1} - \Delta p_o) + \rho_0 \Delta p_o}{p_{s1}} \\
&= (1 - \pi_1) - \frac{1 - \rho_1 - \rho_0}{p_{s1}} \Delta p_o
\end{aligned}$$

Hence,

$$P(\hat{y} = 0|s = 1) = 1 - P(\hat{y} = 1|s = 1) = \pi_1 + \frac{1 - \rho_1 - \rho_0}{p_{s1}} \Delta p_o$$

Therefore,

$$\begin{aligned}
LB_{y=1} &= (1 - \rho_1) \cdot P(\hat{y} = 1|s = 1) + \rho_0 \cdot P(\hat{y} = 0|s = 1) \\
&= (1 - \rho_1) \cdot \left((1 - \pi_1) - \frac{1 - \rho_1 - \rho_0}{p_{s1}} \Delta p_o \right) + \rho_0 \cdot \left(\pi_1 + \frac{1 - \rho_1 - \rho_0}{p_{s1}} \Delta p_o \right) \\
&= LB_{y=1}^* - \frac{(1 - \rho_1 - \rho_0)^2}{p_{s1}} \Delta p_o
\end{aligned} \tag{A.74}$$

where $LB_{y=1}^*$ is the $LB_{y=1}$ value when $g(x)$ is ideal. We see in Eq. (A.74) that the overlapping support introduces a non-positive correction to $LB_{y=1}^*$ compared with the ideal condition.

Similarly, we have

$$UB_{y=0} = UB_{y=0}^* + \frac{(1 - \rho_1 - \rho_0)^2}{1 - p_{s1}} \Delta p_o \tag{A.75}$$

II. Calculating $LB_{y=1}$ and $UB_{y=0}$ when g is unasssuming

Define $\Delta g(x) = g(x) - g^*(x)$. When the support may overlap, we have

$$\begin{aligned}
LB_{y=1} &= E_{x \in \tilde{P}}[g(x)] \\
&= E_{x \in \tilde{P}}[g^*(x)] + E_{x \in \tilde{P}}[\Delta g(x)] \\
&= LB_{y=1}^* - \frac{(1 - \rho_1 - \rho_0)^2}{p_{s1}} \Delta p_o + E_{x \in \tilde{P}}[\Delta g(x)]
\end{aligned} \tag{A.76}$$

Similarly, we have

$$\begin{aligned}
UB_{y=0} &= E_{x \in \tilde{N}}[g(x)] \\
&= E_{x \in \tilde{N}}[g^*(x)] + E_{x \in \tilde{N}}[\Delta g(x)] \\
&= UB_{y=0}^* + \frac{(1 - \rho_1 - \rho_0)^2}{1 - p_{s1}} \Delta p_o + E_{x \in \tilde{N}}[\Delta g(x)]
\end{aligned} \tag{A.77}$$

In summary, Eq. (A.76) (A.77) give the expressions for $LB_{y=1}$ and $UB_{y=0}$, respectively, when g is unasssuming.

(2) $\hat{\rho}_i^{conf}$ under unasssuming condition

Now let's calculate $\hat{\rho}_i^{conf}$, $i = 0, 1$. For simplicity, define

$$\left\{
\begin{array}{l}
PP = \{x \in P | s = 1\} \\
PN = \{x \in P | s = 0\} \\
NP = \{x \in N | s = 1\} \\
NN = \{x \in N | s = 0\} \\
\Delta_{PP_1} = \{x \in PP | g(x) < LB_{y=1}\} \\
\Delta_{NP_1} = \{x \in NP | g(x) \geq LB_{y=1}\} \\
\Delta_{PN_1} = \{x \in PN | g(x) < LB_{y=1}\} \\
\Delta_{NN_1} = \{x \in NN | g(x) \geq LB_{y=1}\}
\end{array}
\right. \tag{A.78}$$

For $\hat{\rho}_1^{conf}$, we have:

$$\hat{\rho}_1^{conf} = \frac{|\tilde{N}_{y=1}|}{|\tilde{P}_{y=1}| + |\tilde{N}_{y=1}|}$$

Here

$$\begin{aligned}\tilde{P}_{y=1} &= \{x \in \tilde{P} | g(x) \geq LB_{y=1}\} \\ &= \{x \in PP | g(x) \geq LB_{y=1}\} \cup \{x \in NP | g(x) \geq LB_{y=1}\} \\ &= (PP \setminus \Delta_{PP_1}) \cup \Delta_{NP_1}\end{aligned}$$

Similarly, we have

$$\tilde{N}_{y=1} = (PN \setminus \Delta_{PN_1}) \cup \Delta_{NN_1}$$

Therefore

$$\begin{aligned}\hat{\rho}_1^{conf} &= \frac{|PN| - |\Delta_{PN_1}| + |\Delta_{NN_1}|}{[|PP| - |\Delta_{PP_1}|] + [|PN| - |\Delta_{PN_1}|] + (|\Delta_{NN_1}| + |\Delta_{NP_1}|)} \\ &= \frac{|PN| - |\Delta_{PN_1}| + |\Delta_{NN_1}|}{|P| - |\Delta P_1| + |\Delta N_1|}\end{aligned}\tag{A.79}$$

where in the second equality we have used the definition of ΔP_1 and ΔN_1 in Eq. (A.73).

Using the definition of ρ_1 , we have

$$\begin{aligned}
\frac{|PN| - |\Delta_{PN_1}|}{|P| - |\Delta P_1|} &= \frac{|\{x \in PN | g(x) \geq LB_{y=1}\}|}{|\{x \in P | g(x) \geq LB_{y=1}\}|} \\
&= \frac{P(x \in PN, g(x) \geq LB_{y=1})}{P(x \in P, g(x) \geq LB_{y=1})} \\
&= \frac{P(x \in PN | x \in P, g(x) \geq LB_{y=1}) \cdot P(x \in P, g(x) \geq LB_{y=1})}{P(x \in P, g(x) \geq LB_{y=1})} \\
&= \frac{P(x \in PN | x \in P) \cdot P(x \in P, g(x) \geq LB_{y=1})}{P(x \in P, g(x) \geq LB_{y=1})} \\
&= \rho_1
\end{aligned}$$

Here we have used the property of CNP that $(s \perp\!\!\!\perp x) | y$, leading to $P(x \in PN | x \in P, g(x) \geq LB_{y=1}) = P(x \in PN | x \in P) = \rho_1$.

Similarly, we have

$$\frac{|\Delta_{NN_1}|}{|\Delta N_1|} = 1 - \rho_0$$

Combining with Eq. (A.79), we have

$$\hat{\rho}_1^{conf} = \rho_1 + \frac{1 - \rho_1 - \rho_0}{|P| - |\Delta P_1| + |\Delta N_1|} |\Delta N_1| \quad (\text{A.80})$$

Similarly, we have

$$\hat{\rho}_0^{conf} = \rho_0 + \frac{1 - \rho_1 - \rho_0}{|N| - |\Delta N_0| + |\Delta P_0|} |\Delta P_0| \quad (\text{A.81})$$

From the two equations above, we see that

$$\hat{\rho}_1^{conf} \geq \rho_1, \hat{\rho}_0^{conf} \geq \rho_0 \quad (\text{A.82})$$

In other words, $\hat{\rho}_i^{\text{conf}}$ is an **upper bound** of ρ_i , $i = 0, 1$. The equality for $\hat{\rho}_1^{\text{conf}}$ holds if $|\Delta N_1| = 0$. The equality for $\hat{\rho}_0^{\text{conf}}$ holds if $|\Delta P_0| = 0$.

Proof of Theorem 4

Let's restate Theorem 4 below:

Theorem 4 *Given non-overlapping support condition,*

If $\forall x \in N, \Delta g(x) < LB_{y=1} - \rho_0$, then $\hat{\rho}_1^{\text{conf}} = \rho_1$.

If $\forall x \in P, \Delta g(x) > -(1 - \rho_1 - UB_{y=0})$, then $\hat{\rho}_0^{\text{conf}} = \rho_0$.

Theorem 4 directly follows from Eq. (A.80) and (A.81). Assuming non-overlapping support, we have $g^*(x) = P(s = 1|x) = (1 - \rho_1) \cdot \mathbb{1}[[y = 1]] + \rho_0 \cdot \mathbb{1}[[y = 0]]$. In other words, the contribution of overlapping support to $|\Delta N_1|$ and $|\Delta P_0|$ is 0. The only source of deviation comes from imperfect $g(x)$.

For the first half of the theorem, since $\forall x \in N, \Delta g(x) < LB_{y=1} - \rho_0$, we have $\forall x \in N, g(x) = \Delta g(x) + g^*(x) < (LB_{y=1} - \rho_0) + \rho_0 = LB_{y=1}$, then $|\Delta N_1| = |\{x \in N | g(x) \geq LB_{y=1}\}| = 0$, so we have $\hat{\rho}_1^{\text{conf}} = \rho_1$.

Similarly, for the second half of the theorem, since $\forall x \in P, \Delta g(x) > -(1 - \rho_1 - UB_{y=0})$, then $|\Delta P_0| = |\{x \in P | g(x) \leq UB_{y=0}\}| = 0$, so we have $\hat{\rho}_0^{\text{conf}} = \rho_0$.

