Trade-offs in Data Memorization via Strong Data Processing Inequalities

Vitaly Feldman [1] **Guy Kornowski** [2] **Xin Lyu** [3 1]

Abstract

Recent research demonstrated that training large language models involves memorization of a significant fraction of training data. Such memorization can lead to privacy violations when training on sensitive user data and thus motivates the study of data memorization's role in learning. In this work, we develop a general approach for proving lower bounds on excess data memorization, that relies on a new connection between strong data processing inequalities and data memorization. We then demonstrate that several simple and natural binary classification problems exhibit a trade-off between the number of samples available to a learning algorithm, and the amount of information about the training data that a learning algorithm needs to memorize to be accurate. In particular, $\Omega(d)$ bits of information about the training data need to be memorized when $O(1)$ d-dimensional examples are available, which then decays as the number of examples grows at a problem-specific rate. Further, our lower bounds are generally matched (up to logarithmic factors) by simple learning algorithms. We also extend our lower bounds to more general mixture-of-clusters models. Our definitions and results build on the work of Brown et al. (2021) and address several limitations of the lower bounds in their work.

1. Introduction

The machine learning (ML) methodology is traditionally thought of as constructing a model by extracting patterns in the training data. Theoretical understanding of machine learning focuses on understanding how to ensure that the constructed model generalizes well from the training data to the unseen instances. In this context, memorization of training data is typically thought of as antithetical to generalization. Yet, it has been empirically demonstrated that a variety of modern LLMs memorize a significant portion of training data (Carlini et al., 2021; 2023; Nasr et al., 2025). Specifically, (Nasr et al., 2025) demonstrate an attack allowing them to estimate that at least 0.852% of the training data used by ChatGPT (`gpt-3.5-turbo` used in production by OpenAI as of 2023) can be extracted from the model. Importantly, this includes information such as personal addresses and URLs that appears to be both highly sensitive and not particularly relevant to the task of modeling language. In many applications training data includes either personally sensitive information or copyrighted works. This makes such data memorization highly concerning and motivates the research into the role of data memorization in learning.

Explicit data memorization is known to be a crucial part of some learning algorithm, most notably those based on the nearest neighbor classifier. Further, a significant number of classical and modern works establish theoretical generalization guarantees for such methods (e.g. Cover & Hart, 1967; Biau & Devroye, 2015). It is less clear how data memorization emerges when training NNs and how the resulting models encode training data (Radhakrishnan et al., 2020; Zhang et al., 2020). However, in this work we focus not on the mechanics of memorization by specific algorithms, but on the question of whether data memorization is necessary for solving natural learning problems, as opposed to just being an artifact of the choice of the learning algorithm.

This question was first addressed by (Brown et al., 2021), who proposed to measure "irrelevant" training data memorization as the mutual information between the model and the dataset $I(\mathcal{A}(X_{1:n}); X_{1:n})$,[1] where $\mathcal{A}$ is the learning algorithm and $X_{1:n}$ is the dataset which consists of n i.i.d. samples from a data distribution. For this notion, they demonstrated existence of a simple multi-class classification problem over $\{0,1\}^d$, where each accurate learner needs to memorize a constant fraction of all training data, namely satisfies $I(\mathcal{A}(X_{1:n}); X_{1:n}) = \Omega(nd)$.

[1]Apple. [2]Weizmann Institute of Science, research done while at Apple. [3]UC Berkeley. Correspondence to: Guy Kornowski <guy.kornowski@weizmann.ac.il>.

Published at ICML 2025 Workshop on the Impact of Memorization on Trustworthy Foundation Models, Vancouver, Canada. PMLR 267, 2025.

[1]A crucial step in the study of data memorization is capturing the intuitive notion using a formal definition. We discuss some alternative approaches in Section A. We also note that the measure $I(\mathcal{A}(X_{1:n}); X_{1:n})$ was previously used in the context of generalization and we describe work done in this context in Section A.

The results of (Brown et al., 2021) rely on two components, which we briefly recall to motivate our work. The first is the focus on the accuracy for classes that only have a *single* example present in the dataset (referred to as singletons). This is motivated by the "long-tail" view of the data distribution proposed in (Feldman, 2020), where it was shown that for data distributions that are long-tailed mixtures of clusters, the accuracy of the learning algorithm on the tail of the data distribution is determined by the accuracy on singletons. The second component is a memorization lower bound for a cluster identification problem, when given only a single example of that class. Each cluster in (Brown et al., 2021) is distributed uniformly over examples satisfying a $\widetilde{O}(\sqrt{d})$-sparse boolean conjunction, and the clusters are sufficiently different so that each cluster can be accurately classified using a single example. However, as shown therein, doing so requires $\Omega(d)$ bits of information about the single example, most of which is "irrelevant" to the learning problem. Overall, this suggests that data memorization is necessary for learning in high-dimensional Boolean settings with just a single relevant sample.

In this work we aim to develop a more general understanding of data memorization in learning. In particular, we address two specific limitations of the lower bounds in (Brown et al., 2021). The key limitation is that lower bounds in (Brown et al., 2021) are tailored to a specific sparse Boolean clustering problem in which clusters are defined by uniform distributions over Boolean conjunctions. Such data distribution are not directly related to the real-valued data representations typically manipulated by neural networks. Thus we aim to develop techniques that apply to large classes of data distribution that include natural data distributions over $\mathbb{R}^d$.

The second limitation is the fragility of the lower bound: it applies only to a single example (per cluster). Empirically, it is observed that gathering more data can mitigate memorization, eventually allowing models to forget specific samples (Jagielski et al., 2023). Thus going beyond a single example and understanding the trade-off between the number of available examples and memorization for a given data distribution is an important question.

1.1. Our Contribution

In this work, we develop a general technique for proving lower bounds on data memorization. Our technique focuses on simple binary classification problems in which the goal is to distinguish points coming from a "cluster" from those coming from some fixed "null" distribution given a small number of examples. We argue that a complex learning problem over a natural data distribution implicitly involves solving many such classification problems (see Section 1.2 for more details).

Our technique relies on establishing a tight connection between our binary classification setting and *strong data processing* inequalities (SDPIs), an important tool in information theory dating back to (Ahlswede & Gács, 1976). As such SDPIs are known for a relatively limited number of pairs of jointly distributed random variables (referred to as *channels* in the context of SDPIs), we develop an approach based on approximate reductions that enables applying them in the context of learning from datasets.

We then use our general framework to analyze three natural problem instances: Gaussian cluster identification, Boolean cluster identification and sparse Boolean hypercube identification. The first problem is particularly simple and fundamental: classify an example as either sampled from an isotropic Gaussian around the origin or an isotropic Gaussian around a different point (sufficiently far from the origin). For the first two problems we prove an excess memorization lower bound of $\widetilde{\Omega}(d/n)$ for any learning algorithm (and, in the Boolean case, $n \lesssim \sqrt{d}$). We further show that this lower bound is tight up to logarithmic factors. The third problem is the one defined in (Brown et al., 2021), for which we show a lower bound of $\Omega(d/2^{2n})$ for $n \lesssim \log d$, and we again match with a tight upper bound.

Problem setting and excess memorization. We now describe our approach and results in more detail starting with some more formal definitions and notation. We study binary classification problems in which an algorithm $\mathcal{A}$ is given training data $X_{1:n} = X_1, \ldots, X_n \overset{iid}{\sim} \mathcal{P}_\theta$ sampled from some distribution $\mathcal{P}_\theta$ over $\mathcal{X}$ parameterized by some parameter θ. To make our lower bounds stronger, following (Feldman, 2020; Brown et al., 2021) we state them for average case problems, namely, we think of θ as itself being chosen from a known meta-distribution $\theta \sim \Psi$. For a given data distribution $\mathcal{P}_\theta$ and algorithm $\mathcal{A}$, we measure training data memorization of $\mathcal{A}$ by the mutual information between the model and the dataset drawn i.i.d. from $\mathcal{P}_\theta$:

$$\mathrm{mem}_n(\mathcal{A}, \mathcal{P}_\theta) := I\left(\mathcal{A}(X_{1:n}); X_{1:n}\right) \ ,$$

where $X_{1:n} \sim \mathcal{P}_\theta^n$. Notably, in this definition $\mathcal{P}_\theta$ is fixed, and therefore $\mathrm{mem}_n(\mathcal{A}, \mathcal{P}_\theta)$ does not count any information about the unknown data distribution θ. To emphasize this property we refer to $\mathrm{mem}_n(\mathcal{A}, \mathcal{P}_\theta)$ as *excess* data memorization. We further denote the average excess memorization for $\mathcal{A}$ on an average-case problem $\mathbf{P} = (\mathcal{P}_\theta)_{\theta\sim\Psi}$ by

$$\begin{aligned}\mathrm{mem}_n(\mathcal{A}, \mathbf{P}) &:= \mathop{\mathbb{E}}_{\theta\sim\Psi}[\mathrm{mem}_n(\mathcal{A}, \mathcal{P}_\theta)] = I\left(\mathcal{A}(X_{1:n}); X_{1:n} \mid \theta\right) \\ &= I\left(\mathcal{A}(X_{1:n}); X_{1:n}\right) - I\left(\mathcal{A}(X_{1:n}); \theta\right) \ .\end{aligned}$$

Note, again, that conditioning on θ ensures that we are not counting the information that $\mathcal{A}$ learns about θ which is necessary for learning, but rather measuring excess memorization of the dataset. Moreover, the latter equality (which holds by the chain rule for mutual information) provides an

intuitive interpretation of this quantity: it is information the model has about the training data, after subtracting the "relevant" information about θ, thus leaving the purely excess memorization.

Strong Data Processing Inequalities (SDPIs) imply memorization (Theorem C.7): We establish a direct connection between SDPIs and excess data memorization. We recall that the (regular) data processing inequality states that mutual information cannot increase as a result of post-processing, that is, in a Markov chain $A \to B \to C$, $I(A;C) \le I(B;C)$. Strong data processing inequality holds when for a pair of jointly distributed random variables (A, B), the step $A \to B$ necessarily reduces the mutual information by some factor $\rho < 1$, referred to as the SDPI constant for (A, B).

In our context, for a randomly chosen $\theta \sim \Psi$, a dataset $X_{1:n} \sim \mathcal{P}_\theta^n$, and an additional fresh test sample $X \sim \mathcal{P}_\theta$, we have a Markov chain $X \to X_{1:n} \to \mathcal{A}(X_{1:n})$ since the only information $\mathcal{A}$ has about X is through $X_{1:n}$. Thus we can deduce that $I(\mathcal{A}(X_{1:n}); X) \le \rho \cdot I(\mathcal{A}(X_{1:n}); X_{1:n})$, where ρ is the SDPI constant for $(X, X_{1:n})$. As is well-known, accurate binary prediction requires information, namely $I(\mathcal{A}(X_{1:n}); X) = \Omega(1)$ for any $\mathcal{A}$ with error that is $\le 1/3$. As a result of applying the SDPI, we get that $I(\mathcal{A}(X_{1:n}); X_{1:n}) = \Omega(1/\rho)$.

As discussed, when θ is a random variable, $I(\mathcal{A}(X_{1:n}); X_{1:n})$ also counts the information that $\mathcal{A}(X_{1:n})$ contains about θ. To obtain a lower bound on excess data memorization we need to subtract $I(\mathcal{A}(X_{1:n}); \theta)$ from $I(\mathcal{A}(X_{1:n}); X_{1:n})$. To achieve this, we consider the Markov chain $\theta \to X_{1:n} \to \mathcal{A}(X_{1:n})$ and denote by τ the SDPI constant of the pair $(\theta, X_{1:n})$. Applying the SDPI to $\mathcal{A}(X_{1:n})$ and combining it with the lower bound on $I(\mathcal{A}(X_{1:n}); X_{1:n})$ then gives us the summary of the connection between SDPIs and memorization:

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}) = \Omega\left(\frac{1-\tau_n}{\rho_n}\right) ,$$

where we emphasize the fact τ and ρ depend on n. We remark, that for $n = 1$, the first SDPI coefficient ρ_1 is related to the proof technique of (Brown et al., 2021) who reduce their learning problem to a variant of the so-called Gap-Hamming communication problem and then give an SDPI-based lower bound adapted from (Hadar et al., 2019).

Approximate SDPIs via dominating variables (Theorem C.8): Our framework reduces memorization to computation of SDPI constants for pairs $(X_{1:n}, X)$ and $(X_{1:n}, \theta)$. However, known SDPIs deal primarily with individual samples from several very specific distributions (most notably, Bernoulli and Gaussian), and not datasets that appear to be much more challenging to analyze directly. We bypass this difficulty via a notion of a *dominating* random variable for the dataset. Specifically, if there exists a random variable $Z_\theta^{\mathrm{train}}$ and post-processing Φ such that $\Phi(Z_\theta^{\mathrm{train}}) \approx X_{1:n}$ (as distributions), then it suffices to prove SDPIs for the pair $(Z_\theta^{\mathrm{train}}, X)$. For our applications, it is crucial to allow $\Phi(Z_\theta^{\mathrm{train}})$ to approximate $X_{1:n}$ and thus our reduction incorporates the effects of approximation error in both SDPIs. Our reduction also allows using a dominating variable Z_θ^{test} for the test point X, but in our applications X is simple enough and this step is not needed.

Applications: memorization trade-offs and matching upper bounds (Theorem D.1, Theorem D.3 and Theorem D.4): We apply the techniques we developed to demonstrate that several natural learning problems exhibit smooth trade-offs between excess memorization and sample size. The first problem we consider is Gaussian clustering. In this problem, negative examples are sampled from $\mathcal{N}(0_d, I_d)$, while positive examples are sampled from $\mathcal{N}(\lambda\theta, (1-\lambda^2)I_d)$ for some scale λ. Our lower bounds are for $\theta \sim \mathcal{N}(0_d, I_d)$. We pick $\lambda = \Theta(1/d^{1/4})$, ensuring that accurate learning is possible with just a single positive sample (by using either nearest neighbor or linear classifier). At the same time, our analysis demonstrates that any learning algorithm $\mathcal{A}$ for this problem that achieves non-trivial error satisfies

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}) = \Omega\left(\frac{d}{n}\right) .$$

Moreover, for this problem we show that a matching (up to log factors) upper bound can be achieved by simple learning algorithms whenever $n \le \sqrt{d}$. Hence, we overall establish that memorization can be reduced by using more data, and that is the only way to it (while maintaining accuracy).

We next consider a Boolean clustering problem. In this problem $\theta \in \{\pm 1\}^d$, negative examples are sampled from the uniform distribution over $\{\pm 1\}^d$, whereas positive examples are sampled from a product distribution over $\{\pm 1\}^d$ with mean $\lambda \cdot \theta$. Thus for θ chosen uniformly from $\{\pm 1\}^d$, a random positive example from $\mathcal{P}_\theta$ is coordinate-wise λ-correlated with θ. The problem can be thought of as a Boolean analogue of the Gaussian setting above, and we obtain nearly tight (up to a log factor) upper and lower bounds for this problem. The bounds are similar to those in the Gaussian case for $n \le \sqrt{d}$ but almost no memorization is needed when $n \ge \sqrt{d}\log d$. We note however that the analysis of the approximately dominating variable is more involved in this case and is derived from composition results in differential privacy.

Finally, we consider a sparse Boolean hypercube clustering problem introduced in (Brown et al., 2021). In this setting, the data distribution is defined by a pair $\theta = (S, Y)$, where $S \subseteq [d]$ is a subset of coordinates and $Y \in \{\pm 1\}^S$ are the

values assigned to these coordinates. In the data distribution $\mathcal{P}_\theta$ negative examples are uniform over $\{\pm 1\}^d$, whereas the positive examples are uniform over the hypercube of all the points x whose values in coordinates in S are exactly Y: namely $x \in \{\pm 1\}^d$ that satisfy the conjunction $\bigwedge_{i \in S}(x_i = Y_i)$. To sample θ we include each index in S with probability $\approx 1/\sqrt{d}$ independently at random, and then assign to each coordinate a uniformly random value in $Y \in \{\pm 1\}^S$. As noted in (Brown et al., 2021), when $n = 1$ this problem is identical to the Boolean clustering problem defined above. However, for larger n, we show it requires much less memorization. Specifically, we give a lower and upper bound (tight up to log factors) that are:

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}) = \tilde{\Theta}\left(\frac{d}{2^{2n}}\right) .$$

We remark that in these example applications, we chose parameters so as to ensure that each problem is learnable from a single positive example, but requires $\Omega(d)$ bits to be memorized to do so. Beyond this extreme case, our techniques easily extend to show lower bounds in terms of correlation between samples, which also determines the smallest n and d that would be required for the learnability.

1.2. From cluster classification to LLMs

We now briefly discuss how lower bounds for the simple classification problems we consider are related to the data memorization by LLMs. We first note that while LLMs are often used as generative models, underlying the sampler is a (soft) predictor of the next token given the preceding context. Thus an LLM is also a multiclass classifier. Second, LLMs (and many other ML models) either explicitly or implicitly rely on semantic data embeddings of the context, that is, embeddings in which semantically similar contexts are mapped to points close in Euclidean distance (or cosine similarity). In particular, nearby points are typically classified the same. As a result, when viewed in the embedding space, natural data distributions correspond to mixtures of (somewhat-disjoint) clusters of data points where points in the same cluster typically have the same label (cf. Reif et al., 2019; Cai et al., 2021; Radford et al., 2021).

As has been widely observed (and discussed in (Feldman, 2020)), for natural data distributions in many domains, the frequencies of these clusters tend to be long-tailed with a significant fraction of the entire data distribution being in low-frequency clusters. Such low frequency clusters have only few representatives in the training dataset (possibly just one). Accurate classification of a point from a low-frequency cluster requires being able to classify whether a test point belongs to the same cluster based on just a few examples of that cluster.

This shows that classifying points as belonging to some cluster or not is a subproblem that arises when learning from natural data. This raises the question of how to model such "clusters". While in practice cluster distributions will depend strongly on the representations used and may not have a simple form, one prototypical and widely studied choice is the Gaussian distribution. This distribution is known to be prevalent in natural phenomena (earning it the name "normal"). The ubiquity of mixtures of Gaussian-like distributions is also the reason for the utility of techniques such as Gaussian Mixture Models for distribution modeling (cf. Reynolds et al., 2009).

Putting these insights together, we get to the key application of our techniques: the problem of distinguishing a point from a Gaussian distribution from some null distribution given few samples, as a subproblem when the data distribution is a long-tailed mixture of Gaussian clusters.