Proof of Theorem 5

Theorem 5 reads as follows:

Theorem 5 *If g range separates P and N and $\hat{\rho}_i = \rho_i$, $i = 0, 1$, then for any classifier f_θ and any bounded loss function $l(\hat{y}_i, y_i)$, we have*

$$R_{\tilde{l}, \mathcal{D}_\rho}(f_\theta) = R_{l, \mathcal{D}}(f_\theta) \quad (\text{A.83})$$

where $\tilde{l}(\hat{y}_i, s_i)$ is Rank Pruning's loss function given by

$$\tilde{l}(\hat{y}_i, s_i) = \frac{1}{1 - \hat{\rho}_1} l(\hat{y}_i, s_i) \cdot \mathbb{1}[[x_i \in \tilde{P}_{conf}]] + \frac{1}{1 - \hat{\rho}_0} l(\hat{y}_i, s_i) \cdot \mathbb{1}[[x_i \in \tilde{N}_{conf}]] \quad (\text{A.84})$$

and \tilde{P}_{conf} and \tilde{N}_{conf} are given by

$$\tilde{P}_{conf} := \{x \in \tilde{P} \mid g(x) \geq k_1\}, \tilde{N}_{conf} := \{x \in \tilde{N} \mid g(x) \leq k_0\} \quad (\text{A.85})$$

where k_1 is the $(\hat{\pi}_1|\tilde{P}|)^{th}$ smallest $g(x)$ for $x \in \tilde{P}$ and k_0 is the $(\hat{\pi}_0|\tilde{N}|)^{th}$ largest $g(x)$ for $x \in \tilde{N}$

Proof:

Since \tilde{P} and \tilde{N} are constructed from P and N with noise rates π_1 and π_0 using the class-conditional extension of the Classification Noise Process [AL88], we have

$$\begin{cases} P = PP \cup PN \\ N = NP \cup NN \\ \tilde{P} = PP \cup NP \\ \tilde{N} = PN \cup NN \end{cases} \quad (\text{A.86})$$

where

$$\begin{cases} PP = \{x \in P \mid s = 1\} \\ PN = \{x \in P \mid s = 0\} \\ NP = \{x \in N \mid s = 1\} \\ NN = \{x \in N \mid s = 0\} \end{cases} \quad (\text{A.87})$$

satisfying

$$\begin{cases} PP \sim PN \sim P \\ NP \sim NN \sim N \\ \frac{|NP|}{|\tilde{P}|} = \pi_1, \frac{|PP|}{|\tilde{P}|} = 1 - \pi_1 \\ \frac{|PN|}{|\tilde{N}|} = \pi_0, \frac{|NN|}{|\tilde{N}|} = 1 - \pi_0 \\ \frac{|PN|}{|P|} = \rho_1, \frac{|PP|}{|P|} = 1 - \rho_1 \\ \frac{|NP|}{|N|} = \rho_0, \frac{|NN|}{|N|} = 1 - \rho_0 \end{cases} \quad (\text{A.88})$$

Here the \sim means obeying the same distribution.

Since g range separates P and N , there exists a real number z such that $\forall x_1 \in P$ and $\forall x_0 \in N$, we have $g(x_1) > z > g(x_0)$. Since $P = PP \cup PN$, $N = NP \cup NN$, we have

$$\begin{aligned} \forall x \in PP, g(x) &> z; \quad \forall x \in PN, g(x) > z; \\ \forall x \in NP, g(x) &< z; \quad \forall x \in NN, g(x) < z \end{aligned} \quad (\text{A.89})$$

Since $\hat{\rho}_1 = \rho_1$ and $\hat{\rho}_0 = \rho_0$, we have

$$\begin{cases} \hat{\pi}_1 = \frac{\hat{\rho}_0}{p_{s1}} \frac{1-p_{s1}-\hat{\rho}_1}{1-\hat{\rho}_1-\hat{\rho}_0} = \frac{\rho_0}{p_{s1}} \frac{1-p_{s1}-\rho_1}{1-\rho_1-\rho_0} = \pi_1 \equiv \frac{\rho_0|N|}{|\tilde{P}|} \\ \hat{\pi}_0 = \frac{\hat{\rho}_1}{1-p_{s1}} \frac{p_{s1}-\hat{\rho}_0}{1-\hat{\rho}_1-\hat{\rho}_0} = \frac{\rho_1}{1-p_{s1}} \frac{p_{s1}-\rho_0}{1-\rho_1-\rho_0} = \pi_0 \equiv \frac{\rho_1|P|}{|\tilde{N}|} \end{cases} \quad (\text{A.90})$$

Therefore, $\hat{\pi}_1|\tilde{P}| = \pi_1|\tilde{P}| = \rho_0|N|$, $\hat{\pi}_0|\tilde{N}| = \pi_0|\tilde{N}| = \rho_1|P|$. Using \tilde{P}_{conf} and \tilde{N}_{conf} 's definition in Eq. (A.85), and $g(x)$'s property in Eq. (A.89), we have

$$\tilde{P}_{conf} = PP \sim P, \tilde{N}_{conf} = NN \sim N \quad (\text{A.91})$$

Hence P_{conf} and N_{conf} can be seen as a uniform downsampling of P and N , with a downsampling ratio of $(1 - \rho_1)$ for P and $(1 - \rho_0)$ for N . Then according to Eq. (A.84), the loss function $\tilde{l}(\hat{y}_i, s_i)$ essentially sees a fraction of $(1 - \rho_1)$ examples in P and a fraction of $(1 - \rho_0)$ examples in N , with a final reweighting to restore the class balance. Then for

any classifier f_θ that maps $x \rightarrow \hat{y}$ and any bounded loss function $l(\hat{y}_i, y_i)$, we have

$$\begin{aligned}
R_{\tilde{l}, \mathcal{D}_\rho}(f_\theta) &= E_{(x,s) \sim \mathcal{D}_\rho} [\tilde{l}(f_\theta(x), s)] \\
&= \frac{1}{1 - \hat{\rho}_1} \cdot E_{(x,s) \sim \mathcal{D}_\rho} [l(f_\theta(x), s) \cdot \mathbb{1}[[x \in \tilde{P}_{conf}]]] \\
&\quad + \frac{1}{1 - \hat{\rho}_0} \cdot E_{(x,s) \sim \mathcal{D}_\rho} [l(f_\theta(x), s) \cdot \mathbb{1}[[x \in \tilde{N}_{conf}]]] \\
&= \frac{1}{1 - \rho_1} \cdot E_{(x,s) \sim \mathcal{D}_\rho} [l(f_\theta(x), s) \cdot \mathbb{1}[[x \in \tilde{P}_{conf}]]] \\
&\quad + \frac{1}{1 - \rho_0} \cdot E_{(x,s) \sim \mathcal{D}_\rho} [l(f_\theta(x), s) \cdot \mathbb{1}[[x \in \tilde{N}_{conf}]]] \\
&= \frac{1}{1 - \rho_1} \cdot E_{(x,s) \sim \mathcal{D}_\rho} [l(f_\theta(x), s) \cdot \mathbb{1}[[x \in PP]]] \\
&\quad + \frac{1}{1 - \rho_0} \cdot E_{(x,s) \sim \mathcal{D}_\rho} [l(f_\theta(x), s) \cdot \mathbb{1}[[x \in NN]]] \\
&= \frac{1}{1 - \rho_1} \cdot (1 - \rho_1) \cdot E_{(x,y) \sim \mathcal{D}} [l(f_\theta(x), y) \cdot \mathbb{1}[[x \in P]]] \\
&\quad + \frac{1}{1 - \rho_0} \cdot (1 - \rho_0) \cdot E_{(x,y) \sim \mathcal{D}} [l(f_\theta(x), y) \cdot \mathbb{1}[[x \in N]]] \\
&= E_{(x,y) \sim \mathcal{D}} [l(f_\theta(x), y) \cdot \mathbb{1}[[x \in P]] + l(f_\theta(x), y) \cdot \mathbb{1}[[x \in N]]] \\
&= E_{(x,y) \sim \mathcal{D}} [l(f_\theta(x), y)] \\
&= R_{l, \mathcal{D}}(f_\theta)
\end{aligned}$$

Therefore, we see that the expected risk for Rank Pruning with corrupted labels, is exactly the same as the expected risk for the true labels, for any bounded loss function l and classifier f_θ . The reweighting ensures that after pruning, the two sets still remain unbiased w.r.t. to the true dataset.

Since the ideal condition is more strict than the range separability condition, we immediately have that when g is ideal and $\hat{\rho}_i = \rho_i$, $i = 0, 1$, $R_{\tilde{l}, \mathcal{D}_\rho}(f_\theta) = R_{l, \mathcal{D}}(f_\theta)$ for any f_θ and bounded loss function l .

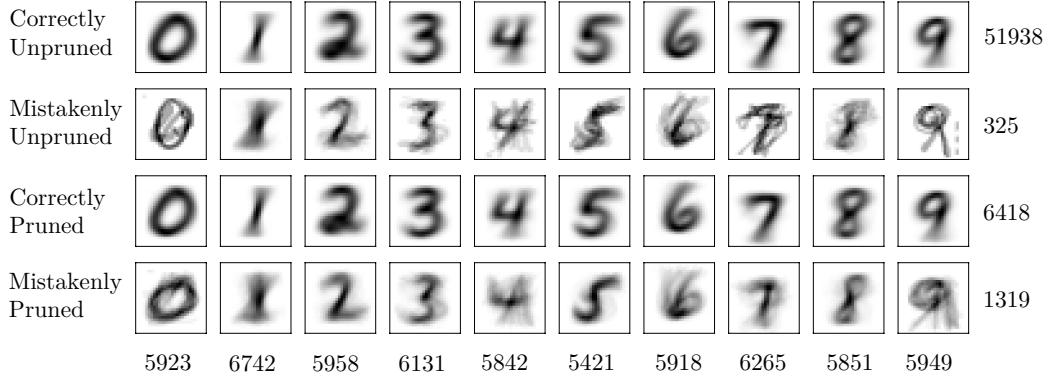


Figure S A.10: Average image for each digit for the binary classification problem “1” or “not 1” in MNIST with logistic regression and significant mislabeling ($\rho_1 = 0.5, \pi_1 = 0.5$). The right and bottom numbers count the total number of example images averaged in the corresponding row or column.

A.7.2 Additional Figures

Figure BA.10 shows the average image for each digit for the problem $\|IJ1\|$ or $\|IJnot1\|$ in MNIST with logistic regression and high noise ($\rho_1 = 0.5, \pi_1 = 0.5$). The number on the bottom and on the right counts the total number of examples (images). From the figure we see that Rank Pruning makes few mistakes, and when it does, the mistakes vary greatly in image from the typical digit.

A.7.3 Additional Tables

Here we provide additional tables for the comparison of error, Precision-Recall AUC (AUC-PR, [DG06b]), and F1 score for the algorithms *RP*, *Nat13*, *Elk08*, *Liu16* with ρ_1, ρ_0 given to all methods for fair comparison. Additionally, we provide the performance of the ground truth classifier (*true*) trained with uncorrupted labels (X, y) , as well as the complete Rank Pruning algorithm (RP_ρ) trained using the noise rates estimated by Rank Pruning. The top model scores are in bold with RP_ρ in red if its performance is better than non-RP models. The $\pi_1 = 0$ quadrant in each table represents the “PU learning” case of $\tilde{P}\tilde{N}$ learning.

Whenever $g(x) = P(\hat{s} = 1|x)$ is estimated for any algorithm, we use a 3-fold cross-validation to estimate the probability $g(x)$. For improved performance, a higher fold may be

used.

For the logistic regression classifier, we use scikit-learn's LogisticRegression class ([sl16]) with default settings (L2 regularization with inverse strength $C = 1$).

For the convolutional neural networks (CNN), for MNIST we use the structure in [Cho16b] and for CIFAR-10, we use the structure in [Cho16a]. A 10% holdout set is used to monitor the weighted validation loss (using the sample weight given by each algorithm) and ends training when there is no decrease for 10 epochs, with a maximum of 50 epochs for MNIST and 150 epochs for CIFAR-10.

The following list comprises the MNIST and CIFAR experimental result tables for error, AUC-PR and F1 score metrics:

Table CA.3: Error for MNIST with logistic regression as classifier.

Table CA.4: AUC-PR for MNIST with logistic regression as classifier.

Table CA.5: Error for MNIST with CNN as classifier.

Table CA.6: AUC-PR for MNIST with CNN as classifier.

Table CA.7: F1 score for CIFAR-10 with logistic regression as classifier.

Table CA.8: Error for CIFAR-10 with logistic regression as classifier.

Table CA.9: AUC-PR for CIFAR-10 with logistic regression as classifier.

Table CA.10: Error for CIFAR-10 with CNN as classifier.

Table CA.11: AUC-PR for CIFAR-10 with CNN as classifier.

Due to sensitivity to imperfect probability estimation, here *Liu16* always predicts all labels to be positive or negative, resulting in the same metric score for every digit/image in each scenario. Since $p_{y1} \simeq 0.1$, when predicting all labels as positive, *Liu16* has an F1 score of 0.182, error of 0.90, and AUC-PR of 0.55; when predicting all labels as negative, *Liu16* has an F1 score of 0.0, error of 0.1, and AUC-PR of 0.55.

A.7.4 Additional Related Work

In this section we include tangentially related work which was unable to make it into the final manuscript.

One-class classification

One-class classification [MKH93] is distinguished from binary classification by a training set containing examples from only one class, making it useful for outlier and novelty detection [HFW08]. This can be framed as $\tilde{P}\tilde{N}$ learning when outliers take the form of mislabeled examples. The predominant approach, one-class SVM, fits a hyper-boundary around the training class [PSST⁺99], but often performs poorly due to boundary oversensitivity [MY02] and fails when the training class contains mislabeled examples.

$\tilde{P}\tilde{N}$ learning for Image Recognition and Deep Learning

Variations of $\tilde{P}\tilde{N}$ learning have been used in the context of machine vision to improve robustness to mislabeling [XXY⁺15b]. In a face recognition task with 90% of non-faces mislabeled as faces, a bagging model combined with consistency voting was used to remove images with poor voting consistency [AAMP05]. However, no theoretical justification was provided. In the context of deep learning, consistency of predictions for inputs with mislabeling enforces can be enforced by combining a typical cross-entropy loss with an auto-encoder loss [RLA⁺15]. This method enforces label consistency by constraining the network to uncover the input examples given the output prediction, but is restricted in architecture and generality.

Table S A.3: Comparison of **error** for one-vs-rest MNIST (averaged over all digits) using a **logistic regression** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if better (smaller) than non-RP models.

MODEL, $\rho_1 =$	$\pi_1 = 0$			$\pi_1 = 0.25$			$\pi_1 = 0.5$			$\pi_1 = 0.75$					
	0.25	0.50	0.75	0.00	0.25	0.50	0.75	0.00	0.25	0.50	0.75	0.00	0.25	0.50	0.75
TRUE	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020
RP_ρ	0.023	0.025	0.031	0.024	0.025	0.027	0.038	0.040	0.037	0.039	0.049	0.140	0.128	0.133	0.151
RP	0.022	0.025	0.031	0.021	0.024	0.027	0.035	0.023	0.027	0.031	0.043	0.028	0.036	0.045	0.069
NAT13	0.025	0.030	0.038	0.025	0.029	0.034	0.042	0.030	0.033	0.038	0.047	0.035	0.039	0.046	0.067
ELK08	0.025	0.030	0.038	0.026	0.028	0.032	0.042	0.030	0.031	0.035	0.051	0.092	0.093	0.123	0.189
LIU16	0.187	0.098	0.100	0.100	0.738	0.738	0.419	0.100	0.820	0.821	0.821	0.098	0.760	0.741	0.820

Table S A.4: Comparison of **AUC-PR** for one-vs-rest MNIST (averaged over all digits) using a **logistic regression** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if greater than non-RP models.

MODEL, $\rho_1 =$	$\pi_1 = 0$			$\pi_1 = 0.25$			$\pi_1 = 0.5$			$\pi_1 = 0.75$					
	0.25	0.50	0.75	0.00	0.25	0.50	0.75	0.00	0.25	0.50	0.75	0.00	0.25	0.50	0.75
TRUE	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.935
RP_ρ	0.921	0.913	0.882	0.928	0.920	0.906	0.853	0.903	0.902	0.879	0.803	0.851	0.835	0.788	0.640
RP	0.922	0.913	0.882	0.930	0.921	0.906	0.858	0.922	0.903	0.883	0.811	0.893	0.841	0.799	0.621
NAT13	0.922	0.908	0.878	0.918	0.909	0.890	0.839	0.899	0.892	0.862	0.794	0.863	0.837	0.784	0.645
ELK08	0.921	0.903	0.864	0.917	0.908	0.884	0.821	0.898	0.892	0.861	0.763	0.852	0.837	0.772	0.579
LIU16	0.498	0.549	0.550	0.550	0.500	0.550	0.505	0.550	0.550	0.550	0.549	0.503	0.512	0.550	0.550

Table S A.5: Comparison of **error** for one-vs-rest MNIST (averaged over all digits) using a **CNN** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if better (smaller) than non-RP models.

IMAGE	TRUE	$\pi_1 = 0$			$\pi_1 = 0.25$			$\pi_1 = 0.5$			$\pi_1 = 0.75$										
		RP_ρ	RP	$\rho_1 = 0.5$	$\rho_1 = 0.5$	RP_ρ	RP	$\rho_1 = 0.25$	$\rho_1 = 0.25$	RP_ρ	RP	$\rho_1 = 0$	$\rho_1 = 0$	RP_ρ	RP	$\rho_1 = 0.5$	$\rho_1 = 0.5$				
0	0.0013	0.0018	0.0023	0.0045	0.0047	0.9020	0.0017	0.0016	0.0034	0.0036	0.9020	0.0017	0.0016	0.0031	0.0026	0.0029	0.0021	0.0022	0.116	0.0069	0.9020
1	0.0015	0.0022	0.0020	0.0025	0.0034	0.8865	0.0019	0.0019	0.0035	0.0030	0.8865	0.0023	0.0020	0.0018	0.0016	0.0023	0.0025	0.0036	0.0027	0.8865	
2	0.0027	0.0054	0.0049	0.0057	0.0062	0.8968	0.0032	0.0035	0.0045	0.0051	0.8968	0.0030	0.0029	0.0031	0.0029	0.0024	0.0059	0.0050	0.0066	0.0083	0.8968
3	0.0020	0.0032	0.0032	0.0055	0.0038	0.8990	0.0029	0.0029	0.0043	0.0043	0.8990	0.0021	0.0027	0.0023	0.0032	0.0038	0.0042	0.0084	0.0057	0.8990	
4	0.0012	0.0037	0.0040	0.0038	0.0044	0.9018	0.0029	0.0025	0.0055	0.0069	0.9018	0.0026	0.0020	0.0019	0.0021	0.0030	0.0044	0.0035	0.0086	0.0077	0.9018
5	0.0019	0.0032	0.0035	0.0039	0.0038	0.9108	0.0027	0.0031	0.0062	0.0060	0.9108	0.0021	0.0024	0.0024	0.0028	0.0023	0.0061	0.0056	0.0066	0.0074	0.9108
6	0.0021	0.0027	0.0028	0.0053	0.0035	0.9042	0.0028	0.0025	0.0042	0.0036	0.9042	0.0029	0.0029	0.0024	0.0028	0.0032	0.0035	0.0098	0.0075	0.9042	
7	0.0026	0.0039	0.0041	0.0066	0.0103	0.8972	0.0050	0.0052	0.0058	0.0058	0.8972	0.0049	0.0040	0.0030	0.0037	0.0035	0.0054	0.0064	0.0113	0.0085	0.8972
8	0.0022	0.0047	0.0043	0.0106	0.0063	0.9026	0.0034	0.0036	0.0062	0.0091	0.9026	0.0036	0.0030	0.0035	0.0041	0.0032	0.0044	0.0048	0.0234	0.0077	0.9026
9	0.0036	0.0067	0.0052	0.0056	0.0124	0.8991	0.0048	0.0051	0.0065	0.0064	0.8991	0.0048	0.0050	0.0051	0.0043	0.0059	0.0081	0.0114	0.0131	0.0112	0.8991
AVG	0.0021	0.0038	0.0036	0.0054	0.0059	0.9000	0.0031	0.0032	0.0050	0.0054	0.9000	0.0030	0.0028	0.0028	0.0029	0.0032	0.0046	0.0049	0.0103	0.0074	0.9000

Table S A.6: Comparison of **AUC-PR** for one-vs-rest MNIST (averaged over all digits) using a **CNN** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if greater than non-RP models.