Lower bounds for mixtures of clusters (Theorem E.5): Finally, in Section E we discuss how our techniques can be extended to a more detailed mixture-of-clusters model of data. Specifically, we consider data models based on a prior distribution over frequencies of clusters, as studied in (Feldman, 2020). The lower bound by Brown et al. (2021) is given in this model, but only clusters from which a single example was observed contribute to the memorization lower bound. We show that our more general lower bound approach extends to this mixture-of-clusters setting, with each cluster contributing to the total memorization lower bound according to the number of examples of that cluster in the training dataset. In particular, our results demonstrate a smooth trade-off in which clusters with less representatives in the training data contribute more to the excess memorization of the learned model.

2. Discussion

All in all, our proof techniques and lower bounds for specific problem instances support the following intuition for the phenomenon of data memorization observed in practice: The tail of real-world data distributions contains many subproblems for which relatively little data is available. Non-trivial accuracy on these subproblems can be achieved by exploiting all the relatively weak correlations present between points in the dataset and unseen points from the same subproblem. This however, requires memorizing (almost) all the features of the available data points. More data allows the learning algorithm to average out some of the inherent randomness (or "noise") in the features of the given examples, and thus increase the correlations with the features of an unseen point. In turn, this allows the learning algorithm to memorize fewer features of the training data on that subpopulation, specifically those with the strongest correlations.

Acknowledgements

GK is supported through an Azrieli Foundation graduate fellowship. XL is supported by a Google fellowship.

Impact Statement

This paper presents work whose goal is to develop a theory that explains and quantifies the phenomenon of data memorization in modern ML. In particular, our theoretical framework reveals trade-offs between inherent quantities of the learning problem such as correlation between different samples, cluster sizes, and the amount of excess memorization required for learning. As data memorization is a widespread phenomenon with important consequences, we believe our work has potential impact, none of which we feel must be specifically highlighted here.

References

Ahlswede, R. and Gács, P. Spreading of sets in product spaces and hypercontraction of the markov operator. *The annals of probability*, pp. 925–939, 1976.

Attias, I., Dziugaite, G. K., Haghifam, M., Livni, R., and Roy, D. M. Information complexity of stochastic convex optimization: Applications to generalization, memorization, and tracing. In *Forty-first International Conference on Machine Learning*, 2024.

Bassily, R., Moran, S., Nachum, I., Shafer, J., and Yehudayoff, A. Learners that use little information. In *Algorithmic Learning Theory*, pp. 25–55. PMLR, 2018.

Biau, G. and Devroye, L. *Lectures on the nearest neighbor method*, volume 246. Springer, 2015.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. Occam's razor. *Information processing letters*, 24 (6):377–380, 1987.

Brown, G., Bun, M., Feldman, V., Smith, A., and Talwar, K. When is memorization of irrelevant training data necessary for high-accuracy learning? In *Proceedings of the 53rd annual ACM SIGACT symposium on theory of computing*, pp. 123–132, 2021.

Bu, Y., Zou, S., and Veeravalli, V. V. Tightening mutual information-based bounds on generalization error. *IEEE Journal on Selected Areas in Information Theory*, 1(1): 121–130, 2020.

Cai, X., Huang, J., Bian, Y., and Church, K. Isotropy in the contextual embedding space: Clusters and manifolds. In *International conference on learning representations*, 2021.

Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., and Song, D. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pp. 267–284, 2019.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Carlini, N., Chien, S., Nasr, M., Song, S., Terzis, A., and Tramer, F. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914. IEEE, 2022.

Carlini, N., Hayes, J., Nasr, M., Jagielski, M., Sehwag, V., Tramer, F., Balle, B., Ippolito, D., and Wallace, E. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 5253–5270, 2023.

Cover, T. and Hart, P. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

Cover, T. M. and Thomas, J. A. *Elements of information theory*. John Wiley & Sons, 2nd edition, 1999.

Dwork, C., Rothblum, G. N., and Vadhan, S. P. Boosting and differential privacy. In *FOCS*, pp. 51–60. IEEE Computer Society, 2010.

Dwork, C., Feldman, V., Hardt, M., Pitassi, T., Reingold, O., and Roth, A. Generalization in adaptive data analysis and holdout reuse. *Advances in neural information processing systems*, 28, 2015.

Feldman, V. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 954–959, 2020.

Feldman, V. and Steinke, T. Calibrating noise to variance in adaptive data analysis. In *Conference On Learning Theory*, pp. 535–544. PMLR, 2018.

Hadar, U., Liu, J., Polyanskiy, Y., and Shayevitz, O. Communication complexity of estimating correlations. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pp. 792–803, 2019.

Jagielski, M., Thakkar, O., Tramer, F., Ippolito, D., Lee, K., Carlini, N., Wallace, E., Song, S., Thakurta, A. G., Papernot, N., et al. Measuring forgetting of memorized training examples. In *The Eleventh International Conference on Learning Representations*, 2023.

Kairouz, P., Oh, S., and Viswanath, P. The composition theorem for differential privacy. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1376–1385. JMLR.org, 2015.

Littlestone, N. and Warmuth, M. Relating data compression and learnability. 1986.

Livni, R. Information theoretic lower bounds for information theoretic upper bounds. *Advances in Neural Information Processing Systems*, 36, 2024.

Livni, R. and Moran, S. A limitation of the pac-bayes framework. *Advances in Neural Information Processing Systems*, 33:20543–20553, 2020.

McAllester, D. A. Some pac-bayesian theorems. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 230–234, 1998.

Nachum, I., Shafer, J., and Yehudayoff, A. A direct sum result for the information complexity of learning. In *Conference On Learning Theory*, pp. 1547–1568. PMLR, 2018.

Nasr, M., Carlini, N., Hayase, J., Jagielski, M., Cooper, A. F., Ippolito, D., Choquette-Choo, C. A., Wallace, E., Tramèr, F., and Lee, K. Scalable extraction of training data from aligned, production language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Polyanskiy, Y. and Wu, Y. Strong data-processing inequalities for channels and bayesian networks. In *Convexity and Concentration*, pp. 211–249. Springer, 2017.

Polyanskiy, Y. and Wu, Y. *Information theory: From coding to learning*. Cambridge university press, 2024.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Radhakrishnan, A., Belkin, M., and Uhler, C. Overparameterized neural networks implement associative memory. *Proceedings of the National Academy of Sciences*, 117 (44):27162–27170, 2020.

Raginsky, M. Strong data processing inequalities and Φ-sobolev inequalities for discrete channels. *IEEE Transactions on Information Theory*, 62(6):3355–3389, 2016.

Raginsky, M., Rakhlin, A., Tsao, M., Wu, Y., and Xu, A. Information-theoretic analysis of stability and bias of learning algorithms. In *2016 IEEE Information Theory Workshop (ITW)*, pp. 26–30. IEEE, 2016.

Reif, E., Yuan, A., Wattenberg, M., Viegas, F. B., Coenen, A., Pearce, A., and Kim, B. Visualizing and measuring the geometry of bert. *Advances in neural information processing systems*, 32, 2019.

Reynolds, D. A. et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663):3, 2009.

Russo, D. and Zou, J. How much does your data exploration overfit? controlling bias via information usage. *IEEE Transactions on Information Theory*, 66(1):302–323, 2019.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Steinke, T. and Zakynthinou, L. Reasoning about generalization via conditional mutual information. In *Conference on Learning Theory*, pp. 3437–3452. PMLR, 2020.

Vershynin, R. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Xu, A. and Raginsky, M. Information-theoretic analysis of generalization capability of learning algorithms. *Advances in neural information processing systems*, 30, 2017.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.

Zhang, C., Bengio, S., Hardt, M., Mozer, M. C., and Singer, Y. Identity crisis: Memorization and generalization under extreme overparameterization. In *International Conference on Learning Representations*, 2020.

Contents

A. Related Work

A fundamental theme in learning theory is that "simple" learning rules generalize (Blumer et al., 1987). In particular, there is a long line of work studying generalization bounds which provide various formalizations of the intuition that learners who use little information about their dataset must generalize. Classical such notions include compression schemes (Littlestone & Warmuth, 1986) and the PAC-Bayes framework (McAllester, 1998). This theme is also the basis for the more recent use of mutual information (MI) between the dataset and the output of the algorithm to derive generalization bounds. The approach was first proposed in the context of adaptive data analysis by Dwork et al. (2015), who used max-information to derive high-probability generalization bounds. Building on this approach, Russo & Zou (2019) proposed using the classical notion of MI to derive (in expectation) generalization bounds, with numerous subsequent works strengthening and applying their results (Raginsky et al., 2016; Xu & Raginsky, 2017; Feldman & Steinke, 2018; Bu et al., 2020). More recent developments in this line of work rely on the notion of *conditional mutual information* (CMI). Here, the conditioning is over a ghost sample which is different from the conditioning over the data distribution (i.e. θ) considered in this work (Steinke & Zakynthinou, 2020). The CMI roughly measures *identifiability* of the samples in the dataset given the model. It is closely related to membership inference attacks (Shokri et al., 2017; Carlini et al., 2022; Attias et al., 2024).

To demonstrate limitations of generalization methods based on MI, Livni (2024) considers the setting of stochastic convex optimization (SCO). In this setting, he proves a lower bound on the MI of learners achieving asymptotically optimal error, which scales as d/n^C for some constant C. While superficially this lower bound is similar to our result, the goal of the problem therein is to estimate an unknown d-dimensional parameter. Moreover, the coordinates of this parameter are chosen independently and thus the estimation problem requires effectively solving d independent one-dimensional problems. This is in contrast to our setting, in which the problem is binary classification, and we make no assumptions on the representation of the model. We also remark that the trade-off in (Livni, 2024) appears to be mostly an artifact of the proof, with natural algorithms achieving nearly-optimal rates requiring excess memorization of $\Omega(d)$ bits for any n. Attias et al. (2024) recently proved lower bound on CMI in the SCO setting demonstrating that CMI cannot be used to recover known generalization bounds for SCO. By the nature of the definition, the CMI is at most n and thus the lower bounds of the CMI are incomparable to the trade-offs in our work.

Several works study lower bounds on the MI in the context of distribution-independent PAC learning of threshold functions establishing lower bounds which, at best, only scale logarithmically with the description length of examples (which in our instances corresponds to the dimension) (Bassily et al., 2018; Nachum et al., 2018; Livni & Moran, 2020).

The literature on phenomena related to memorization relies on a large variety of mostly informal notions. In the context of data extraction attacks, the definitions rely on the success of specific attacks that either feed a partial prompt (Carlini et al., 2021) or examine the relative likelihood of training data under the model (Carlini et al., 2019). Such definitions are useful for analyzing the success of specific attacks but are sensitive to the learning algorithms. In particular, minor changes to the algorithm can greatly affect the measures of memorization. They also do not distinguish between memorization of data relevant to the learning problem (e.g. memorization of capitals of countries in the context of answering geographic queries) from the irrelevant one.

Another related class of definitions considers memorization resulting from fitting of noisy data points (e.g. Zhang et al., 2017). Such memorization is referred to as *label* memorization and does not, in general, require memorization of data points that we study here. The known formal definition is not information-theoretic but rather directly examines the influence of the data point on the label (Feldman, 2020). At the same time, both label memorization and excess data memorization appear to be artifacts of learning from long-tailed data distributions.

On a technical side, SDPIs have a number of important applications in machine learning (and beyond), most notably, in the context of privacy preserving and/or distributed learning and estimation. We refer the reader to the book of (Polyanskiy & Wu, 2024) for a detailed overview. These applications of SDPIs are not directly related to our use.

B. Formal Problem Setting

Notation. We abbreviate a sequence $X_1, \ldots, X_n$ by $X_{1:n}$. I_d denotes the $d \times d$ identity matrix, 0_d denotes the d-dimensional zero vector, and we occasional omit the subscript when the dimension is clear from the context. Given a finite set S, we denote by $\Delta(S)$ the set of all distributions over S, and by $\mathcal{U}(S)$ the uniform distribution over S. We denote $X \perp\!\!\!\perp Y$ when X, Y are independent random variables. $\|\cdot\|$ denotes the Euclidean norm, and $d_{\mathrm{TV}}(\cdot,\cdot)$ denotes the total variation distance (which when applied to random variables, is the distance between their corresponding distributions). For $x \in \{\pm 1\}^d$,

$\mathrm{Bin}_p(x)$ denotes the product distribution over $\{\pm 1\}^d$ in which for $Y \sim \mathrm{Bin}_p(x)$, $\Pr[Y_i \neq x_i] = 1 - \Pr[Y_i = x_i] = p$ independently for every $i \in [d]$. The mapping from x to Y is usually referred to as the *binary symmetric channel (BSC)* in the context of data processing inequalities. Given random variables X, Y, Z, we denote by $H(X)$ the entropy of X,[2] by $I(X;Y) := H(X) - H(X \mid Y)$ the mutual information between X and Y, and by $I(X;Y \mid Z) := \mathbb{E}_Z[I(X \mid Z; Y \mid Z)]$ the conditional mutual information. We denote the binary entropy function $H_2(p) := p\log_2(1/p) + (1-p)\log_2(1/(1-p))$.

Formal setting. We now formalize the problem setting we consider throughout this work, outlined in the introduction. The learning algorithm's goal is binary classification of a point drawn from a mixture distribution in which with probability $1/2$ a (positive) point is drawn from the parameter dependent "cluster" distribution $\mathcal{P}_\theta$ over $\mathcal{X}$, and with probability $1/2$ a (negative) point is drawn from a fixed "null" distribution $\mathcal{P}_0$. Formally, for a problem parameter θ, let $(X^{\text{test}}, Y^{\text{test}}) \sim \mathcal{P}_\theta^{\text{test}}$ be the mixture distribution over $\mathcal{X} \times \{0,1\}$ such that with probability $\frac{1}{2}$: $X^{\text{test}} \sim \mathcal{P}_\theta$ and $Y^{\text{test}} = 1$; otherwise with probability $\frac{1}{2}$: $X^{\text{test}} \sim \mathcal{P}_0$ and $Y^{\text{test}} = 0$. The algorithm returns a classifier $h : \mathcal{X} \to \{0,1\}$, aiming at minimizing the classification error:

$$\mathrm{err}(h) := \Pr_{(X^{\text{test}}, Y^{\text{test}}) \sim \mathcal{P}_\theta^{\text{test}}} \left[h(X^{\text{test}}) \neq Y^{\text{test}} \right] .$$

Note that $\mathcal{P}_0$ is fixed and therefore negative examples do not carry any information about the learning problem (and can be generated by the learning algorithm itself). Therefore, without loss of generality, we can assume that the training dataset given to a learning algorithm $\mathcal{A}$ consists of n positive data points $X_1, \ldots, X_n \overset{iid}{\sim} \mathcal{P}_\theta$.

In our problems, the parameter θ will be sampled from a known meta-distribution $\theta \sim \Psi$ and we denote the resulting (average-case) learning problem by $\mathbf{P} = (\mathcal{P}_\theta)_{\theta \sim \Psi}$. Further, the null distribution $\mathcal{P}_0$ will always be equal to the marginal distribution of $X \sim \mathcal{P}_\theta$ for $\theta \sim \Psi$ and thus we do not specify it explicitly. The loss of a learning algorithm $\mathcal{A}$ on the problem $\mathbf{P} = (\mathcal{P}_\theta)_{\theta \sim \Psi}$ is defined accordingly as

$$\mathrm{err}(\mathcal{A}, \mathbf{P}) := \mathop{\mathbb{E}}_{\substack{\theta \sim \Psi \\ X_1, \ldots, X_n \sim \mathcal{P}_\theta \\ h \leftarrow \mathcal{A}(X_{1:n})}} [\mathrm{err}(h)] .$$

We also recall the definition of excess data memorization for an algorithm $\mathcal{A}$ and data distribution $\mathcal{P}_\theta$ (Bassily et al., 2018; Brown et al., 2021):

$$\mathrm{mem}_n(\mathcal{A}, \mathcal{P}_\theta) := I\left(\mathcal{A}(X_{1:n}); X_{1:n}\right) ,$$

where $X_{1:n} \sim \mathcal{P}_\theta^n$. For an average case problem $\mathbf{P} = (\mathcal{P}_\theta)_{\theta \sim \Psi}$ excess memorization for $\mathcal{A}$ is defined as

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}) := \mathop{\mathbb{E}}_{\theta \sim \Psi} [\mathrm{mem}_n(\mathcal{A}, \mathcal{P}_\theta)] = I\left(\mathcal{A}(X_{1:n}); X_{1:n} \mid \theta\right) .$$

We also denote the minimal (i.e., necessary) memorization for algorithms with error of at most α by

$$\mathrm{mem}_n(\mathbf{P}, \alpha) := \inf_{\mathcal{A} \colon \mathrm{err}(\mathcal{A}, \mathbf{P}) \leq \alpha} \mathrm{mem}_n(\mathcal{A}, \mathbf{P}) ,$$

and let $\mathrm{mem}_n(\mathbf{P}) := \mathrm{mem}_n(\mathbf{P}, 1/3)$.

C. General Framework: SDPIs and Memorization

In this section, we will introduce the main machinery that allows us to derive excess memorization lower bounds via *strong data processing inequalities* (SDPIs). We start by recalling the definition of an SDPI.

Definition C.1. Given a pair of jointly distributed random variables (A, B), we say that A, B satisfy ρ-SDPI if for any M such that $A \perp\!\!\!\perp M \mid B$: $I(M;A) \leq \rho I(M;B)$.

Recall that all random variables satisfy the definition above for $\rho = 1$, which is simply the "regular" data processing inequality (DPI). A *strong* DPI refers to the case $\rho < 1$. This means that any M which is a post-processing of B (i.e., $A \perp\!\!\!\perp M \mid B$), has *strictly* less information about A than it does about B. Equivalently, any post-processing of B must have

[2] We slightly abuse information theoretic notation by using it both for discrete and continuous random variables. In the latter case, definitions are with respect to the differential entropy.

ρ^{-1}-times *more* information about B. This observation will serve as the basis of our results, and accordingly, we will aim to prove SDPIs with a small SDPI constant ρ.

SDPIs constitute a fundamental concept in information theory dating back to Ahlswede & Gács (1976), and their study remains an active area of research (Raginsky, 2016; Polyanskiy & Wu, 2017; 2024). Here we recall two canonical examples of SDPI in which the coefficient ρ results from weak correlation between the marginals.

Fact C.2 (Polyanskiy & Wu, 2024, Example 33.7+Proposition 33.11)**.** *Suppose (A, B) is a $2d$-dimensional Gaussian distribution such that marginally $A, B \sim \mathcal{N}(0_d, I_d)$ and for each coordinate $i \in [d]$: $\mathbb{E}[A_i \cdot B_i] = \sqrt{\rho}$. Then A, B satisfy the ρ-SDPI.*

Fact C.3 (Polyanskiy & Wu, 2024, Example 33.2)**.** *Suppose (A, B) are $\sqrt{\rho}$-correlated uniform Boolean vectors, namely $A \sim \mathcal{U}(\{\pm 1\}^d)$, $B = \mathrm{Bin}_{\frac{1-\sqrt{\rho}}{2}}(A)$. Then A, B satisfy the ρ-SDPI.*

In some applications, using "approximate" SDPIs appears to be crucial for achieving meaningful lower bounds, so we introduce the following approximate version of this notion.

Definition C.4. Given a pair of jointly distributed random variables (A, B), we say that A, B satisfy (ρ, δ)-approximate-SDPI, or simply (ρ, δ)-SDPI, if they can be coupled with a random variable $\tilde{B}$ such that: (1) A and $\tilde{B}$ satisfy ρ-SDPI, and (2) for every $a \in \mathrm{supp}(A)$, we have $d_{\mathrm{TV}}(\tilde{B} \mid A = a, B \mid A = a) \le \delta$.

Clearly, the definition above reduces to the standard notion of SDPI when $\delta = 0$, with $\tilde{B} = B$. Its main utility is through the following result, which quantifies how the δ-approximation results in an additive factor to the standard SDPI setting.

Proposition C.5. *Suppose (A, B) are joint random variables satisfying (ρ, δ)-SDPI. Let $\mathcal{A} : \mathcal{B} \to \mathcal{M}$ be a (possibly randomized) post-processing of B, where $\mathcal{B}$ denotes the support of B. Then*

$$I(\mathcal{A}(B); A) \le \rho I(\mathcal{A}(B); B) + 8\delta \log(|\mathcal{M}|/\delta) \, .$$

Remark C.6. Throughout the paper, we use the convention $\delta \log(|\mathcal{M}|/\delta) = 0$ when $\delta = 0, |\mathcal{M}| = \infty$.

The proof of Proposition C.5 is provided in Appendix F.1. The rest of this section is structured as follows. In Section C.1 we show how to bound the quantity of interest $\mathrm{mem}_n(\mathcal{A}, \mathbf{P})$ whenever certain SDPIs are present in the learning problem. Subsequently, in Section C.2, we will address the issue of when we should expect the required SDPIs to hold, and how to compute their corresponding coefficients via an approximate reduction.

C.1. SDPIs imply Memorization

We now present the main result that relates excess memorization to SDPIs. Recalling that the problem definition involves the distributions $\mathcal{P}_\theta$ and the parameter distribution Ψ, the next theorem shows that memorization is necessary whenever certain SDPIs hold in the learning problem.

Theorem C.7. *Let $\mathbf{P} = (\mathcal{P}_\theta)_{\theta \sim \Psi}$ be a learning problem satisfying the following:*

1. *(Data generation SDPI) The variables $(\theta, X_{1:n})$ for $\theta \sim \Psi$ and $X_{1:n} \sim \mathcal{P}_\theta^n$ satisfy (τ_n, ϵ_n)-SDPI.*
2. *(Test/train SDPI) The variables $(X, X_{1:n})$ for $\theta \sim \Psi$, $X \sim \mathcal{P}_\theta$ and $X_{1:n} \sim \mathcal{P}_\theta^n$ satisfy (ρ_n, δ_n)-SDPI.*

Then any algorithm $\mathcal{A} : \mathcal{X}^n \to \mathcal{M}$ for $\mathbf{P}$ satisfies the excess memorization bound:

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}) \ge \frac{1 - \tau_n}{\rho_n} I(\mathcal{A}(X_{1:n}); X) - \mathrm{neg}_n \, ,$$

where $\mathrm{neg}_n := 8\delta_n \frac{1-\tau_n}{\rho_n} \log(|\mathcal{M}|/\delta_n) + 8\epsilon_n \log(|\mathcal{M}|/\epsilon_n)$. *Moreover, for any $\alpha < \frac{1}{2}$:*

$$\mathrm{mem}_n(\mathbf{P}, \alpha) \ge \frac{1 - \tau_n}{\rho_n} C_\alpha - \mathrm{neg}_n \, , \quad \textit{where} \quad C_\alpha := (1 - 2\alpha) \log\left(\frac{1 - \alpha}{\alpha}\right) \, .$$

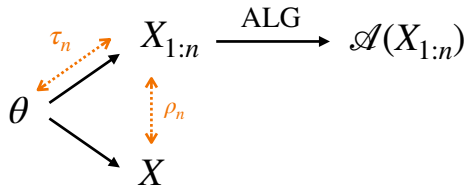

Figure 1. Illustration for Theorem C.7. Given $\theta \sim \Psi$, $X_{1:n} \sim \mathcal{P}_\theta^n$, $X \sim \mathcal{P}_\theta$, the orange arrows represent two SDPIs, which together necessitate excess memorization on the order of $(1-\tau_n)/\rho_n$.

Proof of Theorem C.7. Note that the data generation SDPI implies by Proposition C.5 that

$$I(\mathcal{A}(X_{1:n});\theta) \le \tau_n I(\mathcal{A}(X_{1:n});X_{1:n}) + 8\epsilon_n \log(|\mathcal{M}|/\epsilon_n)\,. \tag{data gen SDPI}$$

Similarly, the test/train SDPI implies by Proposition C.5, after rearrangement, that

$$I(\mathcal{A}(X_{1:n});X_{1:n}) \ge \frac{1}{\rho_n} I(\mathcal{A}(X_{1:n});X) - 8(\delta_n/\rho_n)\log(|\mathcal{M}|/\delta_n)\,. \tag{test/train SDPI}$$

We therefore see that

$$\begin{aligned}
\mathrm{mem}_n(\mathcal{A},\mathbf{P}) &= I(\mathcal{A}(X_{1:n});X_{1:n} \mid \theta) \\
&= I(\mathcal{A}(X_{1:n});X_{1:n},\theta) - I(\mathcal{A}(X_{1:n});\theta) && \text{[chain rule]} \\
&= I(\mathcal{A}(X_{1:n});X_{1:n}) - I(\mathcal{A}(X_{1:n});\theta) && [\mathcal{A}(X_{1:n}) \perp\!\!\!\perp \theta \mid X_{1:n}] \\
&\ge (1-\tau_n) I(\mathcal{A}(X_{1:n});X_{1:n}) - 8\epsilon_n \log(|\mathcal{M}|/\epsilon_n) && \text{(data gen SDPI)} \\
&\ge (1-\tau_n)\left(\frac{1}{\rho_n}\cdot I(\mathcal{A}(X_{1:n});X) - (8\delta_n/\rho_n)\log(|\mathcal{M}|/\delta_n)\right) && \text{(test/train SDPI)} \\
&\quad - 8\epsilon_n \log(|\mathcal{M}|/\epsilon_n)\,.
\end{aligned}$$

This establishes the first claim. To further prove the second claim, it remains to show that if $\mathrm{err}(\mathcal{A},\mathbf{P}) \le \alpha$ then $I(\mathcal{A}(X_{1:n});X) \ge C_\alpha$, namely that a non-trivial error bound implies a mutual information lower bound. This Fano-type argument is rather standard, and we defer its proof to Appendix F.2. □

Before continuing, we discuss the typical use of Theorem C.7. Our aim is to show that problems of interest satisfy SDPI with $\tau_n, \rho_n \ll 1$, and to quantify these SDPI constants as a function of n. In our results ϵ_n, δ_n will be negligible, resulting in a negligible term neg_n. Temporarily ignoring this negligible additive term, as long as $\tau_n \le 1/2$ we see that Theorem C.7 implies for any learning algorithm $\mathcal{A}$: $\mathrm{mem}_n(\mathcal{A},\mathbf{P}) = \Omega(1/\rho_n)$. In some of the applications, we will see that $\rho_n = \Theta(n\rho_1)$, which is closely related to "advanced composition" from the differential privacy (Dwork et al., 2010). Intuitively, this means that the dataset $X_{1:n}$ becomes more correlated with θ and $X \sim \mathcal{P}_\theta$ as the number of samples grows, hence having more information about test. Finally, in the simplest setting of interest in which the inner product between two independent samples scales as $\sqrt{d}$, even a single sample suffices to achieve low (expected) classification error, yet in this setting of parameters, $\mathrm{mem}_n(\mathcal{A},\mathbf{P}) \gtrsim 1/\rho_n \gtrsim 1/n\rho_1 \gtrsim d/n$.

Overall, Theorem C.7 reduces proving memorization lower bounds to proving two SDPIs, and quantifies memorization via the coefficients τ_n, ρ_n. We remark that the data generation SDPI can be bypassed whenever it is easy (or easier) to prove an explicit upper bound on $I(\mathcal{A}(X_{1:n});\theta)$ instead of relating it to $I(\mathcal{A}(X_{1:n});X_{1:n})$, as we do in our third application in Section D.3. The next section addresses the question of computing these coefficients.

C.2. Proving SDPIs via Dominating Variables

Having established in Theorem C.7 that memorization follows from SDPIs in the process that generates the data, we address the computation of the corresponding SDPI coefficients. Our derivation of excess memorization lower bounds proceeds by reducing memorization to implicit variables that dominate the learning problem. The following theorem formalizes this, as illustrated in Figure 2.

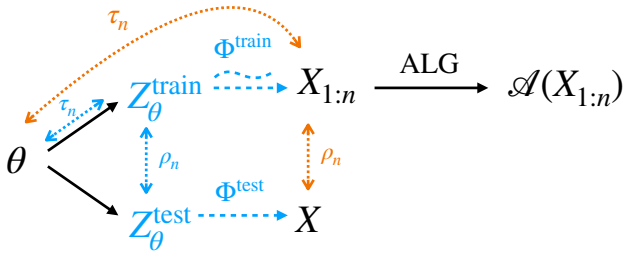

Figure 2. Illustration of Theorem C.8. The blue variables dominate the learning problem, and each blue SDPI implies an (approximate) SDPI in orange, resulting in excess memorization.

Theorem C.8. *Let $\mathbf{P} = (\mathcal{P}_\theta)_{\theta\sim\Psi}$ be a learning problem, and suppose $(Z_\theta^{\text{train}}, Z_\theta^{\text{test}})$ are jointly distributed random variables parameterized by θ so that $Z_\theta^{\text{train}} \perp\!\!\!\perp Z_\theta^{\text{test}} \mid \theta$, and that there are mappings $\Phi^{\text{train}}, \Phi^{\text{test}}$ such that for all θ, $d_{\text{TV}}(\Phi^{\text{train}}(Z_\theta^{\text{train}}), \mathcal{P}_\theta^n) \le \delta_n$, and $\Phi^{\text{test}}(Z_\theta^{\text{test}}) \sim \mathcal{P}_\theta$. Then:*

1. *(Data generation SDPI) If the marginal pair $(\theta, Z_\theta^{\text{train}})$ for $\theta \sim \Psi$ satisfies τ_n-SDPI, then $(\theta, X_{1:n})$ for $\theta \sim \Psi$, $X_{1:n} \sim \mathcal{P}_\theta^n$ satisfy (τ_n, δ_n)-SDPI.*
2. *(Test/train SDPI) If the marginal pair $(Z_\theta^{\text{test}}, Z_\theta^{\text{train}})$ satisfies ρ_n-SDPI, then $(X, X_{1:n})$ for $X \sim \mathcal{P}_\theta$ and $X_{1:n} \sim \mathcal{P}_\theta^n$ satisfies (ρ_n, δ_n)-SDPI.*

Proof of Theorem C.8. To prove the first item, recall that for all θ it holds that $d_{\text{TV}}(\Phi^{\text{train}}(Z_\theta^{\text{train}}), \mathcal{P}_\theta^n) \le \delta$, so by definition it suffices to show that $(\theta, \Phi^{\text{train}}(Z_\theta^{\text{train}}))$ satisfy τ_n-SDPI. Indeed, for any M such that $\theta \perp\!\!\!\perp M \mid \Phi^{\text{train}}(Z_\theta^{\text{train}})$:

$$I(M;\theta) \le \tau_n I(M; Z_\theta^{\text{train}}) \le \tau_n I(M; \Phi^{\text{train}}(Z_\theta^{\text{train}})) \,,$$

where the first inequality follows from the data generation SDPI assumption, and the second inequality follows from the DPI since $Z_\theta^{\text{train}} \perp\!\!\!\perp M \mid \Phi^{\text{train}}(Z_\theta^{\text{train}})$.

To prove the second item, we first note that it suffices to show that $(\Phi^{\text{test}}(Z_\theta^{\text{test}}), \Phi^{\text{train}}(Z_\theta^{\text{train}}))$ satisfy ρ_n-SDPI. This is true since $\Phi^{\text{test}}(Z_\theta^{\text{test}})$ is distributed as X (according to $\mathcal{P}_\theta$) and $Z_\theta^{\text{train}} \perp\!\!\!\perp Z_\theta^{\text{test}} \mid \theta$. By our assumption, this means that conditioned on any value of $\Phi^{\text{test}}(Z_\theta^{\text{test}})$, the distribution of $\Phi^{\text{train}}(Z_\theta^{\text{train}})$ is δ_n close in TV distance to $\mathcal{P}_\theta^n$. Thus for every θ a pair $(X, X_{1:n}) \sim \mathcal{P}_\theta \times \mathcal{P}_\theta^n$ can be seen as a sample from $\Phi^{\text{test}}(Z_\theta^{\text{test}})$ and an independent sample from a distribution close in TV distance to the distribution of $\Phi^{\text{train}}(Z_\theta^{\text{train}}))$.

Now, for any M such that $\Phi^{\text{test}}(Z_\theta^{\text{test}}) \perp\!\!\!\perp M \mid \Phi^{\text{train}}(Z_\theta^{\text{train}})$:

$$I(M; \Phi^{\text{test}}(Z_\theta^{\text{test}})) \overset{(1)}{\le} I(M; Z_\theta^{\text{test}}) \overset{(2)}{\le} \rho_n I(M; Z_\theta^{\text{train}}) \overset{(3)}{\le} \rho_n I(M; \Phi^{\text{train}}(Z_\theta^{\text{train}})) \,,$$

where (1) is the DPI, (2) follows from the test/train SDPI assumption since $\Phi^{\text{train}}(Z_\theta^{\text{train}}) \perp\!\!\!\perp Z_\theta^{\text{test}} \mid Z_\theta^{\text{train}}$ and therefore $M \perp\!\!\!\perp Z_\theta^{\text{test}} \mid Z_\theta^{\text{train}}$, and (3) follows from the DPI since $M \perp\!\!\!\perp Z_\theta^{\text{train}} \mid \Phi^{\text{train}}(Z_\theta^{\text{train}})$.

□

The theorem above shows that if a pair of variables $Z_\theta^{\text{train}}, Z_\theta^{\text{test}}$ simulate the train and test data (up to some approximation), then we can reduce the computation of the data generation coefficient τ_n and test/train coefficient ρ_n to these dominating variables with only (presumably small) additive loss.

Remark C.9. The dominating variables may appear related to the concept of "sufficient statistics" (cf. Cover & Thomas, 1999; Polyanskiy & Wu, 2024) of the sample and test data. The main difference is that the discussed variables need not be statistics of the data, i.e. computable from it.

D. Applications

In this section, we describe several applications of our framework. Our focus here will be on problems where high accuracy can be achieved given a single (positive) example. This setting is the closest to our motivating problem of memorization of entire data points. However, by appropriately choosing the parameters, the trade-off can be shown in problems with higher sample complexity.

D.1. Gaussian Clustering

We consider a Gaussian clustering problem. Formally, given $\lambda \in (0,1)$ to be fixed later, the problem $\mathbf{P}_{\mathrm{G}} = (\mathcal{P}_\theta)_{\theta\sim\Psi}$ is defined as

$$\mathcal{P}_\theta \;=\; \mathcal{N}(\lambda\theta, (1-\lambda^2)I_d)\;, \qquad \theta\sim\Psi \;=\; \mathcal{N}(0_d, I_d)\;.$$

Note that the null distribution, namely the marginal of $\mathcal{P}_\theta$ over $\theta\sim\Psi$, equals $\mathcal{P}_0 = \mathcal{N}(0_d, I_d)$. Hence, the problem corresponds to classifying between samples that are λ-correlated with an unknown parameter θ, and samples that have zero correlation with θ.

Our main result for this problem instance is the following:

Theorem D.1. *In the Gaussian clustering problem $\mathbf{P}_{\mathrm{G}}$, assume $\lambda = Cd^{-1/4}$ for some sufficiently large absolute constant $C>0$. Then the following hold:*

- *There exists an algorithm $\mathcal{A}$, that given a single sample (i.e. $n=1$) satisfies $\mathrm{err}(\mathcal{A},\mathbf{P}_{\mathrm{G}})\le 0.01$.*
- *Any algorithm $\mathcal{A}$ with constant $\mathrm{err}(\mathcal{A},\mathbf{P}_{\mathrm{G}})\le\alpha<\frac{1}{2}$ satisfies*

$$\mathrm{mem}_n(\mathcal{A},\mathbf{P}_{\mathrm{G}}) \ge \frac{1-\lambda^2}{\lambda^4 n}\cdot(1-2\alpha)\log\left(\frac{1-\alpha}{\alpha}\right) = \Omega\left(\frac{d}{n}\right)\;.$$

- *As long as $n\le c\sqrt{d}$ for some sufficiently small absolute constant $c>0$, the lower bound above is nearly-tight: There is a learning algorithm $\mathcal{A}$ such that $\mathrm{err}(\mathcal{A},\mathbf{P}_{\mathrm{G}})\le 0.01$ and $\mathrm{mem}_n(\mathcal{A},\mathbf{P}_{\mathrm{G}}) = O(d\cdot\log(\frac{d}{n})/n)$.*

Remark D.2. For our upper bounds, we focus on small constant error chosen as 0.01 for simplicity. More generally, if the correlation is set $\lambda = Cd^{-1/4}$ for $C=\Theta(\sqrt{\log(1/\alpha)})$, the same statements hold for learners with error at most α, affecting only logarithmic terms in the resulting excess memorization.

We provide here a sketch of the proof of Theorem D.1, which appears in Appendix G.