IMAGE	TRUE	$\pi_1 = 0$			$\pi_1 = 0.25$			$\pi_1 = 0.5$					
		RP_ρ	RP	$\rho_1 = 0.5$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0.25$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0.5$ NAT13 ELK08 LIU16
0	0.9998	0.9992 0.9990	0.9986 0.9982 0.5490	0.9996 0.9996	0.9986 0.9979 0.5490	0.9989 0.9995	0.9976 0.9979 0.9956	0.9984 0.9982	0.9963 0.9928 0.5490	0.9984 0.9982	0.9963 0.9928 0.5490	0.9984 0.9982	0.9963 0.9928 0.5490
1	0.9999	0.9995 0.9995	0.9976 0.9974 0.5568	0.9996 0.9993	0.9995 0.9995 0.5568	0.9995 0.9998	0.9982 0.9972 0.9965	0.9995 0.9994	0.9978 0.9985 0.5568	0.9995 0.9994	0.9978 0.9985 0.5568	0.9995 0.9994	0.9978 0.9985 0.5568
2	0.9994	0.9971 0.9969	0.9917 0.9942 0.5516	0.9980 0.9977	0.9971 0.9945 0.5516	0.9988 0.9992	0.9958 0.9934 0.9940	0.9938 0.9947	0.9847 0.9873 0.5516	0.9938 0.9947	0.9847 0.9873 0.5516	0.9938 0.9947	0.9847 0.9873 0.5516
3	0.9996	0.9986 0.9987	0.9983 0.9984 0.5505	0.9991 0.9989	0.9982 0.9980 0.5505	0.9993 0.9994	0.9991 0.9974 0.9974	0.9969 0.9959	0.9951 0.9959 0.5505	0.9969 0.9959	0.9951 0.9959 0.5505	0.9969 0.9959	0.9951 0.9959 0.5505
4	0.9997	0.9982 0.9989	0.9939 0.9988 0.0891	0.9992 0.9991	0.9976 0.9965 0.5491	0.9994 0.9996	0.9985 0.9978 0.9986	0.9983 0.9977	0.9961 0.9919 0.5491	0.9983 0.9977	0.9961 0.9919 0.5491	0.9983 0.9977	0.9961 0.9919 0.5491
5	0.9997	0.9982 0.9976	0.9969 0.9956 0.5446	0.9986 0.9987	0.9983 0.9979 0.5446	0.9984 0.9982	0.9971 0.9963 0.9929	0.9958 0.9965	0.9946 0.9934 0.5446	0.9958 0.9965	0.9946 0.9934 0.5446	0.9958 0.9965	0.9946 0.9934 0.5446
6	0.9987	0.9976 0.9970	0.9928 0.9931 0.5479	0.9974 0.9980	0.9956 0.9959 0.5479	0.9968 0.9983	0.9933 0.9950 0.9905	0.9964 0.9957	0.9942 0.9961 0.5479	0.9964 0.9957	0.9942 0.9961 0.5479	0.9964 0.9957	0.9942 0.9961 0.5479
7	0.9989	0.9973 0.9972	0.9965 0.9944 0.0721	0.9968 0.9973 0.9966	0.9979 0.5514	0.9969 0.9983 0.9961	0.9958 0.9974 0.9974	0.9933 0.9937	0.9896 0.9886 0.5514	0.9933 0.9937	0.9896 0.9886 0.5514	0.9933 0.9937	0.9896 0.9886 0.5514
8	0.9996	0.9974 0.9964	0.9964 0.9946 0.5487	0.9981 0.9981	0.9973 0.9971 0.5487	0.9983 0.9988	0.9984 0.9976 0.9989	0.9976 0.9975	0.9873 0.9893 0.5487	0.9976 0.9975	0.9873 0.9893 0.5487	0.9976 0.9975	0.9873 0.9893 0.5487
9	0.9979	0.9931 0.9951	0.9901 0.9922 0.5504	0.9935 0.9951	0.9933 0.9920 0.5504	0.9961 0.9951	0.9924 0.9922 0.9912	0.9877 0.9876	0.9819 0.9828 0.5504	0.9877 0.9876	0.9819 0.9828 0.5504	0.9877 0.9876	0.9819 0.9828 0.5504
AVG	0.9993	0.9976 0.9976	0.9953 0.9957 0.4561	0.9980 0.9982	0.9972 0.9967 0.5500	0.9983 0.9986	0.9966 0.9960 0.9953	0.9958 0.9957	0.9918 0.9917 0.5500	0.9958 0.9957	0.9918 0.9917 0.5500	0.9958 0.9957	0.9918 0.9917 0.5500

Table S A.7: Comparison of **F1 score** for one-vs-rest CIFAR-10 (averaged over all images) using a **logistic regression** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if greater than non-RP models.

IMAGE	TRUE	$\pi_1 = 0$			$\pi_1 = 0.25$			$\pi_1 = 0.5$					
		RP_ρ	RP	$\rho_1 = 0$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0.25$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0.5$ NAT13 ELK08 LIU16
PLANE	0.272	0.311 0.252	0.217 0.220 0.182	0.329 0.275	0.222 0.224 0.182	0.330 0.265	0.231 0.259 0.0	0.266 0.188	0.183 0.187 0.182	0.266 0.188	0.183 0.187 0.182	0.266 0.188	0.183 0.187 0.182
AUTO	0.374	0.389 0.355	0.318 0.320 0.182	0.388 0.368	0.321 0.328 0.182	0.372 0.355	0.308 0.341 0.0	0.307 0.287	0.287 0.287 0.297	0.307 0.287	0.287 0.287 0.297	0.307 0.287	0.287 0.287 0.297
BIRD	0.136	0.241 0.167	0.143 0.130 0.182	0.248 0.185	0.137 0.137 0.182	0.258 0.147	0.100 0.126 0.0	0.206 0.153	0.132 0.150 0.182	0.206 0.153	0.132 0.150 0.182	0.206 0.153	0.132 0.150 0.182
CAT	0.122	0.246 0.170	0.141 0.150 0.182	0.232 0.163	0.112 0.127 0.182	0.241 0.125	0.068 0.103 0.0	0.209 0.148	0.119 0.157 0.182	0.209 0.148	0.119 0.157 0.182	0.209 0.148	0.119 0.157 0.182
DEER	0.166	0.250 0.184	0.153 0.164 0.182	0.259 0.175	0.146 0.163 0.182	0.259 0.177	0.126 0.164 0.0	0.222 0.162	0.132 0.164 0.182	0.222 0.162	0.132 0.164 0.182	0.222 0.162	0.132 0.164 0.182
DOG	0.139	0.245 0.174	0.146 0.148 0.182	0.262 0.171	0.115 0.126 0.182	0.262 0.152	0.075 0.120 0.0	0.203 0.151	0.128 0.137 0.182	0.203 0.151	0.128 0.137 0.182	0.203 0.151	0.128 0.137 0.182
FROG	0.317	0.322 0.315	0.289 0.281 0.182	0.350 0.319	0.283 0.299 0.182	0.346 0.305	0.239 0.279 0.0	0.308 0.252	0.244 0.244 0.269	0.308 0.252	0.244 0.244 0.269	0.308 0.252	0.244 0.244 0.269
HORSE	0.300	0.300 0.299	0.283 0.263 0.182	0.334 0.313	0.272 0.281 0.182	0.322 0.310	0.260 0.292 0.0	0.275 0.258	0.240 0.245 0.182	0.275 0.258	0.240 0.245 0.182	0.275 0.258	0.240 0.245 0.182
SHIP	0.322	0.343 0.322	0.297 0.272 0.182	0.385 0.319	0.287 0.289 0.182	0.350 0.303	0.250 0.293 0.0	0.304 0.248	0.230 0.237 0.182	0.304 0.248	0.230 0.237 0.182	0.304 0.248	0.230 0.237 0.182
TRUCK	0.330	0.359 0.323	0.273 0.261 0.182	0.369 0.327	0.293 0.290 0.182	0.343 0.302	0.278 0.299 0.0	0.313 0.246	0.252 0.262 0.182	0.313 0.246	0.252 0.262 0.182	0.313 0.246	0.252 0.262 0.182
AVG	0.248	0.301 0.256	0.226 0.221 0.182	0.316 0.262	0.219 0.226 0.182	0.308 0.244	0.194 0.228 0.000	0.261 0.209	0.195 0.195 0.210	0.261 0.209	0.195 0.195 0.210	0.261 0.209	0.195 0.195 0.210

Table S A.8: Comparison of **error** for one-vs-rest CIFAR-10 (averaged over all images) using a **logistic regression** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if better (smaller) than non-RP models. Here the logistic regression classifier severely underfits CIFAR, resulting in Rank Pruning pruning out some correctly labeled examples that “confuse” the classifier, hence in this scenario, RP and RP_ρ generally have slightly smaller precision, much higher recall, and hence larger F1 scores than other models and even the ground truth classifier (Table CA.7). Due to the class imbalance ($p_{y1} = 0.1$) and their larger recall, RP and RP_ρ here have larger error than the other models.

IMAGE	TRUE	$\pi_1 = 0$			$\pi_1 = 0.25$			$\pi_1 = 0.5$					
		RP_ρ	RP	$\rho_1 = 0$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0.25$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0$ NAT13 ELK08 LIU16	RP_ρ	RP	$\rho_1 = 0.5$ NAT13 ELK08 LIU16
PLANE	0.107	0.287 0.133	0.123 0.122	0.900 0.177 0.128	0.119 0.123	0.900 0.177 0.128	0.119 0.123	0.900 0.248 0.124	0.110 0.118 0.100	0.100 0.118 0.100	0.202 0.147 0.142	0.160 0.900 0.900	0.142 0.160 0.900
AUTO	0.099	0.184 0.120	0.110 0.110	0.900 0.132 0.114	0.105 0.109	0.900 0.189 0.110	0.105 0.110 0.105	0.110 0.159 0.129	0.125 0.139 0.139	0.125 0.139 0.139	0.125 0.139 0.139	0.125 0.139 0.139	0.125 0.139 0.139
BIRD	0.117	0.354 0.148	0.133 0.131	0.900 0.217 0.135	0.120 0.125	0.900 0.277 0.135	0.115 0.123 0.100	0.122 0.226 0.147	0.139 0.158 0.900	0.139 0.226 0.147	0.139 0.158 0.900	0.139 0.226 0.147	0.139 0.158 0.900
CAT	0.114	0.351 0.138	0.129 0.129	0.900 0.208 0.139	0.122 0.125	0.900 0.303 0.132	0.114 0.122 0.100	0.122 0.225 0.151	0.141 0.158 0.900	0.141 0.225 0.151	0.141 0.		

Table S A.9: Comparison of **AUC-PR** for one-vs-rest CIFAR-10 (averaged over all images) using a **logistic regression** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if greater than non-RP models. Since $p_{y1} = 0.1$, here *Liu16* always predicts all labels as positive or negative, resulting in a constant AUC-PR of 0.550.