Proof sketch of Theorem D.1. The first item follows from standard Gaussian concentration bounds, since for a single sample $X_1\sim\mathcal{P}_\theta=\mathcal{N}(\lambda\theta,(1-\lambda^2)I_d)$, it holds that on one hand for $X\sim\mathcal{P}_\theta$: $\langle X,X_1\rangle \gtrsim \lambda^2 d = C^2\sqrt{d}$ with high probability, while on the other hand for $X\sim\mathcal{P}_0=\mathcal{N}(0_d,I_d)$: $\langle X,X_1\rangle\lesssim\sqrt{d}$. Therefore, for a sufficiently large $C>0$, a linear classifier will have small error.

To prove the second item, we rely on the ideas presented in Section C. Particularly, we note that the information about θ contained in the dataset $X_1,\ldots,X_n\sim\mathcal{N}(\lambda\theta,(1-\lambda^2)I_d)$ is dominated by the empirical average $\hat{X}:=\frac{1}{n}\sum_{i\in[n]}X_i\sim\mathcal{N}(\lambda\theta,\frac{1-\lambda^2}{n}I_d)$. Noting that $\lambda\theta\sim\mathcal{N}(0_d,\lambda^2 I_d)$, we see that the variance of $\hat{X}$ in every direction is $\lambda^2+\frac{1-\lambda^2}{n}=\frac{1+\lambda^2(n-1)}{n}$. Therefore, by rescaling $\hat{X}$ by $\sqrt{n/(1+\lambda^2(n-1))}$, we get a Gaussian with unit variance $Z_\theta^{\mathrm{train}}=\sqrt{n/(1+\lambda^2(n-1))}\cdot\hat{X}$ which is a dominating variable for the dataset, whose coordinate-wise correlation with a fresh sample $X\sim\mathcal{N}(\lambda\theta,(1-\lambda^2)I)$ is $\lambda^2\sqrt{n/(1+\lambda^2(n-1))}$. By Fact C.2, this gives us a test/train SDPI with $\rho_n=\lambda^4 n/(1+\lambda^2(n-1))$. The same dominating variable also proves a data generation τ_n-SDPI by computing the coordinate-wise correlation between θ and $Z_\theta^{\mathrm{train}}$ as $\lambda\sqrt{n/(1+\lambda^2(n-1))}$, and therefore $\tau_n=\lambda^2 n/(1+\lambda^2(n-1))$ once again by Fact C.2. Plugging these SDPI coefficients into Theorem C.7, we see that $\mathrm{mem}_n(\mathcal{A},\mathbf{P}_{\mathrm{G}})\gtrsim(1-\tau_n)/\rho_n=\frac{1-\lambda^2 n/(1+\lambda^2(n-1))}{\lambda^4 n/(1+\lambda^2(n-1))}=\frac{1-\lambda^2}{\lambda^4 n}\approx d/n$.

To prove the last item, we provide a simple low error algorithm that returns a low error classifier which is describable using $\widetilde{O}(d/n)$ bits, by using projections. To that end, considering once again the empirical average $\hat{X}\sim\mathcal{N}(\lambda\theta,\frac{1-\lambda^2}{n}I_d)$, we project it onto $\mathbb{R}^\ell$ for $\ell\approx d/n$, obtain $\hat{X}^{[1:\ell]}$, and consider the linear classifier that projects onto $\mathbb{R}^\ell$ and takes the inner product

with $\hat{X}^{[1:\ell]}$. The idea here is that Gaussian concentration arguments ensure that on one hand for $X \sim \mathcal{P}_\theta$: $\langle X^{[1:\ell]}, \hat{X}^{[1:\ell]} \rangle \gtrsim \lambda^2 \ell \approx C^2 d^{1/2}/n$ with high probability, while on the other hand for $X \sim \mathcal{P}_0 = \mathcal{N}(0_d, I_d)$: $\langle X^{[1:\ell]}, \hat{X}^{[1:\ell]} \rangle \lesssim \sqrt{\ell/n} \approx d^{1/2}/n$. Therefore the classifier achieves low error, and moreover, requires roughly ℓ bits to fully describe (ignoring logarithmic terms due to quantization). In particular, the classifier cannot contain more than $\widetilde{O}(\ell)$ bits of information about the dataset, proving the claimed memorization upper bound. □

D.2. Boolean Clustering

We consider a Boolean clustering problem, which is the Boolean analogue of the previously discussed Gaussian mean estimation problem. Formally, given $\lambda \in (0,1)$ to be chosen later, the problem $\mathbf{P}_{\mathrm{B}} = (\mathcal{P}_\theta)_{\theta \sim \Psi}$ corresponds to

$$\mathcal{P}_\theta \;=\; \mathrm{Bin}_{\frac{1-\lambda}{2}}(\theta)\,, \qquad \theta \sim \Psi \;=\; \mathcal{U}(\{\pm 1\}^d)\,.$$

Note that the null distribution is uniform $\mathcal{P}_0 = \mathcal{U}(\{\pm 1\}^d)$. Namely, given samples correlated with θ (which can be thought of as cluster around θ), the problem corresponds to classifying between fresh samples that are λ-correlated coordinate-wise with some θ, and uniformly generated samples. Our main result for this problem instance is the following:

Theorem D.3. *In the Boolean clustering problem $\mathbf{P}_{\mathrm{B}}$, assume $\lambda = Cd^{-1/4}$ for some sufficiently large absolute constant $C > 0$. Then the following hold:*

- *There exists an algorithm $\mathcal{A}$, that given a single sample (i.e. $n = 1$) satisfies $\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) \le 0.01$.*
- *Any algorithm $\mathcal{A} : \mathcal{X}^n \to \mathcal{M}$ with $\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) \le \alpha < \frac{1}{2}$ satisfies*

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = \Omega\left(\frac{1 - \lambda^2 n \log\log|\mathcal{M}|^d}{\lambda^4 n \log\log|\mathcal{M}|^d}(1-2\alpha)\log\left(\frac{1-\alpha}{\alpha}\right)\right)\,.$$

  In particular, as long as $n \le c\sqrt{d}$ for some sufficiently small absolute constant $c > 0$, and $|\mathcal{M}| \le \exp(d^{\tilde{C}})$ for some absolute constant $\tilde{C} > 0$, then

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = \widetilde{\Omega}\left(\frac{1-\lambda^2 n}{\lambda^4 n}(1-2\alpha)\log\left(\frac{1-\alpha}{\alpha}\right)\right) = \widetilde{\Omega}\left(\frac{d}{n}\right)\,.$$

- *The lower bound above, as well as the sample size condition, are both nearly-tight: On one hand, if $n \le \sqrt{d}$ then there is a learning algorithm $\mathcal{A}$ such that $\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) \le 0.01$ and $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = O(d/n)$. On the other hand, there is a learning algorithm $\mathcal{A}$ such that $\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) \le 0.01$ and $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = O(d^2 \exp(-n/\sqrt{d}))$, and so if $n \gtrsim \sqrt{d}\log d$ then $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) \le 1/\mathrm{poly}(d)$.*

The theorem shows that up to $n \approx \sqrt{d}$ samples the excess memorization decays as $\tilde{\Theta}(d/n)$, and that afterwards it drops to nearly zero. We note that the extremely mild condition that $|\mathcal{M}| \in \exp(\mathrm{poly}(d))$, namely the hypothesis class size is not super-exponential in the dimension, is likely just an artifact of the proof technique which is based on the approximation strategy introduced in Section C.2.

The proof of Theorem D.3 is similar in spirit to that of Theorem D.1 as we previously sketched, yet the technical details in the proof of the memorization lower bound (second item) are more challenging. This follows from the fact that, as opposed to the Gaussian case, the boolean dataset $X_{1:n} \sim \mathrm{Bin}_{\frac{1-\lambda}{2}}(\theta)^n$ does not have a simple dominating variable. Instead, we use arguments related to advanced composition from the differential privacy literature, to argue that $X_{1:n}$ is statistically close to a post-processing of $Z_\theta^{\mathrm{train}} \sim \mathrm{Bin}_{\frac{1-\xi}{2}}(\theta)$ for $\xi \approx \sqrt{n}\lambda$, namely a single variable which is $\sqrt{n}$-times more correlated with θ. We can then invoke Fact C.3 and use our approximate reduction in Theorem C.8 to obtain the required SDPIs, by noting that the coordinate-wise correlation of θ and $Z_\theta^{\mathrm{train}}$ is $\sqrt{\tau_n} = \xi$, whereas that of $Z_\theta^{\mathrm{train}}$ and $X \sim \mathrm{Bin}_{\frac{1-\lambda}{2}}(\theta)$ is $\sqrt{\rho_n} = \xi\lambda$. Consequently, up to negligible additive factors, we obtain $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) \gtrsim \frac{1-\xi^2}{\lambda^2\xi^2} \approx \frac{1-\lambda^2 n}{\lambda^4 n}$, and under the assignment of λ and assumption that $n \ll \sqrt{d}$, the latter simplifies to d/n.

The nearly-matching upper bounds are realized by an algorithm that computes the bit-wise majority vote over the sample. For the regime $n \lesssim \sqrt{d}$, the algorithm only computes the majority along the first $\ell \approx d/n$ coordinates, and returns a linear

classifier in the projected space, similarly to the Gaussian case. Concentration arguments ensure that the algorithm has small error, while clearly requiring at most ℓ bits of memory, thus in particular no more than $\ell \approx d/n$ bits from the training set can be memorized. When $n \gtrsim \sqrt{d}\log(d)$, computing the majority vote in each coordinate reconstructs the parameter θ with very high confidence, so in this regime, accurate learning is possible with nearly zero excess memorization. The full proof appears in Appendix H.

D.3. Sparse Boolean Hypercube

Finally, we apply our framework to the sparse Boolean hypercube clustering problem defined by Brown et al. (2021). Given $\nu > 0$ to be chosen later, the problem $\mathbf{P}_{\mathrm{sB}} = (\mathcal{P}_\theta)_{\theta\sim\Psi}$ is defined as follows. The parameter $\theta = (S, y)$ is sampled by choosing $S \subseteq [d]$ to be a random subset that includes each $i \in S$ independently with probability ν, and picking $y_j \sim \mathcal{U}(\{\pm 1\})$ independently for every $j \in S$. The distribution $\mathcal{P}_\theta$ is defined to be the distribution of X such that for $j \in S$, $X_j = y_j$ with probability 1, and $X_j \sim \mathcal{U}(\{\pm 1\})$ independently for every $j \notin S$.

This problem can be seen as learning a sparse Boolean conjunction since positive samples x satisfy the conjunction $\bigwedge_{j\in S}(x_j == y_j)$. Our next result characterizes the memorization trade-off for this problem, establishing a faster memorization decay compared to the previous problems:

Theorem D.4. *In the sparse Boolean hypercube clustering problem $\mathbf{P}_{\mathrm{sB}}$, assume $\nu = Cd^{-1/2}$ for some sufficiently large absolute constant $C > 0$, and that $n \le c\log d$ for some sufficiently small absolute constant $c > 0$. Then the following hold:*

- *There exists an algorithm $\mathcal{A}$, that given a single sample (i.e. $n = 1$) satisfies $\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) \le 0.01$.*
- *Any algorithm $\mathcal{A}$ with $\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) \le \alpha < \frac{1}{2}$ satisfies*

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) = \Omega\left(\frac{(1-2\alpha)\log\left(\frac{1-\alpha}{\alpha}\right)}{\nu^2 2^{2n}}\right) - O(\sqrt{d}\log d) = \Omega\left(\frac{d}{2^{2n}}\right).$$

- *The lower bound above is nearly-tight: there is a learning algorithm $\mathcal{A}$ such that $\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) \le 0.01$ and $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) = O(d\log(d)/2^{2n})$.*

The proof of Theorem D.4 appears in Appendix I. To prove the memorization lower bound, we establish a test/train ρ_n-SDPI for $\rho_n \approx \nu^2 2^{2n}$. To do so, we introduce a dominating variable $Z_\theta^{\mathrm{train}} \sim \mathrm{Bin}_{\frac{1}{2}+\xi}(X)$ for $X \sim \mathcal{P}_\theta$, $\xi \approx \nu 2^n$, and show that processing it into $\tilde{X}_{1:n}$ by fixing each coordinate with some suitable probability, or else drawing each sample independently along that coordinate, results in a training set which is identically distributed as $X_{1:n}$. Then, we avoid the need of a data generation SDPI by directly upper bounding the entropy of θ by $H(\theta) \lesssim \sqrt{d}\log d$, hence obtaining by the chain rule $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) \ge I(\mathcal{A}(X_{1:n}); X_{1:n}) - H(\theta) \gtrsim \frac{1}{\rho_n} - \sqrt{d}\log d = \Omega\left(\frac{d}{2^{2n}}\right)$, the latter holding under our assumptions on ν and n. The nearly-matching upper bound follows by considering an algorithm which only stores a subset of $O(d/2^{2n})$ coordinates in which the sample is constant, and arguing that sufficiently many of them are indeed in S with high probability, which suffices for generalization via standard concentration bounds.

E. Lower Bounds for Mixtures of Clusters

We now consider "mixture-of-clusters" generalization of our learning setting similar to that defined in (Feldman, 2020; Brown et al., 2021). In this setting, data is sampled from some unknown mixture of clusters. The learner however has a prior over the distribution of frequencies of the clusters. More formally, let $\mathbf{P} = (\mathcal{P}_\theta)_{\theta\sim\Psi}$ denote a problem in the binary classification setting defined in Section B. Recall, that in this setting a distribution $\mathcal{P}_\theta$ is drawn from a meta-distribution $\theta \sim \Psi$. The learning algorithm is then given a number of examples $X_{1:n} \sim \mathcal{P}_\theta^n$ and needs to classify a fresh example as coming from $\mathcal{P}_\theta$ or the "null" distribution $\mathcal{P}_0$ (defined as the marginal of $X \sim \mathcal{P}_{\theta'}$ where $\theta' \sim \Psi$).

For a natural number $k \in \mathbb{N}$ representing the number of clusters, we model the prior information about frequencies of clusters using a meta-distribution Π over $\Delta([k])$ (i.e., Π is a distribution over distributions on $[k]$). We define a multi-cluster version of $\mathbf{P}$, denoted by $\mathbf{P}^{\mathsf{mult}} = (\{\mathcal{P}_\theta\}_{\theta\sim\Psi}, \Pi, k)$ as follows. First, k random cluster parameters $\theta_{1:k} = (\theta_1, \ldots, \theta_k)$ are drawn i.i.d. from Ψ and a frequency vector π is drawn from Π. Then, let $\mathcal{P}_{\theta_{1:k},\pi}$ be the mixture distribution of $\mathcal{P}_{\theta_i}$'s where $\mathcal{P}_{\theta_i}$ has weight π_i. Namely, $\mathcal{P}_{\theta_{1:k},\pi}$ is the distribution of $X \sim \mathcal{P}_{\theta_I}$ where $I \sim \pi$. The learning algorithm is then given

training data $X_1, \ldots, X_n \sim \mathcal{P}^n_{\theta_{1:k},\pi}$, and needs to distinguish samples coming from $\mathcal{P}_{\theta_{1:k},\pi}$ from those coming from $\mathcal{P}_0$. Formally, let $(X, Y) \sim \mathcal{P}^{\text{test}}_{\theta,\pi}$ be a distribution such that $Y \sim \{0, 1\}$ is uniformly random. Conditioning on Y, we have $X \sim \mathcal{P}_{\theta_{1:k},\pi}$ if $Y = 1$ and $X \sim \mathcal{P}_0$ otherwise. The error of a predictor $h : \mathcal{X} \to \{0, 1\}$ is then defined as

$$\operatorname{err}(h) := \Pr_{(X,Y)\sim\mathcal{P}^{\text{test}}_{\theta_{1:k},\pi}}[h(X) \neq Y].$$

Suppose $\mathcal{A}$ is a learning algorithm operating on n samples. We define its (average-case) error as

$$\operatorname{err}(\mathcal{A}, \mathbf{P}^{\mathsf{mult}}) := \mathop{\mathbb{E}}_{\substack{\theta_1,\ldots,\theta_k\sim\Psi^k,\pi\sim\Pi \\ X_{1:n}\sim\mathcal{P}^n_{\theta_{1:k},\pi} \\ h\leftarrow\mathcal{A}(X_{1:n})}} [\operatorname{err}(h)].$$

As in (Feldman, 2020; Brown et al., 2021), we will only consider product priors Π in which frequencies of clusters are chosen independently up to a normalization constant. Specifically for a distribution p over $[0, 1]$, the *product prior* Π^k_p is defined by independently sampling $p_1, \ldots, p_k \sim p^k$, and defining $\pi_i = \frac{p_i}{\sum_{i'} p_{i'}}$.

Note that this setting is slightly different from the setting considered by (Brown et al., 2021). There, the algorithm gets *labeled* data (X, I) where $I \sim \pi$ and $X \sim \mathcal{P}_{\theta_I}$. Then, given a test example $X \sim \mathcal{P}_{\theta,\pi}$, the algorithm is tasked to label which cluster was X sampled from (i.e., this is a multi-class classification problem). As we demonstrate below, the two versions of the problems are subject to the same memorization phenomenon up to a factor logarithmic in k. Here, we present the result for the binary classification setting (i.e., the algorithm needs to tell whether a point is from any one cluster, or is from $\mathcal{P}_0$), noting that essentially the same proof works for the multi-class clustering setting.

E.1. Memorization Lower Bound

Given a learning problem $\mathbf{P}^{\mathsf{mult}}$, we would like to understand the amount of memorization required to achieve a close-to-optimal error. Let us first consider the natural upper bound of memorization. To ease our discussion, we assume that the learner gets the *additional* knowledge of the cluster ID of its examples. Namely, the learners gets i.i.d. examples from the distribution $\widetilde{\mathcal{P}}_{\theta,\pi}$, where an element $(X, i) \sim \widetilde{\mathcal{P}}_{\theta,\pi}$ is sampled by first drawing $i \sim \pi$ and then $X \sim \mathcal{P}_{\theta_i}$. We note that this assumption only makes our lower bounds stronger since a learning algorithm can always ignore the cluster index information. To solve the multi-cluster problem it suffices to be able to distinguish each of the clusters $\mathcal{P}_{\theta_1}, \ldots, \mathcal{P}_{\theta_k}$ from $\mathcal{P}_0$ with low error. Let $\pi \sim \Pi$ be a random frequency vector. For each $i \in [k]$, the number of examples from $\mathcal{P}_{\theta_i}$ is expected to be $n \cdot \pi_i$. Consequently, the amount of memorization is roughly $\operatorname{mem}_{n\pi_i}(\mathcal{A}, \mathbf{P})$, where $\mathcal{A}$ is the algorithm we use on each cluster. By adding up the memorization from different clusters we get $\sum_{i\in[k]} \operatorname{mem}_{n\pi_i}(\mathcal{A}, \mathbf{P})$ as an upper bound. We show that any nearly-optimal algorithm is subject to a lower bound of essentially the same form.