IMAGE	TRUE	$\pi_1 = 0$				$\pi_1 = 0.25$				$\pi_1 = 0.5$						
		RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16
PLANE	0.288	0.225	0.224	0.225	0.207	0.550	0.261	0.235	0.225	0.217	0.550	0.285	0.251	0.245	0.248	0.550
AUTO	0.384	0.350	0.317	0.312	0.316	0.550	0.342	0.335	0.331	0.331	0.550	0.328	0.348	0.334	0.333	0.550
BIRD	0.198	0.160	0.169	0.166	0.161	0.550	0.188	0.185	0.179	0.177	0.550	0.186	0.173	0.174	0.175	0.550
CAT	0.188	0.164	0.175	0.174	0.175	0.550	0.163	0.169	0.168	0.170	0.550	0.148	0.156	0.154	0.152	0.550
DEER	0.215	0.161	0.177	0.180	0.183	0.550	0.194	0.180	0.180	0.182	0.550	0.174	0.175	0.176	0.175	0.550
DOG	0.188	0.162	0.161	0.165	0.155	0.550	0.175	0.160	0.161	0.158	0.550	0.173	0.169	0.162	0.164	0.550
FROG	0.318	0.246	0.264	0.262	0.258	0.550	0.292	0.277	0.272	0.273	0.550	0.276	0.274	0.277	0.277	0.550
HORSE	0.319	0.242	0.267	0.269	0.260	0.550	0.283	0.264	0.264	0.263	0.550	0.288	0.282	0.279	0.278	0.550
SHIP	0.317	0.257	0.267	0.271	0.248	0.550	0.296	0.266	0.267	0.259	0.550	0.279	0.268	0.259	0.262	0.550
TRUCK	0.329	0.288	0.261	0.271	0.263	0.550	0.298	0.275	0.286	0.284	0.550	0.289	0.272	0.276	0.277	0.550
AVG	0.274	0.226	0.228	0.229	0.223	0.550	0.249	0.235	0.233	0.231	0.550	0.243	0.237	0.234	0.234	0.550
		0.197	0.187	0.183	0.181	0.550										

Table S A.10: Comparison of **error** for one-vs-rest CIFAR-10 (averaged over all images) using a **CNN** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if better (smaller) than non-RP models.

IMAGE	TRUE	$\pi_1 = 0$				$\pi_1 = 0.25$				$\pi_1 = 0.5$									
		RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16			
PLANE	0.044	0.054	0.057	0.059	0.063	0.900	0.050	0.051	0.054	0.057	0.900	0.048	0.045	0.049	0.048	0.100	0.063	0.061	
AUTO	0.021	0.040	0.037	0.041	0.043	0.100	0.032	0.034	0.040	0.039	0.900	0.028	0.026	0.026	0.026	0.100	0.047	0.049	
BIRD	0.055	0.083	0.078	0.080	0.082	0.900	0.074	0.074	0.077	0.078	0.900	0.072	0.066	0.072	0.070	0.100	0.124	0.084	
CAT	0.077	0.108	0.091	0.092	0.095	0.100	0.111	0.100	0.090	0.086	0.089	0.090	0.113	0.084	0.086	0.088	0.100	0.117	0.098
DEER	0.049	0.081	0.078	0.078	0.079	0.900	0.080	0.069	0.075	0.070	0.900	0.076	0.062	0.061	0.062	0.100	0.106	0.086	0.091
DOG	0.062	0.075	0.071	0.079	0.080	0.100	0.071	0.069	0.070	0.067	0.900	0.069	0.061	0.057	0.076	0.100	0.103	0.081	0.084
FROG	0.038	0.050	0.048	0.048	0.054	0.100	0.047	0.052	0.056	0.062	0.900	0.045	0.040	0.042	0.043	0.100	0.058	0.062	0.066
HORSE	0.035	0.050	0.052	0.057	0.054	0.900	0.048	0.051	0.052	0.057	0.900	0.040	0.040	0.042	0.046	0.100	0.065	0.063	0.066
SHIP	0.028	0.042	0.042	0.046	0.042	0.900	0.037	0.036	0.042	0.047	0.900	0.035	0.030	0.031	0.033	0.100	0.051	0.049	0.064
TRUCK	0.027	0.044	0.046	0.054	0.056	0.900	0.034	0.032	0.038	0.043	0.900	0.034	0.031	0.034	0.034	0.100	0.060	0.066	0.067
AVG	0.043	0.063	0.060	0.064	0.065	0.580	0.059	0.056	0.059	0.061	0.900	0.056	0.049	0.050	0.053	0.100	0.080	0.070	0.076
		0.077	0.076	0.077	0.077	0.900	0.055	0.055	0.055	0.055	0.900	0.055	0.055	0.055	0.055	0.900	0.077	0.077	0.900

Table S A.11: Comparison of **AUC-PR** for one-vs-rest CIFAR-10 (averaged over all images) using a **CNN** classifier. Except for RP_ρ , ρ_1 , ρ_0 are given to all methods. Top model scores are in bold with RP_ρ in red if greater than non-RP models.

IMAGE	TRUE	$\pi_1 = 0$				$\pi_1 = 0.25$				$\pi_1 = 0.5$									
		RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16	RP_ρ	RP	NAT13	ELK08	LIU16			
PLANE	0.856	0.779	0.780	0.784	0.756	0.550	0.808	0.797	0.770	0.742	0.550	0.813	0.824	0.792	0.794	0.550	0.710	0.722	0.662
AUTO	0.954	0.874	0.889	0.878	0.833	0.550	0.905	0.900	0.871	0.866	0.550	0.931	0.927	0.924	0.910	0.550	0.824	0.814	0.756
BIRD	0.761	0.559	0.566	0.569	0.568	0.550	0.619	0.618	0.584	0.597	0.550	0.623	0.679	0.613	0.619	0.115	0.465	0.492	0.436
CAT	0.601	0.387	0.447	0.463	0.433	0.550	0.423	0.454	0.487	0.480	0.550	0.483	0.512	0.493	0.473	0.050	0.373	0.375	0.382
DEER	0.820	0.620	0.600	0.615	0.573	0.550	0.646	0.660	0.610	0.657	0.550	0.658	0.707	0.700	0.703	0.550	0.434	0.487	0.414
DOG	0.758	0.629	0.662	0.617	0.573	0.550	0.673	0.667	0.658	0.660	0.550	0.705	0.722	0.741	0.705	0.550	0.541	0.545	0.496
FROG	0.891	0.812	0.815	0.812	0.776	0.550	0.821	0.827	0.808	0.749	0.550	0.841	0.851	0.828	0.831	0.550	0.753	0.710	0.691
HORSE	0.897	0.810	0.817	0.799	0.550	0.824	0.809	0.801	0.772	0.550	0.826	0.844	0.818	0.819	0.550	0.736	0.699	0.699	
SHIP	0.922	0.870	0.862	0.864	0.853	0.550	0.889	0.885	0.843	0.848	0.550	0.889	0.897	0.891	0.887	0.550	0.800	0.808	0.767
TRUCK	0.929	0.845	0.848	0.824	0.787	0.550	0.887	0.894	0.873	0.853	0.550	0.904	0.902	0.898	0.883	0.550	0.740	0.709	0.695
AVG	0.839	0.719	0.729	0.722	0.693	0.550	0.750	0.751	0.730	0.722	0.550	0.767	0.787	0.770	0.762	0.457	0.637	0.636	0.600
		0.579	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559	0.559

Bibliography

- [AAMP05] Anelia Angelova, Yaser Abu-Mostafam, and Pietro Perona. Pruning training sets for learning of object categories. In *CVPR*, volume 1, pages 494–501. IEEE, 2005.
- [ADG⁺16] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989, 2016.
- [AFDM16] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.
- [AG19] Rana Ali Amjad and Bernhard Claus Geiger. Learning representations for neural network-based classification using the information bottleneck principle. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
- [AGKN13] Venkat Anantharam, Amin Gohari, Sudeep Kamath, and Chandra Nair. On maximal correlation, hypercontractivity, and the data processing inequality studied by erkip and cover. *arXiv preprint arXiv:1304.6133*, 2013.
- [AKA91] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Mach. Learn.*, 6(1):37–66, 1991.
- [AL88] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.
- [Alb91] James S Albus. Outline for a theory of intelligence. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):473–509, 1991.
- [AMS04] Nicola Ancona, Daniele Marinazzo, and Sebastiano Stramaglia. Radial basis function approach to nonlinear granger causality of time series. *Physical Review E*, 70(5):056221, 2004.
- [AMS18] Alessandro Achille, Glen Mbeng, and Stefano Soatto. The Dynamics of Differential Learning I: Information-Dynamics and Task Reachability. *arXiv preprint arXiv:1810.02440*, 2018.

- [Ana92] Anne Anastasi. What counselors should know about the use and interpretation of psychological tests. *Journal of Counseling & Development*, 70(5):610–615, 1992.
- [AOS⁺16] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.
- [Ari72] Suguru Arimoto. An algorithm for computing the capacity of arbitrary discrete memoryless channels. *IEEE Transactions on Information Theory*, 18(1):14–20, 1972.
- [AS18a] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980, 2018.
- [AS18b] Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.
- [BBC⁺17] Margaret Boden, Joanna Bryson, Darwin Caldwell, Kerstin Dautenhahn, Lilian Edwards, Sarah Kember, Paul Newman, Vivienne Parry, Geoff Pegman, Tom Rodden, et al. Principles of robotics: regulating robots in the real world. *Connection Science*, 29(2):124–129, 2017.
- [BBCG92] Samy Bengio, Yoshua Bengio, Jocelyn Cloutier, and Jan Gecsei. On the optimization of a synaptic learning rule. In *Preprints Conf. Optimality in Artificial and Biological Neural Networks*, pages 6–8. Univ. of Texas, 1992.
- [BCP⁺16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.
- [BES01] Elizabeth Bradley, Matthew Easley, and Reinhard Stolle. Reasoning about nonlinear system identification. *Artificial Intelligence*, 133(1):139 – 188, 2001.
- [BFP⁺73] Manuel Blum, Robert W. Floyd, Vaughan Pratt, Ronald L. Rivest, and Robert E. Tarjan. Time bounds for selection. *J. Comput. Syst. Sci.*, 7(4):448–461, August 1973.
- [BGNR16] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.
- [BHB⁺18] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

- [Bla72] Richard Blahut. Computation of channel capacity and rate-distortion functions. *IEEE transactions on Information Theory*, 18(4):460–473, 1972.
- [BLS10] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Semi-supervised novelty detection. *J. Mach. Learn. Res.*, 11:2973–3009, December 2010.
- [BM98] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *11th Conf. on COLT*, pages 92–100, New York, NY, USA, 1998. ACM.
- [BNVB13] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.
- [BO18] Léonard Blier and Yann Ollivier. The description length of deep learning models. 2018.
- [BPL⁺16] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and koray kavukcuoglu. Interaction networks for learning about objects, relations and physics. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4502–4510. Curran Associates, Inc., 2016.
- [BRB⁺18] Ishmael Belghazi, Sai Rajeswar, Aristide Baratin, R Devon Hjelm, and Aaron Courville. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*, 2018.
- [Bre96] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, August 1996.
- [BRLW17] Manabendra N Bera, Arnau Riera, Maciej Lewenstein, and Andreas Winter. Generalized laws of thermodynamics in the presence of correlations. *Nature communications*, 8(1):2180, 2017.
- [BSW14] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5:4308, 2014.
- [BSXT18] Neil Bramley, Eric Schulz, Fei Xu, and Joshua Tenenbaum. Learning as program induction. 2018.
- [Čer85] Vladimír Černý. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of optimization theory and applications*, 45(1):41–51, 1985.
- [CGTW05] Gal Chechik, Amir Globerson, Naftali Tishby, and Yair Weiss. Information bottleneck for gaussian variables. *Journal of machine learning research*, 6(Jan):165–188, 2005.

- [Chi02] David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.
- [Cho16a] Francois Chollet. *Keras CIFAR CNN*, 2016. bit.ly/2mVKR3d.
- [Cho16b] Francois Chollet. *Keras MNIST CNN*, 2016. bit.ly/2nKiqJv.
- [Cho19] FranÃšois Chollet. On the measure of intelligence, 2019.
- [CJM⁺19] Ryan John Cubero, Junghyo Jo, Matteo Marsili, Yasser Roudi, and Juyong Song. Statistical criticality arises in most informative representations. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(6):063402, 2019.
- [CJS⁺18] David Capper, David TW Jones, Martin Sill, Volker Hovestadt, Daniel Schrimpf, Dominik Sturm, Christian Koelsche, Felix Sahm, Lukas Chavez, David E Reuss, et al. Dna methylation-based classification of central nervous system tumours. *Nature*, 555(7697):469, 2018.
- [CLB19] David G Clark, Jesse A Livezey, and Kristofer E Bouchard. Unsupervised discovery of temporal structure in noisy data with dynamical components analysis. *arXiv preprint arXiv:1905.09944*, 2019.
- [CLRS09] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.
- [CM17] Juan Carrasquilla and Roger G Melko. Machine learning phases of matter. *Nature Physics*, 13(5):431, 2017.
- [CMT16] Matthew Chalk, Olivier Marre, and Gasper Tkacik. Relevant sparse codes with variational information bottleneck. In *Advances in Neural Information Processing Systems*, pages 1957–1965, 2016.
- [Col15] Andrew M Colman. *A dictionary of psychology*. Oxford Quick Reference, 2015.
- [CRBD18] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.
- [CSSM15] Marc Claesen, Frank De Smet, Johan A.K. Suykens, and Bart De Moor. A robust ensemble approach to learn from positive and unlabeled data using {SVM} base models. *Neurocomputing*, 160:73 – 84, 2015.
- [CTMA19] Juan Carrasquilla, Giacomo Torlai, Roger G Melko, and Leandro Aolita. Reconstructing quantum states with generative models. *Nature Machine Intelligence*, 1(3):155, 2019.
- [CUTT16] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

- [CV99] Olivier Chapelle and Vladimir Vapnik. Model selection for support vector machines. In *Proc. of 12th NIPS*, pages 230–236, Cambridge, MA, USA, 1999.
- [CZM⁺18] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.
- [DCB06] Mingzhou Ding, Yonghong Chen, and Steven L Bressler. Granger causality: basic theory and application to neuroscience. *Handbook of time series analysis: recent theoretical developments and applications*, pages 437–460, 2006.
- [DCL08] Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 240–247. ACM, 2008.
- [DG06a] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.
- [DG06b] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proc. of 23rd ICML*, pages 233–240, NYC, NY, USA, 2006. ACM.
- [dHJL19] Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *arXiv preprint arXiv:1905.11979*, 2019.
- [DKPR87] Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216 – 222, 1987.
- [DLT07] Sašo Džeroski, Pat Langley, and Ljupčo Todorovski. *Computational Discovery of Scientific Knowledge*, pages 1–14. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
- [DMAT13] Eyal Dechter, Jonathan Malmaud, Ryan P Adams, and Joshua B Tenenbaum. Bootstrap learning via modular concept discovery. In *IJCAI*, pages 1302–1309, 2013.
- [DN15] Bryan C Daniels and Ilya Nemenman. Automated adaptive inference of phenomenological dynamical models. *Nature communications*, 6:8133, 2015.
- [DSC⁺16] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.
- [DT95] Saso Dzeroski and Ljupco Todorovski. Discovering dynamics: From inductive logic programming to machine discovery. *Journal of Intelligent Information Systems*, 4(1):89–108, Jan 1995.

- [DUB⁺17] Jacob Devlin, Jonathan Uesato, Surya Bhupatiraju, Rishabh Singh, Abdelrahman Mohamed, and Pushmeet Kohli. Robustfill: Neural program learning under noisy i/o. *arXiv preprint arXiv:1703.07469*, 2017.
- [EC98] Elza Erkip and Thomas M Cover. The efficiency of investment information. *IEEE Transactions on Information Theory*, 44(3):1026–1040, 1998.
- [EHJ⁺04] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.
- [EN08] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proc. of 14th KDD*, pages 213–220, NYC, NY, USA, 2008. ACM.
- [ES16] Harrison Edwards and Amos Storkey. Towards a neural statistician. *arXiv preprint arXiv:1606.02185*, 2016.
- [ESLT15] Kevin Ellis, Armando Solar-Lezama, and Josh Tenenbaum. Unsupervised learning by program synthesis. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 973–981. Curran Associates, Inc., 2015.
- [EY36] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.
- [FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.
- [Fis18] Ian Fischer. The conditional entropy bottleneck, 2018.
- [FKPW17] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3601–3610. Curran Associates, Inc., 2017.
- [FS97] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.
- [Für99] Johannes Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, 1999.

- [Geb41] Hans Gebelein. Das statistische problem der korrelation als variations- und eigenwertproblem und sein zusammenhang mit der ausgleichsrechnung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 21(6):364–379, 1941.
- [GHY00] Clive WJ Granger, Bwo-Nung Huangb, and Chin-Wei Yang. A bivariate causality between stock prices and exchange rates: evidence from recent asianflu. *The Quarterly Review of Economics and Finance*, 40(3):337–354, 2000.
- [GIS⁺19] Anirudh Goyal, Riashat Islam, Daniel Strouse, Zafarali Ahmed, Matthew Botvinick, Hugo Larochelle, Sergey Levine, and Yoshua Bengio. Infobot: Transfer and exploration via the information bottleneck. *arXiv preprint arXiv:1901.10902*, 2019.
- [GMP05] Peter D Grünwald, In Jae Myung, and Mark A Pitt. *Advances in minimum description length: Theory and applications*. MIT press, 2005.
- [GPAM⁺14] 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.
- [Gra69] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society*, pages 424–438, 1969.
- [Gra80] Clive WJ Granger. Testing for causality: a personal viewpoint. *Journal of Economic Dynamics and control*, 2:329–352, 1980.
- [GS⁺00] Izrail Moiseevitch Gelfand, Richard A Silverman, et al. *Calculus of variations*. Courier Corporation, 2000.
- [GSD⁺18] Katja Grace, John Salvatier, Allan Dafoe, Baobao Zhang, and Owain Evans. When will ai exceed human performance? evidence from ai experts. *Journal of Artificial Intelligence Research*, 62:729–754, 2018.
- [GSR⁺17a] Dibya Ghosh, Avi Singh, Aravind Rajeswaran, Vikash Kumar, and Sergey Levine. Divide-and-conquer reinforcement learning. *arXiv preprint arXiv:1711.09874*, 2017.
- [GSR⁺17b] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*, 2017.
- [GV04] Peter Grunwald and Paul Vitányi. Shannon information and kolmogorov complexity. *arXiv preprint cs/0410002*, 2004.

- [GVW⁺16] Nicholas Guttenberg, Nathaniel Virgo, Olaf Witkowski, Hidetoshi Aoki, and Ryota Kanai. Permutation-equivariant neural networks applied to dynamics prediction. *arXiv preprint arXiv:1612.04530*, 2016.
- [GWD14] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.
- [GZ87] Richard L Gregory and Oliver Louis Zangwill. *The Oxford companion to the mind*. Oxford university press, 1987.
- [HB12] Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13(Aug):2409–2464, 2012.
- [HD13] Naftali Harris and Mathias Drton. P_c algorithm for nonparanormal graphical models. *The Journal of Machine Learning Research*, 14(1):3365–3383, 2013.
- [HFW08] Kathryn Hempstalk, Eibe Frank, and Ian H. Witten. One-class classification by combining density and class probability estimation. In *Proc. of ECML-PKDD*, pages 505–519, Berlin, Heidelberg, 2008. Springer-Verlag.
- [Hir35] Hermann O Hirschfeld. A connection between correlation and contingency. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 31, pages 520–524. Cambridge University Press, 1935.
- [HJ94] Craig Hiemstra and Jonathan D Jones. Testing for linear and nonlinear granger causality in the stock price-volume relation. *The Journal of Finance*, 49(5):1639–1664, 1994.
- [HMD15] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.
- [HMP⁺17] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.
- [Hor91] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.
- [Hos17] Yedid Hoshen. Vain: Attentional multi-agent predictive modeling. In *Advances in Neural Information Processing Systems*, pages 2701–2711, 2017.
- [Hot36] Harold Hotelling. Relation between two sets of variates. *Biometrika*, 28(3-4):321–377, 1936.
- [Hot92] Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer, 1992.

- [HPTD15] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.
- [HS93] Babak Hassibi and David G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 164–171. Morgan-Kaufmann, 1993.
- [Hut00] Marcus Hutter. A theory of universal artificial intelligence based on algorithmic complexity. *arXiv preprint cs/0004001*, 2000.
- [HvC93] Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, pages 5–13, New York, NY, USA, 1993. ACM.
- [HZRS16a] 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.
- [HZRS16b] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [IMW⁺18] Raban Iten, Tony Metger, Henrik Wilming, Lídia Del Rio, and Renato Renner. Discovering physical concepts with neural networks. *arXiv preprint arXiv:1807.10300*, 2018.
- [JBGW⁺13] Dominik Janzing, David Balduzzi, Moritz Grosse-Wentrup, Bernhard Schölkopf, et al. Quantifying causal influences. *The Annals of Statistics*, 41(5):2324–2358, 2013.
- [JKL⁺17] Michał Januszewski, Jörgen Kornfeld, Peter H Li, Art Pope, Tim Blakely, Larry Lindsey, Jeremy B Maitin-Shepard, Mike Tyka, Winfried Denk, and Viren Jain. High-precision automated reconstruction of neurons with flood-filling networks. *bioRxiv*, page 200675, 2017.
- [KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [KB15] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.
- [KDV16] Viktoriya Krakovna and Finale Doshi-Velez. Increasing the interpretability of recurrent neural networks using hidden markov models. *arXiv preprint arXiv:1606.05320*, 2016.