Before proceeding we will need the following property of product priors from (Feldman, 2020).

Lemma E.1 (Feldman, 2020, Lemma 2.1)**.** *For a distribution p over $[0, 1]$ let Π^k_p be the product prior and denote by $\bar{p}$ the marginal distribution of the frequency of (any) element, namely the distribution of π_1 where $\pi \sim \Pi^k_p$. Consider the random variable $(\pi, i_1, \ldots, i_n)$ where $\pi \sim \Pi^k_p$ and $(i_1, \ldots, i_n) \sim \pi^n$. For any sequence of indices $(j_1, \ldots, j_n)$ that includes $u \in [k]$ exactly $\ell \in [0, n]$ times, it holds that*

$$\mathop{\mathbb{E}}_{\pi\sim\Pi^k_p,(i_1,\ldots,i_n)\sim\pi^n}[\pi_u \mid (i_1, \ldots, i_n) = (j_1, \ldots, j_n)] = \tau_\ell := \frac{\mathbb{E}_{\alpha\sim\bar{p}}[\alpha^{\ell+1}(1-\alpha)^{n-\ell}]}{\mathbb{E}_{\alpha\sim\bar{p}}[\alpha^{\ell}(1-\alpha)^{n-\ell}]}.$$

To interpret Lemma E.1, suppose a learner has the prior knowledge of p and sees a sequence of examples $(X_1, i_1), \ldots, (X_n, i_n)$ in which the cluster u appears exactly ℓ times. Lemma E.1 tells us that the expectation of the posterior value of the frequency of u is τ_ℓ. Therefore, it is natural to guess that if the learner does not perform well on the u-th cluster, it will translate into an error of τ_ℓ for the learner. We will show next that the previously described intuition is indeed correct.

To simplify the discussion, we focus here on the problem settings that are learnable with *vanishing* error. Namely, we make the following assumption:

Assumption E.2. Let $\mathbf{P} = \{\mathcal{P}_\theta\}_{\theta\sim\Psi}$ be a learning problem. We assume that $\mathbf{P}$ is such that, there exists an algorithm that given a single example $X \sim \mathcal{P}_\theta$, with probability $1 - \operatorname{neg}(n, k, d)$ over $\theta \sim \Psi$ and $X \sim \mathcal{P}_\theta$, achieves classification error of $o(\frac{1}{k^2})$.

We remark that Assumption E.2 is not too restrictive: for all applications we have investigated in this paper (i.e. Gaussian clustering, Boolean clustering, and Sparse Boolean clustering), the assumption can be satisfied by increasing the correlation parameter by a factor logarithmic in k (see Remark D.2).

From sub-optimality to memorization: To count the error for clusters of specific sizes we introduce some notation. Let $S = ((X_1, i_1), \dots, (X_n, i_n))$ be a sequence of examples. Having observed S, this induces a posterior distribution on $\theta_{1:k}$ and $\pi_{1:k}$. Furthermore, it is easy to observe that $\theta_{1:k}$ and $\pi_{1:k}$ are *independent*. Let $\theta_{1:k} \mid S$ and $\pi_{1:k} \mid S$ be the posterior distributions. For any $\ell \in [0, n]$, let $I_{n\#\ell}(S) \subseteq [k]$ be the set of clusters i that appear exactly ℓ times in the sequence. For any model $h : \mathcal{X} \to \{0, 1\}$, we define its expected error on the set of clusters i that appear exactly ℓ times in S by

$$\operatorname{errn}(h, \theta_{1:k}, S, \ell) := \frac{1}{2}\left(\sum_{i \in I_{n\#\ell}(S)} \Pr_{X \sim \mathcal{P}_{\theta_i}}[h(X) = 0] + \Pr_{X' \sim \mathcal{P}_0}[h(X') = 0]\right),$$

and the expectation of this error on the posterior distribution $\theta_{1:k} \mid S$ as

$$\operatorname{errn}(h, S, \ell) := \mathop{\mathbb{E}}_{\theta_{1:k} \mid S}\left[\operatorname{errn}(h, \theta_{1:k}, S, \ell)\right].$$

Similarly for an algorithm $\mathcal{A}$, we define

$$\operatorname{errn}(\mathcal{A}, \theta_{1:k}, S, \ell) := \mathop{\mathbb{E}}_{h \sim \mathcal{A}(S)}\left[\operatorname{errn}(h, \theta_{1:k}, S, \ell)\right].$$

and

$$\operatorname{errn}(\mathcal{A}, S, \ell) := \mathop{\mathbb{E}}_{\theta_{1:k} \mid S}\left[\operatorname{errn}(\mathcal{A}, \theta_{1:k}, S, \ell)\right].$$

One can observe that these two definitions do *not* depend on the prior Π_p^k. We also define $\operatorname{opt}(\mathbf{P}^{\mathsf{mult}} \mid S)$ (resp. $\operatorname{opt}(\mathbf{P}^{\mathsf{mult}})$) to be the minimum of $\operatorname{err}(\mathcal{A}, \mathbf{P}^{\mathsf{mult}}, S)$ (resp. $\operatorname{err}(\mathcal{A}, \mathbf{P}^{\mathsf{mult}})$).

The following theorem shows that the sub-optimality of any learning algorithm can be expressed in terms of its expected error on posterior distributions.

Theorem E.3. *Let $\mathbf{P}^{\mathsf{mult}} = (\{\mathcal{P}_\theta\}_{\theta \sim \Psi}, \Pi_p^k, k)$ be a learning problem where $\mathbf{P}$ is subject to Assumption E.2. For any learning algorithm $\mathcal{A}$, with high probability over a data set $S \in (\mathcal{X} \times [k])^n$, it holds that*

$$\operatorname{err}(\mathcal{A}, \mathbf{P}^{\mathsf{mult}} \mid S) \geq \operatorname{opt}(\mathbf{P}^{\mathsf{mult}} \mid S) + \sum_{1 \leq \ell \leq n} \tau_\ell \cdot \operatorname{errn}(\mathcal{A}, S, \ell) - O(1/k).$$

In particular, it follows that

$$\operatorname{err}(\mathcal{A}, \mathbf{P}^{\mathsf{mult}}) \geq \operatorname{opt}(\mathbf{P}^{\mathsf{mult}}) + \mathop{\mathbb{E}}_{\theta_{1:k} \sim \Psi^k, \pi \sim \Pi_p^k, S \sim \widetilde{\mathcal{P}}_{\theta,\pi}}\left[\sum_{1 \leq \ell \leq n} \operatorname{errn}(\mathcal{A}, \theta_{1:k}, S, \ell)\right] - O(1/k).$$

Theorem E.3 says that, if an algorithm $\mathcal{A}$ is sufficiently close to being optimal, then $\operatorname{errn}(\mathcal{A}, S, \ell)$ must be small on average. We next show that low average $\operatorname{errn}(\mathcal{A}, S, \ell)$ implies memorization lower bounds. For this, we will rely on the fact that memorization lower bounds for the problems we consider scale at least linearly with the advantage over random guessing (namely, $1/2 - \operatorname{err}$).

Assumption E.4. Let $\mathbf{P}$ be a (binary) classification problem. We assume that there exists a constant $c_{\mathbf{P}}$ such that $\operatorname{mem}_\ell(\mathbf{P}, \alpha) \geq c_{\mathbf{P}} \cdot (1 - 2\alpha) \cdot \operatorname{mem}_\ell(\mathbf{P})$ for every $\alpha \in (0, 1/2)$.

This assumption is satisfied (up to the lower order terms) by all the learning problems we have investigated (see Theorem C.7). We remark that for this assumption to hold in the case of approximate SDPIs we additionally need to constrain the size of the model output by the algorithm.

The following theorem is our main lower bound. It expresses the memorization for the multi-cluster problem $\mathbf{P}^{\mathsf{mult}}$ as the sum of memorization lower bounds for individual clusters. As in the case of lower bounds for a single cluster classification problem $\mathbf{P}$, the lower bound is scaled by the advantage over random guessing that the algorithm achieves for clusters of each size. Specifically, for clusters of size ℓ, the average advantage over random guessing of $\mathcal{A}$ when given S is equal to $|I_{n\#\ell}(S)|/2 - \mathrm{errn}(\mathcal{A}, \theta_{1:k}, S, \ell)$.

Theorem E.5. *Let $\mathbf{P}^{\mathsf{mult}} = (\{\mathcal{P}_\theta\}_{\theta\sim\Psi}, \Pi_p^k, k)$ be a learning problem where $\mathbf{P}$ is subject to Assumptions E.2 and E.4. Let $S = (X_1, i_n), \ldots, (X_n, i_n)$ be a dataset of n i.i.d. examples from $\widetilde{\mathcal{P}}_{\theta_{1:k},\pi}$ for $\theta_{1:k} \sim \Psi^k, \pi \sim \Pi_p^k$. For every algorithm $\mathcal{A}$, $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}^{\mathsf{mult}}) = I(\mathcal{A}(S); S \mid \theta_{1:k}, \pi)$ satisfies*

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}^{\mathsf{mult}}) \geq c_{\mathbf{P}} \cdot \mathop{\mathbb{E}}_{\theta_{1:k}\sim\Psi^k, \pi\sim\Pi_p^k, S\sim\widetilde{\mathcal{P}}_{\theta_{1:k},\pi}} \left[\sum_{1\leq\ell\leq n} (|I_{n\#\ell}(S)| - 2\cdot \mathrm{errn}(\mathcal{A}, \theta_{1:k}, S, \ell)) \cdot \mathrm{mem}_\ell(\mathbf{P}) \right].$$

Application Example. Theorem E.5 generalizes prior works (Feldman, 2020; Brown et al., 2021) by allowing us to reason about memorization of larger clusters (instead of only singleton clusters) in the multi-cluster context. We will now briefly describe a scenario where our new lower bounds offer a significantly better understanding of memorization. Take $\mathbf{P}$ to be a binary classification task (e.g., Gaussian/Boolean clustering, or the sparse Boolean clustering as considered by (Brown et al., 2021)). Let Π_p^k be a product prior induced by the singleton distribution p that always outputs 1. We consider the multi-cluster learning task of $\mathbf{P}^{\mathsf{mult}} = (\mathbf{P}, \Pi_p^k, k)$.

The lower bound in (Brown et al., 2021) is demonstrated for the training sample size $n = O(k)$, in which case we expect to observe many clusters with only a single example (singleton clusters). In order to achieve a close-to-optimal accuracy,[3] a learning algorithm must perform well on the singleton clusters and thus needs to memorize $\Omega(dn)$ bits. Our general technique recovers this lower bound (up to a logarithmic factor needed to ensure that Assumption E.2 holds). However, if one just slightly increases n from k to $k \log k$, the probability of observing a singleton cluster quickly approaches zero and thus the results in (Brown et al., 2021) do not lead to a meaningful memorization lower bound.

In the latter regime, we will observe $\Omega(k)$ (i.e., most) clusters of size on the order $\log k$. As we have already described in Section D, for several canonical clustering problems (such as Gaussian/Boolean clusters), with $d^{o(1)}$ training examples, the memorization remains significant, roughly d/ℓ for each cluster of size ℓ. Thus in this regime Theorem E.5 implies that algorithms that achieve (positive) constant advantage over random guessing will need to memorize $\tilde{\Omega}(kd/\log k) = \tilde{\Omega}(nd)$ bits.

F. Proofs for Section C

F.1. Proof of Proposition C.5

Note that the statement for $\delta = 0$ is simply the definition of an SDPI, so without loss of generality we can assume that $\delta > 0$. Thus, we can further assume that $|\mathcal{M}| < \infty$, since otherwise the statement trivially holds. We start the proof by proving two information theoretic results, which intuitively, show that if two random variables are statistically close, then (1) they have roughly the same Shannon entropy, and (2) any fixed randomized algorithm/channel extracts roughly the same amount of information from either source.

Lemma F.1. *Suppose P, Q are two random variables with pmfs p, q respectively, supported on a finite domain $\mathcal{M}$ such that $d_{\mathrm{TV}}(p, q) \leq \delta$. Then, we have $|H(P) - H(Q)| \leq 2\delta \log\left(\frac{|\mathcal{M}|}{\delta}\right)$.*

Proof of Lemma F.1. For brevity, write $p_y = \Pr[P = y]$ and $q_y = \Pr[Q = y]$. Since the function $x \mapsto x\log(1/x)$ is

[3] Note that when $n < O(k)$, with high probability there will be some clusters not present in the training data. Therefore, no algorithm can achieve a vanishing classification error.

monotone in $(0,1)$, we define sets $L = \{y : p_y > q_y\}$ and $R = \{y : q_y > p_y\}$ and deduce that

$$|H(P) - H(Q)| \leq \max\left\{\sum_{y\in L} p_y \log(1/p_y) - q_y \log(1/q_y), \sum_{y\in R} q_y \log(1/q_y) - p_y \log(1/p_y)\right\}$$
$$\leq \max\left\{\sum_{y\in L}(p_y - q_y)\log(1/p_y), \sum_{y\in R}(q_y - p_y)\log(1/q_y)\right\}. \tag{1}$$

We show how to upper bound the first term $\sum_{y\in L}(p_y - q_y)\log(1/p_y)$. The bound for the second term can be analogously established. Consider the set $S_P = \{y \in L : p_y < \frac{\delta}{|\mathcal{M}|}\}$. We have

$$\begin{aligned}\sum_{y\in L}(p_y - q_y)\log(1/p_y) &\leq \sum_{y\in L\setminus S_P}(p_y - q_y)\log(1/p_y) + \sum_{y\in S_P} p_y \log(1/p_y)\\ &\leq \sum_{y\in L\setminus S_P}(p_y - q_y)\log\left(\frac{\delta}{|\mathcal{M}|}\right) + \sum_{y\in S_P} p_y \log(1/p_y)\\ &\leq \delta\log\left(\frac{|\mathcal{M}|}{\delta}\right) + \sum_{y\in S_P} p_y \log(1/p_y).\end{aligned}$$

By the monotonicity of $x \mapsto x\log(1/x)$, the bound of $p_y < \frac{\delta}{|\mathcal{M}|}$, and the cardinality bound of $|S_P|$, we obtain

$$\sum_{y\in S_P} p_y \log(1/p_y) \leq \sum_{y\in S_P}\frac{\delta}{|\mathcal{M}|}\cdot\log\left(\frac{|\mathcal{M}|}{\delta}\right) \leq \delta\log\left(\frac{|\mathcal{M}|}{\delta}\right).$$

Overall, we conclude that $\sum_{y\in L}(p_y - q_y)\log(1/p_y) \leq 2\delta\log\left(\frac{|\mathcal{M}|}{\delta}\right)$. The bound for the second term of (1) can be similarly established, concluding the proof. □

Lemma F.2. *Let B_1, B_2 be two random variables over some space $\mathcal{B}$ such that $d_{\mathrm{TV}}(B_1, B_2) \leq \delta$, and suppose $\mathcal{A}\colon \mathcal{B} \to \mathcal{M}$ is some randomized algorithm. Then*

$$|I(\mathcal{A}(B_1); B_1) - I(\mathcal{A}(B_2); B_2)| \leq 4\delta\log(|\mathcal{M}|/\delta).$$

Proof of Lemma F.2. It holds that

$$|I(\mathcal{A}(B_1); B_1) - I(\mathcal{A}(B_2); B_2)| \leq |H(\mathcal{A}(B_1)) - H(\mathcal{A}(B_2))| + |H(\mathcal{A}(B_1) \mid B_1) - H(\mathcal{A}(B_2) \mid B_2)|.$$

By Lemma F.1 and the observation $d_{\mathrm{TV}}(\mathcal{A}(B_1), \mathcal{A}(B_2)) \leq d_{\mathrm{TV}}(B_1, B_2) \leq \delta$, we obtain

$$|H(\mathcal{A}(B_1)) - H(\mathcal{A}(B_2))| \leq 2\delta\log\left(\frac{|\mathcal{M}|}{\delta}\right).$$

We also note that

$$|H(\mathcal{A}(B_1) \mid B_1) - H(\mathcal{A}(B_2) \mid B_2)| \leq \sum_{z\in\mathcal{B}} |\Pr[B_1 = z] - \Pr[B_2 = z]| \cdot H(\mathcal{A}(z)) \leq 2\delta\log(|\mathcal{M}|).$$

Combining both inequalities completes the proof of Lemma F.2. □

Given the lemmas above, we are ready to complete the proof of Proposition C.5. Let $\tilde{B}$ be the approximation of B which is given by the SDPI assumption, and denote $M := \mathcal{A}(B),\ \tilde{M} := \mathcal{A}(\tilde{B})$. Applying Lemma F.2 and the ρ-SDPI assumption that holds for $(A, \tilde{B})$, we see that

$$I(M; B) \geq I(\tilde{M}; \tilde{B}) - 4\delta\log(|\mathcal{M}|/\delta) \geq \frac{1}{\rho} I(\tilde{M}; A) - 4\delta\log(|\mathcal{M}|/\delta)\ .$$

Further note that $d_{\mathrm{TV}}(\tilde{M}, M) \le d_{\mathrm{TV}}(\tilde{B}, B) \le \delta$, which implies $|H(\tilde{M}) - H(M)| \le 2\delta \log(|\mathcal{M}|/\delta)$ by Lemma F.1. Furthermore, by assumption, the same is true even after we condition on A. Therefore, we may conclude that

$$|I(\tilde{M}; A) - I(M; A)| \le |H(\tilde{M}) - H(M)| + |H(M \mid A) - H(\tilde{M} \mid A)| \le 4\delta \log(|\mathcal{M}|/\delta).$$

Plugged into the inequality above, we get that

$$I(M; B) \ge \frac{1}{\rho} \left(I(M; A) - 4\delta \log(|\mathcal{M}|/\delta) \right) - 4\delta \log(|\mathcal{M}|/\delta) \,,$$

or rearranged,

$$I(M; A) \le \rho I(M; B) + 4\delta(1+\rho) \log(|\mathcal{M}|/\delta) \le \rho I(M; B) + 8\delta \log(|\mathcal{M}|/\delta) \,.$$

F.2. Completing the Proof of Theorem C.7

We will show that if $\mathrm{err}(\mathcal{A}, \mathbf{P}) \le \alpha$, then

$$I(\mathcal{A}(X_{1:n}); X) \ge D_{\mathrm{KL}} \left(\mathrm{Ber}(1-\alpha) \,||\, \mathrm{Ber}(\alpha) \right) = (1-2\alpha) \log \left(\frac{1-\alpha}{\alpha} \right) \,.$$

Consider a random variable $X' \sim \mathcal{P}_0$ drawn independently of $X \sim \mathcal{P}_\theta$, $X_{1:n} \sim \mathcal{P}_\theta^n$ and $\mathcal{A}(X_{1:n})$. Since $\mathcal{P}_0$ is the marginal of $\mathcal{P}_\theta$ over θ, the characterization of the mutual information as the KL-divergence between the joint and product distributions shows that

$$\begin{aligned} I(\mathcal{A}(X_{1:n}); X) &= D_{\mathrm{KL}} \left(\mathcal{A}(X_{1:n}), X \,||\, \mathcal{A}(X_{1:n}), X' \right) \\ &\ge D_{\mathrm{KL}} \left(\mathcal{A}(X_{1:n})(X) \,||\, \mathcal{A}(X_{1:n})(X') \right) \\ &= D_{\mathrm{KL}} \left(\mathrm{Ber}(p) \,||\, \mathrm{Ber}(q) \right) \,, \end{aligned}$$

where the inequality follows because post-processing does not increase the KL divergence, and p, q denote the probability of the model classifying positive/negative points as 1, respectively. Now, the error of $\mathcal{A}$ is equal to $\frac{1-p}{2} + \frac{q}{2}$ and therefore the condition $\mathrm{err}(\mathcal{A}, \mathbf{P}) \le \alpha$ implies that $p - q \le 1 - 2\alpha$. Optimizing this expression (as in the proof of Fano's inequality, cf. Cover & Thomas, 1999) we get

$$I(\mathcal{A}(X_{1:n}); X) \ge D_{\mathrm{KL}} \left(\mathrm{Ber}(1-\alpha) \,||\, \mathrm{Ber}(\alpha) \right) = (1-2\alpha) \log \left(\frac{1-\alpha}{\alpha} \right) \,.$$

G. Proof of Gaussian Clustering Application

In this section, we prove Theorem D.1. We will establish the three claims one by one.

G.1. Gaussian sample complexity $(n = 1)$

Given a single sample $X_1 \sim \mathcal{P}_\theta$, we will show that the algorithm that returns the linear classifier

$$h(X) := \mathbb{1} \left\{ \langle X_1, X \rangle \ge \sqrt{4 \log(200) d} \right\}$$

has error at most 0.01. This will follow from standard Gaussian concentration bounds, since with high probability:

$$\begin{aligned} X \sim \mathcal{P}_\theta \quad &\Longrightarrow \quad \langle X, X_1 \rangle \gtrsim \lambda^2 d \gtrsim \sqrt{d} \,, \\ X \sim \mathcal{P}_0 \quad &\Longrightarrow \quad \mathbb{E} \langle X, X_1 \rangle = 0 \,, \ \text{ and hence } \ |\langle X, X_1 \rangle| \lesssim \sqrt{d} \,. \end{aligned}$$

Formally, start by noting that in the null case, X_1 and $X \sim \mathcal{P}_0$ are simply two independent isotropic Gaussians, and therefore a standard bound on their inner product ensures that

$$\Pr_{X \sim \mathcal{P}_0} [\langle X_1, X \rangle \ge \sqrt{4 \log(200) d}] \le e^{-4 \log(200)/4} = \frac{1}{200} \,.$$

On the other hand, if $X \sim \mathcal{P}_\theta$ then both X and X_1 can be seen as distributed as $X = \lambda\theta + g_0,\ X_1 = \lambda\theta + g_1$ where $g_0, g_1 \sim \mathcal{N}(0_d, (1-\lambda^2)I_d)$ are independent of one another as well as of θ. Hence,

$$\begin{aligned}\langle X, X_1\rangle &= \lambda^2\|\theta\|^2 + \lambda\langle\theta, g_0\rangle + \lambda\langle\theta, g_1\rangle + \langle g_0, g_1\rangle \\ &= \frac{C^2}{d^{1/2}}\|\theta\|^2 + \frac{C}{d^{1/4}}\langle\theta, g_0\rangle + \frac{C}{d^{1/4}}\langle\theta, g_1\rangle + \langle g_0, g_1\rangle\ .\end{aligned}$$

Since $1-\lambda^2 < 1$, applying the same argument as in the null case to the latter three summands and union bounding ensures that

$$\Pr_{\theta, g_0, g_1}[\min\{\langle\theta, g_0\rangle, \langle\theta, g_1\rangle, \langle g_0, g_1\rangle\} < -\sqrt{4\log(1200)d}] \le 3e^{-\log(1200)} = \frac{1}{400}\ .$$

Furthermore, a standard bound on the norm of a Gaussian vector (cf. Vershynin, 2018, Theorem 3.1.1) ensures that $\Pr[\|\theta\|^2 < \gamma d] < \frac{1}{400}$ for some absolute constant $\gamma > 0$. Union bounding over this event as well, we overall see that with probability at least $1 - \frac{1}{200}$:

$$\langle X, X_1\rangle \ge C^2\cdot\gamma d^{1/2} - 2C\sqrt{4\log(1200)}d^{1/4} - \sqrt{4\log(1200)d}\ .$$

For sufficiently large absolute constant C, the latter is larger than $\sqrt{4\log(200)d}$, and we see that the linear classifier h achieves error at most $2\cdot\frac{1}{200} = \frac{1}{100}$, completing the proof.

G.2. Gaussian memorization lower bound

In order to prove the lower bound, our goal is to establish that the conditions in Theorem C.7 hold. To do so, we use the dominating variables approach and apply Theorem C.8, without any need for approximation (i.e. $\delta_n = 0$). We construct the dominating random variables $Z_\theta^{\text{train}}, Z_\theta^{\text{test}}$ as

$$\begin{aligned}Z_\theta^{\text{train}} &= \mathcal{N}\left(\frac{\lambda\sqrt{n}}{\sqrt{1+(n-1)\lambda^2}}\cdot\theta, \frac{1-\lambda^2}{1+(n-1)\lambda^2}\cdot I_d\right), \\ Z_\theta^{\text{test}} &= \mathcal{N}(\lambda\theta, (1-\lambda^2)I_d).\end{aligned} \tag{2}$$

To see that these variables satisfy the ρ_n-SDPI, we will use Fact C.2, and therefore need to show the variables are $\sqrt{\rho_n}$-correlated unit Gaussians, for suitable ρ_n. To that end, recalling that $\theta \sim \mathcal{N}(0_d, I_d)$ and therefore marginalized over θ it holds that

$$\begin{aligned}Z_\theta^{\text{train}} &= \frac{\lambda\sqrt{n}}{\sqrt{1+(n-1)\lambda^2}}\mathcal{N}(0_d, I_d) + \mathcal{N}\left(0_d, \frac{1-\lambda^2}{1+(n-1)\lambda^2}\cdot I_d\right) \\ &= \mathcal{N}\left(0_d, \frac{\lambda^2 n}{1+(n-1)\lambda^2}\cdot I_d\right) + \mathcal{N}\left(0_d, \frac{1-\lambda^2}{1+(n-1)\lambda^2}\cdot I_d\right) \\ &= \mathcal{N}(0_d, I_d)\ ,\end{aligned}$$

and similarly

$$Z_\theta^{\text{test}} = \lambda\cdot\mathcal{N}(0_d, I_d) + \mathcal{N}(0_d, (1-\lambda^2)I_d) = \mathcal{N}(0_d, I_d)\ .$$

We see that $Z_\theta^{\text{train}}, Z_\theta^{\text{test}}$ are both unit Gaussians, with each coordinate $i \in [d]$ satisfying

$$\mathbb{E}\left[(Z_\theta^{\text{train}})_i\cdot(Z_\theta^{\text{test}})_i\right] = \frac{\lambda\sqrt{n}}{\sqrt{1+(n-1)\lambda^2}}\cdot\lambda =: \sqrt{\rho_n},$$

thus establishing the ρ_n-SDPI, as claimed.

We turn to argue about the existence of the desired mappings $\Phi^{\text{train}}, \Phi^{\text{test}}$. First note that Z_θ^{train} can be mapped to $\mathcal{N}(\lambda\theta, \frac{1-\lambda^2}{n}I_d)$ simply by rescaling by $\sqrt{\frac{n}{1+(n-1)\lambda^2}}$. Furthermore, it is known that $\mathcal{N}(\lambda\theta, \frac{1-\lambda^2}{n}I_d)$ can be processed to produce $X_1, \ldots, X_n \sim \mathcal{N}(\lambda\theta, (1-\lambda^2)I_d)$ since the average $\frac{1}{n}\sum_{i=1}^n X_i \sim \mathcal{N}(\lambda\theta, \frac{1-\lambda^2}{n}I_d)$ is a sufficient statistic of Gaussians with the same known variance (cf. Cover & Thomas, 1999, Section 2.9). By composition, this provides Φ^{train}.

Moreover, Φ^{test} is simply the identity since $Z_\theta^{\text{test}} \sim \mathcal{P}_\theta$. We therefore establish the test/train condition in Theorem C.8 with $\delta_n \equiv 0$.

To establish the data generation SDPI with $\epsilon_n \equiv 0$ as well, we once again use our construction of the post-processing Z_θ^{train} as in Eq. (2), and claim that θ and Z_θ^{train} satisfy the τ_n-SDPI for $\sqrt{\tau_n} = \frac{\lambda\sqrt{n}}{\sqrt{1+(n-1)\lambda^2}}$. This follows as we know that they are both unit Gaussians, and by noting that they are $\sqrt{\tau_n}$-correlated coordinate-wise by construction of Z_θ^{train}. Therefore, the dataset $X_{1:n}$ which we have shown to be a post-processing of Z_θ^{train} also satisfies τ_n-SDPI with respect to θ, since for every M such that $M \perp\!\!\!\perp \theta \,|\, X_{1:n}$ we can use the SDPI and the (regular) DPI to see that

$$I(M;\theta) \le \tau_n I(M; Z_\theta^{\text{train}}) \le \tau_n I(M; X_{1:n}).$$

Overall, by applying Theorem C.8, we see that the conditions of Theorem C.7 are met, and we get that for any algorithm $\mathcal{A}$ with $\text{err}(\mathcal{A}, \mathbf{P}_{\text{G}}) \le \alpha < \frac{1}{2}$

$$\text{mem}_n(\mathcal{A}, \mathbf{P}_{\text{G}}) \ge \frac{1-\tau_n}{\rho_n} C_\alpha = \frac{1 - \frac{\lambda^2 n}{1+(n-1)\lambda^2}}{\frac{\lambda^4 n}{1+(n-1)\lambda^2}} \cdot (1-2\alpha)\log\left(\frac{1-\alpha}{\alpha}\right) = \frac{1-\lambda^2}{\lambda^4 n} \cdot (1-2\alpha)\log\left(\frac{1-\alpha}{\alpha}\right) .$$

G.3. Gaussian memorization upper bound

Given a sample $X_{1:n} \sim \mathcal{P}_\theta^n$, we will show that there exists an algorithm that returns a good classifier with the claimed memorization upper bound. For that, we argue that an algorithm can return a high accuracy classifier $h = \mathcal{A}(X_{1:n})$ which is describable using $O(d\log(\frac{d}{n})/n)$ bits, hence in particular

$$\text{mem}_n(\mathcal{A}, \mathbf{P}_{\text{G}}) = I(h; X_{1:n} \mid \theta) \overset{(\star)}{\le} I(h; X_{1:n}) \le H(h) = O\left(\frac{d\log(\frac{d}{n})}{n}\right) , \tag{3}$$

where $(\star)$ is due to the fact that $h \perp\!\!\!\perp \theta \,|\, X_{1:n}$.

To construct the predictor we introduce some notation. Given a vector $v \in \mathbb{R}^d$, we denote by $v^{[1:\ell]} := (v_1, \dots, v_\ell, 0, \dots, 0) \in \mathbb{R}^d$ its projection onto the first ℓ coordinates embedded in $\mathbb{R}^d$. Let $\hat{X} := \frac{1}{n}\sum_{i=1}^n X_i$ be the dataset's empirical average, and note that

$$\hat{X} \sim \mathcal{N}\left(\lambda\theta, \frac{1-\lambda^2}{n} I_d\right) . \tag{4}$$

For $\hat{X}^{[1:\ell]} = (\hat{X}_1, \dots, \hat{X}_\ell, 0, \dots, 0) \in \mathbb{R}^d$, let $q_k(\hat{X}^{[1:\ell]}) \in \mathbb{R}^d$ be a quantization of $\hat{X}^{[1:\ell]}$ that stores only k bits of $\hat{X}^{[1:\ell]}$ in each coordinate's binary expansion, allowing up to magnitude of $O(\log(d/n))$. We consider the algorithm $\mathcal{A}$ that returns the linear classifier

$$h(X) := \mathbb{1}\left\{\langle X^{[1:\ell]}, q_k(\hat{X}^{[1:\ell]})\rangle \ge t\right\} .$$

We will argue that for suitable choices of

$$\ell = \Theta(d/n) , \quad k = \Theta\left(\log(\tfrac{d}{n})\right) , \quad t = \Theta(\sqrt{d/n}) ,$$

this classifier satisfies the desired properties. We note that this classifier is described by $\ell k = O(d\log(\frac{d}{n})/n)$ bits, so indeed (3) holds, proving the memorization upper bound.

It remains to show that h is a high accuracy classifier. To bound the classifier's error, recall (4) and note that on one hand for the null case $X \sim \mathcal{P}_0 = \mathcal{N}(0_d, I_d)$:

$$\begin{aligned}\langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle &= \langle X^{[1:\ell]}, \lambda\theta^{[1:\ell]}\rangle + \left\langle X^{[1:\ell]}, \mathcal{N}\left(0_\ell, \frac{1-\lambda^2}{n} I_\ell\right)\right\rangle \\ &= \mathcal{N}(0, \lambda^2\|\theta^{[1:\ell]}\|^2 + \|g\|^2) , \quad g \sim \mathcal{N}\left(0_\ell, \frac{1-\lambda^2}{n} I_\ell\right)\end{aligned}$$

where equality is in the distributional sense. Hence,

$$\mathbb{E}\langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle = 0 ,$$

and applying the general Hoeffding's inequality (Vershynin, 2018, Theorem 2.6.2) ensures that for some absolute constants $c_1, c_2 > 0$:

$$\begin{aligned}\Pr\left[\langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle \geq \frac{t}{2}\right] &\leq \exp\left(-\frac{c_1 t^2}{\lambda^2\|\theta^{[1:\ell]}\|^2 + \|g\|^2}\right) \\ &\overset{(1)}{\leq_p} \exp\left(-\frac{c_2 t^2}{\ell/d^{1/2} + \ell/n}\right) \\ &\overset{(2)}{\leq} 0.001\ , \end{aligned} \tag{5}$$

where $\overset{(1)}{\leq_p}$ holds with high probability (say, at least 0.999) for suitable c_2 since $\|\theta^{[1:\ell]}\|^2 \lesssim \ell$ and $\|g\|^2 \lesssim \ell \cdot \frac{1-\lambda^2}{n} \leq \frac{\ell}{n}$, and $\overset{(2)}{\leq}$ holds for a suitable assignment of t, ℓ as above, under our assumption that $d \gtrsim n^2$.

It remains to further bound the error induced by quantization. Note that a standard bound on the maximum of independent normal variables, each of the first ℓ coordinates of $\hat{X}^{[1:\ell]}$ is bounded by $\leq 1000\sqrt{2\log \ell}$ with probability at least 0.999. Since we are storing the bits allowing up to a magnitude of $O(\log(d/n))$, under this probable event the quantization error is only due to the bits erased at the right-end of the binary expansion. It therefore holds that

$$\begin{aligned}\langle X^{[1:\ell]}, q_k(\hat{X}^{[1:\ell]})\rangle &= \langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle + \langle X^{[1:\ell]}, q_k(\hat{X}^{[1:\ell]}) - \hat{X}^{[1:\ell]}\rangle \\ &\leq \langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle + \|X^{[1:\ell]}\| \cdot \|q_k(\hat{X}^{[1:\ell]}) - \hat{X}^{[1:\ell]}\| \\ &\leq \langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle + \|X^{[1:\ell]}\| \cdot \sqrt{\ell}\underbrace{\|q_k(\hat{X}^{[1:\ell]}) - \hat{X}^{[1:\ell]}\|_\infty}_{\text{coordinate-wise quantization error}} \\ &\leq \langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle + \|X^{[1:\ell]}\| \cdot \sqrt{\ell} 2^{\log_2 O(\sqrt{\log \ell}) - k} \\ &\leq \langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle + \|X^{[1:\ell]}\| \cdot \sqrt{\ell \log \ell} \cdot 2^{-k}\ , \end{aligned}$$

and further applying a Gaussian norm bound that ensures that $\Pr[\|X^{[1:\ell]}\| \geq \widetilde{C}\sqrt{\ell}] < 0.0001$ for absolute constant $\widetilde{C} > 0$, we see that the additional term is negligible for our setting of k. Union bounding with (5) and overall obtain

$$\Pr_{X\sim\mathcal{P}_0}[h(X) = 0] = \Pr_{X\sim\mathcal{P}_0}[\langle X, q_k(\hat{X}^{[1:\ell]})\rangle < t] > 0.995\ . \tag{6}$$

As to $X \sim \mathcal{P}_\theta = \mathcal{N}(\lambda\theta, (1-\lambda^2)I_d)$, note that

$$\mathbb{E}\langle X^{[1:\ell]}, \hat{X}^{[1:\ell]}\rangle = \lambda^2\, \mathbb{E}\, \|\theta^{[1:\ell]}\|^2 = \lambda^2\ell = C^2 \cdot \Theta\left(t\right)\ .$$

Thus, similar concentration and quantization arguments as in the null case yields for sufficiently large absolute constant $C > 0$ (in the problem definition):

$$\Pr_{X\sim\mathcal{P}_\theta}[h(X) = 1] = \Pr_{X\sim\mathcal{P}_\theta}[\langle X^{[1:\ell]}, q_k(\hat{X}^{[1:\ell]})\rangle > t] > 0.995\ . \tag{7}$$

By union bounding (6) and (7) we get overall that

$$\mathrm{err}(\mathcal{A}, \mathbf{P}_{\mathrm{G}}) < 0.01\ ,$$

which completes the proof.

H. Proof of Boolean Clustering Application

In this section, we prove Theorem D.3.