- [KdW05] Il Yong Kim and Oliver L de Weck. Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. *Structural and multidisciplinary optimization*, 29(2):149–158, 2005.
- [KGK⁺17] Hyeji Kim, Weihao Gao, Sreeram Kannan, Sewoong Oh, and Pramod Viswanath. Discovering potential correlations via hypercontractivity. In *Advances in Neural Information Processing Systems*, pages 4577–4587, 2017.
- [KGV83] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.
- [KH09] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.
- [KNH] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).
- [KNH14] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: <http://www.cs.toronto.edu/kriz/cifar.html>*, 55, 2014.
- [Kol63] Andrei N Kolmogorov. On tables of random numbers. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 369–376, 1963.
- [KPR⁺17] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.
- [KS94] Scott Kirkpatrick and Bart Selman. Critical behavior in the satisfiability of random boolean expressions. *Science*, 264(5163):1297–1301, 1994.
- [KSG04] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.
- [KTVK19] Artemy Kolchinsky, Brendan D Tracey, and Steven Van Kuyk. Caveats for information bottleneck in deterministic scenarios. *ICLR*, 2019.
- [Kur00] Ray Kurzweil. *The age of spiritual machines: When computers exceed human intelligence*. Penguin, 2000.
- [KW13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [KY14] Brian M Kurkoski and Hideki Yagi. Quantization of binary-input discrete memoryless channels. *IEEE Transactions on Information Theory*, 60(8):4544–4552, 2014.

- [KZS15] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2, 2015.
- [LA15] Pat Langley and Adam Arvay. Heuristic induction of rate-based process models. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [LALR09] Aurélie C Lozano, Naoki Abe, Yan Liu, and Saharon Rosset. Grouped graphical granger modeling for gene expression regulatory networks discovery. *Bioinformatics*, 25(12):i110–i118, 2009.
- [Lan81] Pat Langley. Data-driven discovery of physical laws. *Cognitive Science*, 5(1):31 – 54, 1981.
- [LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015.
- [LC10] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.
- [LCB10] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *AT&T Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, 2:18, 2010.
- [LCH⁺06] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.
- [LD94] Nada Lavrac and Saso Dzeroski. Inductive logic programming. In *WLP*, pages 146–160. Springer, 1994.
- [LDL⁺03] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. Building text classifiers using positive and unlabeled examples. In *Proc. of 3rd ICDM*, pages 179–, Washington, DC, USA, 2003. IEEE Computer Society.
- [LF92] D Lenat and E Feigenbaum. On the thresholds of knowledge. *Foundations of Artificial Intelligence, MIT Press, Cambridge, MA*, pages 185–250, 1992.
- [LGBS03] Pat Langley, Dileep George, Stephen Bay, and Kazumi Saito. Robust induction of process models from time-series data. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, pages 432–439. AAAI Press, 2003.
- [LH07a] Shane Legg and Marcus Hutter. Universal intelligence: A definition of machine intelligence. *Minds and machines*, 17(4):391–444, 2007.

- [LH⁺07b] Shane Legg, Marcus Hutter, et al. A collection of definitions of intelligence. *Frontiers in Artificial Intelligence and applications*, 157:17, 2007.
- [LH18] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, Dec 2018.
- [LJK10] Percy Liang, Michael I Jordan, and Dan Klein. Learning programs: A hierarchical bayesian approach. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 639–646, 2010.
- [LK18] Christopher Lam and David Kipping. A machine learns to predict the stability of circumbinary planets. *Monthly Notices of the Royal Astronomical Society*, 476(4):5692–5697, 2018.
- [LL03] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *Proc. of 20th ICML*, volume 1, pages 448–455, 12 2003.
- [LML⁺10] Nicholas D Lane, Emiliano Miluzzo, Hong Lu, Daniel Peebles, Tanzeem Choudhury, and Andrew T Campbell. A survey of mobile phone sensing. *IEEE Communications*, 48(9), 2010.
- [LPR17] David Lopez-Paz and Marc' Aurelio Ranzato. Gradient episodic memory for continual learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6467–6476. Curran Associates, Inc., 2017.
- [LPZ08] Joseph T Lizier, Mikhail Prokopenko, and Albert Y Zomaya. Local information transfer as a spatiotemporal filter for complex systems. *Physical Review E*, 77(2):026110, 2008.
- [LT16a] Henry W Lin and Max Tegmark. Criticality in formal languages and statistical physics. *arXiv preprint arXiv:1606.06737*, 2016.
- [LT16b] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2016.
- [LT16c] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(3):447–461, March 2016.
- [LUTG17] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017.
- [LZ89] Pat Langley and Jan M. Zytkow. Data-driven approaches to empirical discovery. *Artificial Intelligence*, 40(1):283 – 312, 1989.

- [LZCL17] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few shot learning. *arXiv preprint arXiv:1707.09835*, 2017.
- [MB16] Vincent C Müller and Nick Bostrom. Future progress in artificial intelligence: A survey of expert opinion. In *Fundamental issues of artificial intelligence*, pages 555–572. Springer, 2016.
- [MCM86] S Ryszard Michalski, G Jaime Carbonell, and M Tom Mitchell. *ML an AI Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1986.
- [MJV⁺12] Aditya Krishna Menon, Xiaoqian Jiang, Shankar Vembu, Charles Elkan, and Lucila Ohno-Machado. Predicting accurate probabilities with a ranking loss. *CoRR*, abs/1206.4661, 2012.
- [MKH93] M. M. Moya, M. W. Koch, and L. D. Hostetler. One-class classifier networks for target recognition applications. *NASA STI/Recon Technical Report N*, 93, 1993.
- [MKS⁺15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidje land, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.
- [MLC16] Henok Mengistu, Joel Lehman, and Jeff Clune. Evolvability search:directly selecting for evolvability in order to study and produce it. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, pages 141–148. ACM, 2016.
- [MPS08a] Daniele Marinazzo, Mario Pellicoro, and Sebastiano Stramaglia. Kernel granger causality and the analysis of dynamical networks. *Physical review E*, 77(5):056215, 2008.
- [MPS08b] Daniele Marinazzo, Mario Pellicoro, and Sebastiano Stramaglia. Kernel method for nonlinear granger causality. *Physical review letters*, 100(14):144103, 2008.
- [Mug91] Stephen Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, Feb 1991.
- [MV14] F. Mordelet and J. P. Vert. A bagging svm to learn from positive and unlabeled examples. *Pattern Recogn. Lett.*, 37:201–209, February 2014.
- [MVDGB08] Lukas Meier, Sara Van De Geer, and Peter Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.
- [MY02] Larry M. Manevitz and Malik Yousef. One-class svms for document classification. *JMLR*, 2:139–154, March 2002.

- [Nak99] Hideyuki Nakashima. Ai as complex information processing. *Minds and machines*, 9(1):57–80, 1999.
- [NCB08] Guilherme Neves, Sam F Cooke, and Tim VP Bliss. Synaptic plasticity, memory and the hippocampus: a neural network approach to causality. *Nature Reviews Neuroscience*, 9(1):65, 2008.
- [NDRT13a] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.
- [NDRT13b] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Adv. in NIPS 26*, pages 1196–1204. Curran Associates, Inc., 2013.
- [NG00] Kamal Nigam and Rayid Ghani. Understanding the behavior of co-training. In *KDD Workshop*, 2000.
- [NHM⁺18] Preetam Nandy, Alain Hauser, Marloes H Maathuis, et al. High-dimensional consistency in score-based and hybrid structure learning. *The Annals of Statistics*, 46(6A):3151–3183, 2018.
- [NLBT17] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.
- [NM92] D. K. Naik and R. J. Mammone. Meta-neural networks that learn by learning. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, volume 1, pages 437–442 vol.1, Jun 1992.
- [NOPF10] David F. Nettleton, Albert Orriols-Puig, and Albert Fornells. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review*, 33(4):275–306, 2010.
- [NWC17] Curtis G Northcutt, Tailin Wu, and Isaac L Chuang. Learning with confident examples: Rank pruning for robust classification with noisy labels. *arXiv preprint arXiv:1705.01936*, 2017.
- [OLV18] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- [P⁺09] Judea Pearl et al. Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146, 2009.
- [Pap85] A Papoulis. Probability, random variables and stochastic processes. 1985.
- [PCI10] Gabriele Paolacci, Jesse Chandler, and Panagiotis G. Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision Making*, 5(5):411–419, 2010.

- [Pea01] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.
- [Pea02] Judea Pearl. Causality: models, reasoning, and inference. *IIE Transactions*, 34(6):583–589, 2002.
- [Pea09] Judea Pearl. *Causality*. Cambridge university press, 2009.
- [Phy] PhysioNet. Physionet data bank.
- [Pia05] Jean Piaget. *The psychology of intelligence*. Routledge, 2005.
- [PJS17] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. MIT press, 2017.
- [PKT⁺18] Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, and Sergey Levine. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. *arXiv preprint arXiv:1810.00821*, 2018.
- [PMS⁺16] Emilio Parisotto, Abdel-rahman Mohamed, Rishabh Singh, Lihong Li, Dengyong Zhou, and Pushmeet Kohli. Neuro-symbolic program synthesis. *arXiv preprint arXiv:1611.01855*, 2016.
- [PSJ⁺18] John Peurifoy, Yichen Shen, Li Jing, Yi Yang, Fidel Cano-Renteria, Brendan G DeLacy, John D Joannopoulos, Max Tegmark, and Marin Soljačić. Nanophotonic particle simulation and inverse design using artificial neural networks. *Science advances*, 4(6):eaar4206, 2018.
- [PSST⁺99] John Platt, Bernhard Schölkopf, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating support of a high dimensional distribution. Technical report, MSR, 1999.
- [PW17] Yury Polyanskiy and Yihong Wu. Strong data-processing inequalities for channels and bayesian networks. In *Convexity and Concentration*, pages 211–249. Springer, 2017.
- [QKKG02] R Quian Quiroga, A Kraskov, T Kreuz, and Peter Grassberger. Performance of different synchronization measures in real data: a case study on electroencephalographic signals. *Physical Review E*, 65(4):041903, 2002.
- [Qui] Rodrigo Quian Quiroga. The dataset can be downloaded from.
- [RDF15] Scott Reed and Nando De Freitas. Neural programmer-interpreters. *arXiv preprint arXiv:1511.06279*, 2015.
- [RDT15] Stuart Russell, Daniel Dewey, and Max Tegmark. Research priorities for robust and beneficial artificial intelligence. *Ai Magazine*, 36(4):105–114, 2015.