H.1. Boolean sample complexity $(n = 1)$

Consider the algorithm that given $X_1 \sim \mathcal{P}_\theta$ returns the predictor

$$h(X) := \mathbb{1}\left\{\langle X_1, X\rangle \geq \sqrt{2\log(200)d}\right\}\ .$$

To see why this predictor suffers from error at most 0.01, first note that for $X \sim \mathcal{P}_0$: $\mathbb{E}\langle X_1, X\rangle = 0$, and by Hoeffding's inequality:

$$\Pr[\langle X_1, X\rangle \geq \sqrt{2\log(200)d}] \leq \frac{1}{200}\,. \tag{8}$$

On the other hand, for $X \sim P_\theta$:

$$\mathbb{E}\langle X_1, X\rangle = \sum_{i=1}^{d} \mathbb{E}[(X_1)_i \cdot (X)_i] = d \cdot \mathbb{E}[(X_1)_1] \cdot \mathbb{E}[(X)_1]\,.$$

Note that on one hand $\mathbb{E}[(X)_1] = \lambda\theta_1$, or equivalently $\theta_1\,\mathbb{E}[(X)_1] = \lambda$. Plugging this into the equation above, we see that

$$\mathbb{E}\langle X_1, X\rangle = d\theta_1^2\,\mathbb{E}[(X_1)_1] \cdot \mathbb{E}[(X)_1] = d\lambda^2 = C^2\sqrt{d}\,.$$

A similar calculation also shows that $\mathrm{Var}\langle X_1, X\rangle = d\mathrm{Var}((X_1)_1)\mathrm{Var}((X)_1) \leq d$, following from Popoviciu's variance bound. Therefore, Chebyshev's inequality yields

$$\Pr[\langle X_1, X\rangle \leq (C^2 - \sqrt{200})\sqrt{d}] \leq \frac{1}{200}\,. \tag{9}$$

Assuming C is sufficiently large so that $C^2 - \sqrt{200} > \sqrt{2\log(200)}$ we see by union bounding over (8) and (9) that the defined predictor h indeed classifies between samples from $\mathcal{P}_\theta$ and $\mathcal{P}_0$ with error at most $\frac{1}{200} + \frac{1}{200} = \frac{1}{100}$, as claimed.

H.2. Boolean memorization lower bound

Once again, we will follow the framework of Theorem C.7 to prove Item 2 of Theorem D.3.

Binomials and advanced composition

To start, we state and prove the following lemma.

Lemma H.1. *Let $\lambda \in (0, 1/10)$ and $n \in \mathbb{N}$ be such that $n < \frac{1}{10\lambda^2}$. Let X be a product distribution over $\{\pm 1\}^n$ where for each coordinate i it holds that $\Pr[X_i = 1] = \frac{1+\lambda}{2}$. Similarly let Y be a product distribution over $\{\pm 1\}^n$ where for each $i \in [n]$ it holds that $\Pr[Y_i = 1] = \frac{1-\lambda}{2}$. Then, for every $\delta \in (0,1)$ and $\rho = \sqrt{8n\log(1/\delta)}\lambda$, there exists a pair of distributions $\mathbf{A}, \mathbf{B}$ over $\{\pm 1\}^n$ such that*

$$d_{\mathrm{TV}}(X, \frac{1+\rho}{2}\mathbf{A} + \frac{1-\rho}{2}\mathbf{B}) < \delta,$$
$$d_{\mathrm{TV}}(Y, \frac{1-\rho}{2}\mathbf{A} + \frac{1+\rho}{2}\mathbf{B}) < \delta.$$

Here, $p\mathbf{A} + (1-p)\mathbf{B}$ denotes the mixture of two distributions with weights p and $(1-p)$.

Lemma H.1 tells us how to post-process a Bernoulli random variable with bias $\frac{1}{2} \pm \rho$, to approximate a sequence of n independent Bernoulli random variables, each of bias $\frac{1}{2} \pm \lambda$ where $\lambda \approx \frac{\rho}{\sqrt{n}}$. This lemma appears to be folklore in the differential privacy literature, yet for completeness, we include a brief proof below. First, we need the notion of "indistinguishability" between distributions.

Definition H.2. Let X, Y be a pair of random variables with the same support Ω. We say that X and Y are (ε, δ)-indistinguishable, if for every measurable $E \subseteq \Omega$, it holds that

$$\Pr[X \in E] \leq e^\varepsilon \Pr[Y \in E] + \delta,$$
$$\Pr[Y \in E] \leq e^\varepsilon \Pr[X \in E] + \delta.$$

We use the following "decomposition" lemma for indistinguishable distributions, the proof of which is usually attributed to (Kairouz et al., 2015).

Proposition H.3 ((Kairouz et al., 2015))**.** *Suppose X, Y are (ε, δ)-indistinguishable. Then, there exists four distributions X', Y', E_1, E_2 such that*

$$X = (1-\delta)X' + \delta E_1,$$
$$Y = (1-\delta)Y' + \delta E_2,$$

and X', Y' are $(\varepsilon, 0)$-indistinguishable. Furthermore, there are two distributions U, V such that

$$X' = \frac{e^\varepsilon}{1+e^\varepsilon} U + \frac{1}{1+e^\varepsilon} V,$$
$$Y' = \frac{e^\varepsilon}{1+e^\varepsilon} V + \frac{1}{1+e^\varepsilon} U.$$

In light of Proposition H.3, to prove Lemma H.1, it suffices to show that the distributions X, Y in the statement are (ρ, δ)-indistinguishable.

In the language of differential privacy, X and Y can be understood as the n-wise composition of the randomized response mechanism,[4] where each individual RR mechanism enjoys $(\lambda + o(1), 0)$-DP. The advanced composition theorem of differential privacy then says the following:

Proposition H.4 ((Dwork et al., 2010; Kairouz et al., 2015))**.** *Suppose $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are n $(\varepsilon, 0)$-DP algorithms operating on $b \in \{\pm 1\}$ and $n < \frac{1}{5\varepsilon^2}$. Then, their composition $(\mathcal{A}_1 \circ \cdots \circ \mathcal{A}_n)$ satisfies $(\sqrt{4n\log(1/\delta)}\varepsilon, \delta)$-DP for every $\delta \in (0,1)$. In other words, $(\mathcal{A}_1 \circ \cdots \circ \mathcal{A}_n)(-1)$ and $(\mathcal{A}_1 \circ \cdots \circ \mathcal{A}_n)(+1)$ are $(\sqrt{4n\log(1/\delta)}\varepsilon, \delta)$-indistinguishable.*

Finally, combining Propositions H.4 and H.3 concludes the proof of Lemma H.1.

The mutual information bound

We are ready to establish the second item of Theorem D.3. Recall the setting of $\mathbf{P}_{\mathrm{B}}$: a random $\theta \sim \{\pm 1\}^d$ is drawn. The input distribution $\mathcal{P}_\theta$ is the binary symmetric channel $\mathrm{Bin}_{\frac{1-\lambda}{2}}(\theta)$. Namely, to draw $y \sim \mathcal{P}_\theta$, for each coordinate $i \in [d]$ set $y_i = \theta_i$ with probability $\frac{1+\lambda}{2}$, and set $y_i = -\theta_i$ otherwise. Next, we verify that the two conditions of Theorem C.7 are met for $\mathbf{P}_{\mathrm{B}}$.

Data generation SDPI. Conditioning on $\theta \in \{\pm 1\}^d$, for each $i \in [d]$, the i-th coordinates of $X_{1:n}$ are n independent draws from a Bernoulli distribution of bias $\frac{1}{2} + \lambda \cdot \theta_i$. As such, we apply Lemma H.1 with some $\delta > 0$ to be specified. Namely, we draw a single Z_i from $\mathrm{Ber}(\frac{1}{2} + \xi\theta_i)$ with $\xi = \sqrt{8n\log(1/\delta)} \cdot \lambda$, and post-process Z_i as in Lemma H.1, to obtain a collection of n Boolean variables $(\tilde{X}_{1,i}, \ldots, \tilde{X}_{n,i})$, such that (over the randomness of Z_i and the post-processing) the TV distance between $(X_{1,i}, \ldots, X_{n,i})$ and $(\tilde{X}_{1,i}, \ldots, \tilde{X}_{n,i})$ is at most δ. We run the same argument for each coordinate $i \in [d]$. Overall, we have shown how to post-process a random variable $Z \sim \mathrm{Bin}_{\frac{1-\xi}{2}}(\theta)$ to obtain a sequence of n samples $\tilde{X}_{1:n}$ such that the TV distance between $\tilde{X}_{1:n}$ and $X_{1:n}$ is bounded by δd. Using Item 1 of Theorem C.8, this implies that the pair $(\theta, X_{1:n})$ satisfies $(\xi^2, \delta d)$-SDPI where $\xi = \sqrt{8n\log(d/\delta)} \cdot \lambda$ and $\delta \in (0,1)$ can be arbitrarily chosen.

Test/train SDPI. We now verify the second condition. Consider the "Z" random variable we have introduced. We have shown a post-processing mapping Φ^{train} such that $\Phi^{\mathrm{train}}(Z)$ is close to $X_{1:n}$ in TV distance. We can also draw $X \sim \mathcal{P}_\theta$ and define Φ^{test} as the identity mapping. It remains to show that Z and X satisfy ρ_n-SDPI for a small ρ_n. Indeed, note that θ is uniform over $\{\pm 1\}^d$. Conditioned on θ, we have $Z \sim \mathrm{Bin}_{\frac{1-\xi}{2}}(\theta)$ and $X \sim \mathrm{Bin}_{\frac{1-\lambda}{2}}(\theta)$, and they are independent conditioning on θ. Therefore, by the symmetry of the binary symmetric channel, it follows that $Z \sim \mathrm{Bin}_{\frac{1-\lambda\xi}{2}}(X)$, which implies that Z and X satisfy ρ_n-SDPI with $\sqrt{\rho_n} = \lambda\xi = \widetilde{O}(\lambda^2\sqrt{n}) = \widetilde{O}(\sqrt{\frac{n}{d}})$. Consequently, by Item 2 of Theorem C.8, we conclude that the pair $(X, X_{1:n})$ satisfies $(\lambda^2\xi^2, \delta d)$-SDPI, where we recall that $\xi^2\lambda^2 = \widetilde{O}(\lambda^4 n) = \widetilde{O}(\frac{n}{d})$.

Conclusion. We have established both conditions of Theorem C.7. Using Theorem C.7, this yields that for any $\delta \in (0,1)$:

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = \Omega\left(\frac{1 - \lambda^2 n\log(d/\delta)}{\lambda^4 n\log(d/\delta)}\left(C_\alpha - \delta d\log(|\mathcal{M}|/\delta)\right)\right),$$

which by setting $\delta = \min\{(\frac{C_\alpha}{d\log(|\mathcal{M}|)})^2, \frac{1}{2}\}$ results in

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = \Omega\left(\frac{1 - \lambda^2 n\log(d\log|\mathcal{M}|)}{\lambda^4 n\log(d\log|\mathcal{M}|)} C_\alpha\right).$$

[4] Recall that the $(\varepsilon, 0)$-DP randomized response (RR) mechanism receives an input $b \in \{\pm 1\}$, and outputs a random bit $\hat{b}$ such that $\Pr[\hat{b} = b] = \frac{e^\varepsilon}{1+e^\varepsilon}$.

In particular, as long as $\lambda^2 n \le \frac{1}{2}$, or equivalently $n \le \sqrt{d}/C^2$, and if $\log|\mathcal{M}| \in \mathrm{poly}(d)$, or equivalently $|\mathcal{M}| \le \exp(d^{\tilde{C}})$ for some absolute constant $\tilde{C} > 0$, then

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = \widetilde{\Omega}\left(\frac{1-\lambda^2 n}{\lambda^4 n} C_\alpha\right) = \widetilde{\Omega}\left(\frac{d}{n}\right).$$

H.3. Boolean memorization upper bounds

Here, we describe algorithms to complement the lower bound and establish Item 3 of Theorem D.3. We split the proof into the two considered regimes $n \lesssim \sqrt{d}$ and $n \gtrsim \sqrt{d}\log d$.

ALGORITHM WHEN $n \lesssim \sqrt{d}$

We will show that when $1 \le n \le \sqrt{d}$ examples are available, we can design an algorithm $\mathcal{A}$ which learns an accurate model h such that $I(h; X_{1:n}) = O\left(\frac{d}{n}\right)$.

Let $C_{sub} > 0$ be a sufficiently large constant. On input n i.i.d. samples $X_{1:n} \sim \mathcal{P}_\theta^n$, let $\hat{\theta}$ be the bit-wise majority vote of $X_{1:n}$. Our algorithm chooses $t = \min\left(d, \frac{Cd}{n}\right)$ and returns the hypothesis

$$h(X) := \mathbb{1}\left\{\langle \hat{\theta}^{[1,t]}, X^{[1,t]}\rangle \ge \sqrt{2\log(200)t}\right\}.$$

Here, recall that for a vector $v \in \mathbb{R}^d$, the notation $v^{[1,t]} = (v_1, \ldots, v_t, 0, \ldots, 0)$ denotes its projection onto the first t coordinates embedded in $\mathbb{R}^d$.

We prove that h is, with high probability, a good predictor. First, for $X \sim \mathcal{P}_0$, the same argument as in Section H.1 proves that $\Pr_{X\sim\mathcal{P}_0}[h(X) = 1] \le \frac{1}{200}$.

It remains to show that h classifies most $X \sim \mathcal{P}_\theta$ correctly. Indeed, for every coordinate $j \in [d]$, we have

$$\Pr[\hat{\theta}_j = \theta_j] \ge \frac{1}{2} + C'\sqrt{n}\lambda$$

for some constant C'. Also,

$$\Pr[X_j = \theta_j] = \frac{1}{2} + \lambda$$

These two combined would imply that

$$\mathbb{E}\langle \hat{\theta}^{[1,t]}, X^{[1,t]}\rangle \ge C'\lambda^2\sqrt{n}t = C''\sqrt{t}.$$

Similarly as in Section H.1, we have

$$\mathrm{Var}\left[\langle \hat{\theta}^{[1,t]}, X^{[1,t]}\rangle\right] \le t.$$

Therefore, Chebyshev's inequality gives

$$\Pr[\langle \hat{\theta}^{[1,t]}, X^{[1,t]}\rangle \le ((C'')^2 - \sqrt{200})\sqrt{t}] \le \frac{1}{200}.$$

By setting C_{sub} to be large enough, we can guarantee that $(C'')^2 - \sqrt{200} \ge \sqrt{2\log(200)}$, which would consequently yield that $\Pr_{X\sim\mathcal{P}_\theta}[h(X) = 0] \le \frac{1}{200}$. This completes the accuracy analysis of h.

Regarding memorization, observe that h can be described using $O(\frac{Cd}{n})$ bits. Hence, we have

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) \le I(h; X_{1:n}) \le H(h) \le O\left(\frac{d}{n}\right),$$

as desired.

ALGORITHM FOR $n \gtrsim \sqrt{d}\log d$

We now provide an accurate algorithm for which $\mathrm{mem}_n(\mathcal{A}, \mathbf{P}) = O(d^2\exp(-d/\sqrt{n}))$, and so if $n \geq \tilde{C}\sqrt{d}\log d$, then $\mathcal{A}$ learns an accurate model h with nearly no excess memorization, particularly, such that $I(h; X_{1:n}) = O(d^{-\tilde{C}/2})$.

Similarly to the previously described algorithm, given n i.i.d. samples $X_{1:n} \sim \mathcal{P}_\theta^n$, let $\hat{\theta}$ be the bit-wise majority vote of $X_{1:n}$. Our algorithm returns the hypothesis

$$h_{\hat{\theta}}(X) := \mathbb{1}\left\{\langle \hat{\theta}, X\rangle \geq \sqrt{2\log(200)d}\right\}.$$

Noting that this is the same predictor as before but with no projection, it is easy to verify that the accuracy analysis follows from the same calculation as in the previous case. We will therefore prove the memorization bound, which follows from different arguments.

By definition,

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = \mathop{\mathbb{E}}_{\theta\sim\mathcal{U}(\{\pm1\}^d)}\left[I(h_{\hat{\theta}} \mid \theta; S \mid \theta)\right] \leq \mathop{\mathbb{E}}_{\theta\sim\mathcal{U}(\{\pm1\}^d)}\left[H(h_{\hat{\theta}} \mid \theta)\right].$$

We will show that for any θ, it holds that $H(h_{\hat{\theta}} \mid \theta) \lesssim d^{-\Omega(\tilde{C})}$, thus implying the claimed bound. Fix θ, and let $\gamma = \Pr_{X_{1:n}}[h_{\hat{\theta}} \neq h_\theta]$ which is the probability that the bit-wise majority vote did not reconstruct θ. By Hoeffding's bound, for any coordinate $j \in [d] : \Pr_{X_{1:n}}[\hat{\theta}_j \neq \theta_j] \leq \exp(-n\lambda^2/2)$ so by union bounding we see that $\gamma \leq d\exp(-n\lambda^2/2)$.

Considering the Markov chain $\theta \to X_{1:n} \to h_{\hat{\theta}}$, Fano's inequality shows that

$$\begin{aligned}
H(h_{\hat{\theta}} \mid \theta) &\leq H_2(\gamma) + \gamma\log(|\{\pm1\}^d| - 1) \\
&< H_2(\gamma) + \gamma d \\
&\leq 2\sqrt{\gamma} + \gamma d \\
&\leq 2\sqrt{d}\exp(-n\lambda^2/4) + d^2\exp(-n\lambda^2/2) \\
&\leq 3d^2\exp(-n\lambda^2/2),
\end{aligned}$$

where we used the easily verifiable numerical bound $H_2(\gamma) \leq 2\sqrt{\gamma}$. Overall we see that

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{B}}) = O(d^2\exp(-n\lambda^2/2)),$$

which under our parameter regime $\lambda = C/d^{1/4}$, yields the claimed bound.

I. Proof of Sparse Boolean Hypercube Application

In this section, we prove Theorem D.4.

I.1. Sparse Boolean sample complexity $(n = 1)$

Recall that in $\mathbf{P}_{\mathrm{sB}}$, the learning problem is parameterized by $\theta = (S, y)$ where S is a random subset including each $i \in [n]$ with probability $\nu \approx \frac{1}{d^{1/2}}$ and $y \sim \{\pm1\}^d$ is uniformly at random. The distribution $\mathcal{P}_\theta$ is a product distribution X on $\{\pm1\}^d$ where $X_S = y_S$ with probability one and $X_{\overline{S}}$ is uniformly at random.