- [Rén59] Alfréd Rényi. On measures of dependence. *Acta mathematica hungarica*, 10(3-4):441–451, 1959.
- [RI96] Yuval Raviv and Nathan Intrator. Bootstrapping with noise: An effective regularization technique. *Connection Science*, 8(3-4):355–372, 1996.
- [Ris78] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465 – 471, 1978.
- [Ris83] Jorma Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11(2):416–431, 1983.
- [RJJ⁺18] R.Evans, J.Jumper, J.Kirkpatrick, L.Sifre, T.F.G.Green, C.Qin, A.Zidek, A.Nelson, A.Bridgland, H.Penedones, S.Petersen, K.Simonyan, D.T.Jones, K.Kavukcuoglu, D.Hassabis, and A.W.Senior. De novo structure prediction with deep-learning based scoring, 2018.
- [RL16] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.
- [RLA⁺15] Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR*, 2015.
- [RMS⁺17] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041*, 2017.
- [RR12] Mélanie Rey and Volker Roth. Meta-gaussian information bottleneck. In *Advances in Neural Information Processing Systems*, pages 1916–1924, 2012.
- [RT18] David Rolnick and Max Tegmark. The power of deeper networks for expressing natural functions. In *International Conference on Learning Representations*, 2018.
- [Rus19] Stuart Russell. *Human Compatible: Artificial Intelligence and the Problem of Control*. Viking, 2019.
- [RV18] Danilo Jimenez Rezende and Fabio Viola. Taming VAEs. *arXiv preprint arXiv:1810.00597*, 2018.
- [SBB15] Anil K Seth, Adam B Barrett, and Lionel Barnett. Granger causality analysis in neuroscience and neuroimaging. *Journal of Neuroscience*, 35(8):3293–3297, 2015.
- [SBB⁺16] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of

Proceedings of Machine Learning Research, pages 1842–1850, New York, New York, USA, 20–22 Jun 2016. PMLR.

- [SBH13] Clayton Scott, Gilles Blanchard, and Gregory Handy. Classification with asymmetric label noise: Consistency and maximal denoising. In *COLT*, pages 489–511, 2013.
- [Sch87] Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta... hook*. PhD thesis, Technische Universität München, 1987.
- [Sch91] Roger C Schank. Where’s the ai? *AI magazine*, 12(4):38–38, 1991.
- [Sch92] Erwin Schrödinger. *What is life?: With mind and matter and autobiographical sketches*. Cambridge University Press, 1992.
- [Sch00] Thomas Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461, 2000.
- [SCHU17] Simone Scardapane, Danilo Comminiello, Amir Hussain, and Aurelio Uncini. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81–89, 2017.
- [Sco15] Clayton Scott. A rate of convergence for mixture proportion estimation, with application to learning from noisy labels. *JMLR*, 38:838–846, 2015.
- [SGL⁺19] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.
- [SGS⁺00] Peter Spirtes, Clark N Glymour, Richard Scheines, David Heckerman, Christopher Meek, Gregory Cooper, and Thomas Richardson. *Causation, prediction, and search*. MIT press, 2000.
- [Sha48a] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.
- [Sha48b] Claude Elwood Shannon. A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27:379–423, 1948.
- [SHS01] Kenji Suzuki, Isao Horiba, and Noboru Sugie. A simple neural network pruning algorithm with application to filter synthesis. *Neural Processing Letters*, 13(1):43–53, 2001.
- [Sim03] DK Simonton. An interview with dr. simonton. *Human intelligence: Historical influences, current controversies, teaching resources*. <http://www.indiana.edu/intell>, 2003.

- [SKCK17] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. PixelCNN++: A PixelCNN Implementation with Discretized Logistic Mixture Likelihood and Other Modifications. In *ICLR*, 2017.
- [SL09] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.
- [sl16] scikit learn. *LogisticRegression Class at scikit-learn*, 2016.
- [Sol64] Ray J Solomonoff. A formal theory of inductive inference. part i. *Information and control*, 7(1):1–22, 1964.
- [SQL12] Vikas Sindhwani, Minh Ha Quang, and Aurélie C Lozano. Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and granger causality. *arXiv preprint arXiv:1210.4792*, 2012.
- [SS17a] DJ Strouse and David J Schwab. The deterministic information bottleneck. *Neural computation*, 29(6):1611–1630, 2017.
- [SS17b] DJ Strouse and David J Schwab. The information bottleneck and geometric clustering. *arXiv preprint arXiv:1712.09657*, 2017.
- [SS19] DJ Strouse and David J Schwab. The information bottleneck and geometric clustering. *Neural computation*, 31(3):596–612, 2019.
- [SSK12] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density Ratio Estimation in ML*. Cambridge University Press, New York, NY, USA, 1st edition, 2012.
- [SST10] Ohad Shamir, Sivan Sabato, and Naftali Tishby. Learning and generalization with the information bottleneck. *Theoretical Computer Science*, 411(29–30):2696–2711, 2010.
- [SsWF15] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc., 2015.
- [SSZ17] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.
- [Sti17] Susanne Still. Thermodynamic cost and benefit of data representations. *arXiv preprint arXiv:1705.00612*, 2017.
- [SW89] James H Stock and Mark W Watson. Interpreting the evidence on money-income causality. *Journal of Econometrics*, 40(1):161–181, 1989.

- [TCF⁺18] Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily Fox. Neural granger causality for nonlinear time series. *arXiv preprint arXiv:1802.05842*, 2018.
- [Teg17] Max Tegmark. *Life 3.0: Being human in the age of artificial intelligence*. Knopf, 2017.
- [Teg19] Max Tegmark. Optimal latent representations: Distilling mutual information into principal pairs. *arXiv preprint arXiv:1902.03364*, 2019.
- [TGM⁺18] Jayne Thompson, Andrew JP Garner, John R Mahoney, James P Crutchfield, Vlatko Vedral, and Mile Gu. Causal asymmetry in a quantum world. *Physical Review X*, 8(3):031013, 2018.
- [Tib96] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- [Tis18] Naftali Tishby. Lecture: the information theory of deep neural networks: the statistical physics aspects. <https://www.perimeterinstitute.ca/videos/information-theory-deep-neural-networks-statistical-physics-aspects> 2018.
- [TMBS19] Andrew Tan, Leenoy Meshulam, William Bialek, and David Schwab. The renormalization group and information bottleneck: a unified framework. *Bulletin of the American Physical Society*, 2019.
- [TP12] Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.
- [TPB00] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.
- [TW19] Max Tegmark and Tailin Wu. Pareto-optimal data compression for binary classification tasks. *arXiv preprint arXiv:1908.08961*, 2019.
- [UT19] Silviu-Marian Udrescu and Max Tegmark. Ai feynman: a physics-inspired method for symbolic regression. *arXiv preprint arXiv:1905.11481*, 2019.
- [VBL⁺16] Oriol Vinyals, Charles Blundell, Tim Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3630–3638. Curran Associates, Inc., 2016.
- [vdOKE⁺16] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional Image Generation with Pixel-CNN Decoders. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and

R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4790–4798. Curran Associates, Inc., 2016.

- [VLBM08] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.
- [VNLH17] Evert PL Van Nieuwenburg, Ye-Hua Liu, and Sebastian D Huber. Learning phase transitions by confusion. *Nature Physics*, 13(5):435, 2017.
- [vSCGS18] Sjoerd van Steenkiste, Michael Chang, Klaus Greff, and Jürgen Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. *arXiv preprint arXiv:1802.10353*, 2018.
- [VV04] Nikolai K Vereshchagin and Paul MB Vitányi. Kolmogorov’s structure functions and model selection. *IEEE Transactions on Information Theory*, 50(12):3265–3290, 2004.
- [Wan95] Pei Wang. On the working definition of intelligence. *Center for Research on Concepts and Cognition CRCC, Indiana University*, 1995.
- [Wan16] Lei Wang. Discovering phase transitions with unsupervised learning. *Physical Review B*, 94(19):195105, 2016.
- [WC09] Halbert White and Karim Chalak. Settable systems: an extension of pearl’s causal model with optimization, equilibrium, and learning. *Journal of Machine Learning Research*, 10(Aug):1759–1799, 2009.
- [WCL11] Halbert White, Karim Chalak, and Xun Lu. Linking granger causality and the pearl causal model with settable systems. In *NIPS Mini-Symposium on Causality in Time Series*, pages 1–29, 2011.
- [WFCT19a] Tailin Wu, Ian Fischer, Isaac Chuang, and Max Tegmark. Learnability for the information bottleneck. *arXiv preprint arXiv:1907.07331*, 2019.
- [WFCT19b] Tailin Wu, Ian Fischer, Isaac Chuang, and Max Tegmark. Learnability for the information bottleneck. *Entropy*, 21(3):924, 2019.
- [WKNT⁺16] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.
- [WL10] Halbert White and Xun Lu. Granger causality and dynamic structural systems. *Journal of Financial Econometrics*, 8(2):193–243, 2010.

- [WLK⁺17] Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 153–164. Curran Associates, Inc., 2017.
- [WZW⁺17a] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4539–4547. Curran Associates, Inc., 2017.
- [WZW⁺17b] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4539–4547. Curran Associates, Inc., 2017.
- [XRV17] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- [XXY⁺15a] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.
- [XXY⁺15b] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *CVPR*, 2015.
- [Yai19] Sho Yaida. Fluctuation-dissipation relations for stochastic gradient descent. In *International Conference on Learning Representations*, 2019.
- [YMJ⁺12] Tianbao Yang, Mehrdad Mahdavi, Rong Jin, Lijun Zhang, and Yang Zhou. Multiple kernel learning from noisy labels by stochastic programming. In *Proc. of 29th ICML*, pages 233–240, New York, NY, USA, 2012. ACM.
- [YSB⁺18] Ilker Yildirim, Kevin A Smith, Mario Belledonne, Jiajun Wu, and Joshua B Tenenbaum. Neurocomputational modeling of human physical scene understanding. In *2nd Conference on Cognitive Computational Neuroscience*, 2018.
- [Zeg15] Pablo Zegers. Fisher information properties. *Entropy*, 17(7):4918–4939, 2015.

- [ZH05] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.
- [ZK16] S. Zagoruyko and N. Komodakis. Wide Residual Networks. *arXiv: 1605.07146*, 2016.
- [ZL16] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.
- [ZLWT18] David Zheng, Vinson Luo, Jiajun Wu, and Joshua B Tenenbaum. Unsupervised learning of latent physical properties using perception-prediction networks. *arXiv preprint arXiv:1807.09244*, 2018.