We claim that learning with a single samples is possible so long as the set S satisfies that $|S| \geq \sqrt{4\log(200)d}$. First, note that this event happens with probability at least $1 - \exp(-\Omega(\sqrt{d}))$ if we choose $\nu = \frac{C}{d^{1/2}}$ for a large enough constant C. Now, suppose we have $|S| \geq \sqrt{4\log(200)d}$ and let X_1 be the training sample. Consider the hypothesis

$$h(X) := \mathbb{1}\left\{\langle X, X_1\rangle \geq \sqrt{2\log(200)d}\right\}.$$

We verify that h has a low generalization error. To see this, we first observe that for $X \sim \mathcal{P}_0$, it holds $\mathbb{E}[\langle X_1, X\rangle] = 0$, and by Hoeffding's bound, we further have $\Pr[\langle X_1, X\rangle \geq \sqrt{2\log(200)d}] \leq 0.005$. On the other hand, conditioned on θ, for every $X \sim \mathcal{P}_\theta$, we have that $X_S = (X_1)_S$ and $X_j \sim \{\pm1\}$ for every $j \notin S$. In this case, it is easy to see that $\langle X, X_1\rangle$ is a random variable with mean $|S| \geq \sqrt{2\log(200)d}$ and standard deviation $O(\sqrt{d - |S|})$. Similarly to the proof for the case of $\mathbf{P}_{\mathrm{B}}$, we see that $\Pr_{X\sim\mathcal{P}_\theta}[\langle X, X_1\rangle \leq \sqrt{2\log(200)d}] \leq 0.005$. Overall, we see that the predictor h has error at most 0.01 as a classifier between $\mathcal{P}_\theta$ and $\mathcal{P}_0$, as desired.

I.2. Sparse Boolean memorization lower bound

In this section, we establish Item 2 of Theorem D.4. We will deviate slightly from the framework of Theorem C.7. In particular, we will establish a lower bound on $I(A(X_{1:n}); X_{1:n})$ via the test/train SDPI as in Theorem C.7, however, we will *not* prove a data generation SDPI. Instead, we directly upper bound $I(X_{1:n}; \theta)$ by the Shannon entropy of θ, which is at most $O(\sqrt{d}\log(d))$. Then, a lower bound on $I(A(X_{1:n}); X_{1:n} \mid \theta)$ follows easily.

Suppose $\theta \sim \Psi$ and $(X, X_{1:n}) \sim \mathcal{P}_\theta^{n+1}$. We show that the pair $(X, X_{1:n})$ satisfy ρ_n-SDPI with $\rho_n = \Theta(\frac{2^n}{d^{1/2}})$. We begin with a basic observation: recall that the parameter θ consists of a pair (S, x_S) where each element is included in S with probability $\nu = \Theta(d^{1/2})$. Here, conditioning on X would *not* change the distribution of S: namely, it is still the case that each i is in S with probability ν. Furthermore, if $i \in S$ then we know the i-th bits of the training samples (namely, $(X_{j,i})_{j=1}^n$) all agree with $X_{0,i}$. Otherwise, $(X_{j,i})_{j=1}^n$ is uniformly distributed in $\{\pm 1\}^n$.

With this observation in mind, we introduce a random variable $Z \sim \mathrm{Bin}_{\frac{1}{2}+\xi}(X)$ for some ξ to be specified. Consider the following post-processing of Z that produces sequence of samples $\tilde{X}_1, \ldots, \tilde{X}_n$:

- Independently for each coordinate $i \in [d]$, with probability $\nu + (1-\nu)2^{-n+1}$, set all of $\tilde{X}_{1,i}, \ldots, \tilde{X}_{n,i}$ as Z_i. Otherwise choose $(\tilde{X}_{1,i}, \ldots, \tilde{X}_{n,i}) \sim \mathcal{U}(\{\pm 1\}^n \setminus \{(-1)^n, (+1)^n\})$ uniformly at random.

We argue that, for a proper choice of ξ, the joint distribution of $(X, \tilde{X}_{1:n})$ is identical to that of $(X, X_{1:n})$. Indeed, for each $i \in [d]$, the probability that all of $(X_{j,i})_{j=1}^n$ are the same is $\nu + (1-\nu)2^{-n+1}$, which is also the case for $\tilde{X}_{1:n}$. Furthermore, conditioning on the event that $(X_{j,i})_{j=1}^n$ are not all-0 nor all-1, they are uniformly distributed in $\{\pm 1\}^n \setminus \{(-1)^n, (+1)^n\}$, which is, again, also the case for $\tilde{X}_{1:n}$.

We turn to the case that all of $(X_{j,i})_{j=1}^n$ are the same. It is easy to calculate that the probability of the event $(X_{j,i})_{j=1}^n = (X_{0,i})^n$ is $\nu + (1-\nu)2^{-n}$, while the same probability term w.r.t. $\tilde{X}$ evaluates to $(\frac{1}{2} + \xi) \cdot (\nu + (1-\nu)2^{-n+1})$. It remains to set ξ properly so that the two probability quantities coincide. Indeed, this gives

$$\xi = \frac{\nu + (1-\nu)2^{-n}}{\nu + (1-\nu)2^{-n+1}} - \frac{1}{2} = \frac{\nu}{2\nu + (1-\nu)2^{-n+2}}.$$

Recall that we set $\nu = \Theta(d^{-1/2})$. Therefore, for all $n \le o(\log d)$, the expression simplifies to $\xi = \Theta(\nu 2^n)$.

We have shown that with the proper choice of ξ, we can define $Z \sim \mathrm{Bin}_{\frac{1}{2}+\xi}(X)$ and post-process Z to obtain a sequence of samples $\tilde{X}_{1:n}$ identically distributed as $X_{1:n}$. This implies that the pair $(X, X_{1:n})$ enjoys at least the same SDPI as the pair (X, Z). Quantitatively, the SDPI parameter we achieve is $\rho_n = \xi^2 = \Theta(\nu^2 2^{2n}) = \Theta(\frac{2^{2n}}{d})$. This consequently implies that

$$I(\mathcal{A}(X_{1:n}); X_{1:n}) \ge \frac{1}{\rho_n} I(\mathcal{A}(X_{1:n}); X) = \Omega\left(\frac{d}{2^{2n}}(1-2\alpha)\log\left(\frac{1-\alpha}{\alpha}\right)\right),$$

as claimed. For all $n \le o(\log d)$, this lower bound is far above $\sqrt{d}\log d$. Hence,

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) = I(\mathcal{A}(X_{1:n}); X_{1:n} \mid \theta) \ge I(\mathcal{A}(X_{1:n}); X_{1:n}) - H(\theta) = \Omega\left(\frac{d}{2^{2n}}\right).$$

I.3. Sparse Boolean memorization upper bound

Similarly to the other memorization upper bound proofs, given a sample $X_{1:n} \sim \mathcal{P}_\theta^n$, we will argue that an algorithm can return a high accuracy classifier $h = \mathcal{A}(X_{1:n})$ which is describable using $\widetilde{O}(d/2^{2n})$ bits, hence in particular

$$\mathrm{mem}_n(\mathcal{A}, \mathbf{P}_{\mathrm{sB}}) = I(h, X_{1:n} \mid \theta) \overset{(h \perp\!\!\!\perp \theta \mid X_{1:n})}{\le} I(h, X_{1:n}) \le H(h) = \widetilde{O}\left(\frac{d}{2^{2n}}\right).$$

Let $\hat{S} \subset [d]$ be the subset that includes coordinates $j \in [d]$ in which the dataset is constant, namely for which $(X_i)_j = (X_{i'})_j$ is the same for all $i, i' \in [n]$. Furthermore, let $\hat{S}^\ell$ be the first ℓ coordinates in $\hat{S}$, and define

$$h(X) = \mathbb{1}\left\{\langle X_{\hat{S}^\ell}, (X_1)_{\hat{S}^\ell}\rangle \ge t\right\} = \mathbb{1}\left\{\sum_{j \in \hat{S}^\ell} X_j \cdot (X_1)_j \ge t\right\}.$$

We will argue that for suitable

$$\ell = \Theta\left(\frac{d}{2^{2n}}\right), \quad t = \sqrt{2\log(200)\ell} = \Theta(\sqrt{\ell}),$$

this classifier satisfies the desired properties. We note that the memorization upper bound readily follows by the description length argument above, since all that needs to be stored are the ℓ bits of $(X_1)_{\hat{S}^\ell}$ alongside their corresponding indices in $[d]$, which requires $O(\ell \log(d))$ memory.

It remains to show that h is a high accuracy classifier. We first note that in the null case $\mathcal{P}_0 = \mathcal{U}(\{\pm 1\}^d)$, so it holds that

$$\mathop{\mathbb{E}}_{X\sim\mathcal{P}_0} \langle X_{\hat{S}^\ell}, (X_1)_{\hat{S}^\ell}\rangle = 0\,,$$

and by Hoeffding's bound

$$\Pr\left[\langle X_{\hat{S}^\ell}, (X_1)_{\hat{S}^\ell}\rangle \geq t\right] = \Pr\left[\langle X_{\hat{S}^\ell}, (X_1)_{\hat{S}^\ell}\rangle \geq \sqrt{2\log(200)\ell}\right] \leq 0.005\,. \tag{10}$$

We now turn to argue about the case $X \sim \mathcal{P}_\theta$. Note that for any coordinate $j \in \hat{S}^\ell$, if $j \in S$ then $X_j = (X_1)_j$ with probability 1, yet if $j \notin S$ then $X_j \cdot (X_1)_j \sim \mathcal{U}(\{\pm 1\})$. Hence, we see that

$$\mathop{\mathbb{E}}_{X\sim\mathcal{P}_\theta} \langle X_{\hat{S}^\ell}, (X_1)_{\hat{S}^\ell}\rangle = |\hat{S}^\ell \cap S|\,. \tag{11}$$

Therefore, we set out to estimate the size of $\hat{S}^\ell \cap S$. This set corresponds to coordinates in $\hat{S}^\ell$ which are "truly" constant, as opposed to uniformly random coordinates that happened to be constant on all seen examples, even thought they are not in S. Accordingly, since $S \subset \hat{S}$, for any coordinate $j \in [d]$:

$$\Pr[j \in S \mid j \in \hat{S}] = \frac{\Pr[j \in \hat{S} \cap j \in S]}{\Pr[j \in \hat{S}]} = \frac{\Pr[j \in S]}{\Pr[j \in \hat{S}]} = \frac{\nu}{\nu + (1-\nu)2^{-n+1}}\,.$$

Moreover, by symmetry, choosing the first ℓ coordinates in $\hat{S}$ into $\hat{S}^\ell$ is equivalent to sub-sampling ℓ coordinates among the constant coordinates, and therefore

$$\mathbb{E}\,|\hat{S}^\ell \cap S| = \ell \cdot \Pr[j \in S \mid j \in \hat{S}] = \ell \cdot \left(\frac{\nu}{\nu + (1-\nu)2^{-n+1}}\right) \gtrsim_{(1)} \ell \cdot \frac{2^n}{\sqrt{d}} \geq_{(2)} C' \cdot \sqrt{\ell}\,,$$

where (1) follows from the assumption that $n = O(\log d)$, and (2) follows for any constant C' of our choice provided that $\ell = C'' \frac{d}{2^{2n}}$ for a large enough C''. For sufficiently large C', Hoeffding's bound further ensures that

$$\Pr[|\hat{S}^\ell \cap S| > 3t] > 0.999\,,$$

and therefore under this probable event and applying yet another Hoeffding bound over $\langle X_{\hat{S}^\ell}, (X_1)_{\hat{S}^\ell}\rangle$, (11) ensures that

$$\Pr_{X\sim\mathcal{P}_\theta}[\langle X_{\hat{S}^\ell}, (X_1)_{\hat{S}^\ell}\rangle \geq t] > 0.995\,.$$

Overall, combined with (10), this ensures that the predictor h has classification error of at most 0.01, completing the proof.

J. Proofs for Section E

J.1. Proof of Theorem E.3

The proof follows that of Feldman (2020, Theorem 2.3). We can write

$$\begin{aligned}
\mathrm{err}(\mathcal{A}, \mathbf{P}^{\mathsf{mult}} \mid S) &= \mathop{\mathbb{E}}_{\pi\sim\Pi_p^k, \theta_{1:k}\sim\Psi^k} \mathop{\mathbb{E}}_{S\sim\widetilde{\mathcal{P}}_{\theta_{1:k},\pi}, h\leftarrow\mathcal{A}(S)} \left[\frac{1}{2}\Pr_{X\sim\mathcal{P}_{\theta_{1:k},\pi}}[h(X)=0] + \frac{1}{2}\Pr_{X\sim\mathcal{P}_0}[h(X)=1] \,\middle|\, S\right] \\
&= \frac{1}{2}\mathop{\mathbb{E}}_{\pi\sim\Pi_p^k, \theta_{1:k}\sim\Psi^k} \mathop{\mathbb{E}}_{S\sim\widetilde{\mathcal{P}}_{\theta_{1:k},\pi}, h\leftarrow\mathcal{A}(S)} \left[\sum_{j\sim\pi}\Pr_{X\sim\mathcal{P}_{\theta_j}}[h(X)=0] + \Pr_{X\sim\mathcal{P}_0}[h(X)=1] \,\middle|\, S\right]
\end{aligned}$$

Using $Z_{n\#\ell}$ to denote the collection of clusters u that appear exactly ℓ times in S, we have

$$\begin{aligned}
&\frac{1}{2}\mathop{\mathbb{E}}_{\pi\sim\Pi,\theta_{1:k}\sim\Psi^k}\mathop{\mathbb{E}}_{S\sim\widetilde{\mathcal{P}}_{\theta_{1:k},\pi},h\leftarrow\mathcal{A}(S)}\left[\Pr_{X\sim\mathcal{P}_{\theta,\pi}}[h(X)=0]+\Pr_{X\sim\mathcal{P}_0}[h(X)=1]\;\middle|\;S\right]\\
&=\sum_{0\le\ell\le n}\mathrm{errn}(\mathcal{A},S,\ell)\cdot\frac{1}{|Z_{n\#\ell}|}\cdot\sum_{u\in Z_{n\#\ell}}\mathbb{E}[\pi_u\mid S]\\
&=\sum_{0\le\ell\le n}\tau_\ell\cdot\mathrm{errn}(\mathcal{A},S,\ell).
\end{aligned}$$

In words, the term τ_ℓ reflects the change of (distribution of) π_u after conditioning on S, while the term errn reflects the change of θ. It turns out, by calculation, that after conditioning on S, the error of the algorithm can be succinctly described by the formula above.

To prove the theorem, it remains to argue that

$$\mathrm{opt}(\mathbf{P}^{\mathsf{mult}}\mid S)\le\tau_0\cdot\mathrm{errn}(\mathcal{A},S,0)+O(1/k). \tag{12}$$

We describe an algorithm $\mathcal{A}^*$ here: for every cluster u that has been seen at least once in S, the algorithm trains a binary-classification model to distinguish $\mathcal{P}_{\theta_u}$ from $\mathcal{P}_0$ with error at most $\frac{1}{k^2}$ (this is possible thanks to Assumption E.2). Then, upon receiving a query X, the algorithm tests whether X was likely from $\mathcal{P}_{\theta_u}$ for every seen cluster u. $\mathcal{A}^*$ outputs 1 if at least one of the tests outputs 1. and 0 otherwise. By simple union bound, for $X\sim\mathcal{P}_0$, the algorithm makes an error on X with probability at most $O(\frac{1}{k})$. For $X\sim\mathcal{P}_{\theta_u}$ where u-th cluster is present in S, the probability that $\mathcal{A}^*$ misclassifies X is at most $\frac{1}{k^2}$. For every $X\sim\mathcal{P}_{\theta_u}$ where $u\in Z_{n\#0}$, $\mathcal{A}^*$ might fail miserably on X. However, by Lemma E.1, the weight of all such clusters, after conditioning on S, is at most $\tau_0\cdot|Z_{n\#0}|$. This shows that the error of $\mathcal{A}^*$ is upper bounded by the right-hand side of (12), as claimed.

To establish the second item of the theorem, we simply take the average over S.

J.2. Proof of Theorem E.5

Let $S=((X_1,i_1),\dots,(X_n,i_n))$. We start by writing

$$\begin{aligned}
I(\mathcal{A}(S);S\mid\theta_{1:k},\pi)&=I(\mathcal{A}(S);S,(i_1,\dots,i_n)\mid\theta_{1:k},\pi)\\
&=I(\mathcal{A}(S);(i_1,\dots,i_n)\mid\theta_{1:k},\pi)+I(\mathcal{A}(S);S\mid\theta_{1:k},\pi,(i_1,\dots,i_n))\\
&\ge I(\mathcal{A}(S);S\mid\theta_{1:k},\pi,(i_1,\dots,i_n)).
\end{aligned}$$

We focus on the last line. By definition of conditional mutual information, we can first sample and condition on $\pi\sim\Pi_p^k$ and $i_1,\dots,i_n\sim\pi^n$, and consider the conditional mutual information $I(\mathcal{A}(S);S\mid\theta_{1:k})$ (where $i_1,\dots,i_n,\pi$ have been fixed). Note that the parameters $\theta_1,\dots,\theta_n$ and examples $X_1,\dots,X_n$ are independent of these choices and still have the same distribution. We denote by $S_{\mathcal{X}}:=(X_1,\dots,X_n)$ the unconditioned part of S.

Next, we want to show that the memorization of different clusters is essentially independent. Toward that goal, let S_j be the set of pairs (X,i) from S such that $i=j$ (i.e., S_j contains examples of S that are from the j-th cluster). We have

$$\begin{aligned}
I(\mathcal{A}(S);S\mid\theta_{1:k})&=H(S\mid\theta_{1:k})-H(S\mid\theta_{1:k},\mathcal{A}(S))\\
&\ge\sum_{j\in[k]}H(S_j\mid\theta_j)-H(S_j\mid\theta_{1:k},\mathcal{A}(S))&&\text{(sub-additivity and additivity of entropy)}\\
&\ge\sum_{j\in[k]}H(S_j\mid\theta_j)-H(S_j\mid\theta_j,\mathcal{A}(S))&&\text{(monotonicity of conditional entropy)}\\
&=\sum_{j\in[k]}I(\mathcal{A}(S);S_j\mid\theta_j).
\end{aligned}$$

We now derive the claimed lower bound. As before, let $I_{n\#\ell}(S)$ be the collection of all j's that appear exactly ℓ times among $i_1,\dots,i_n$.

Having conditioned on $i_1, \ldots, i_n$, let $j \in I_{n\#\ell}(S)$. We consider how well the algorithm distinguishes $\mathcal{P}_{\theta_j}$ from $\mathcal{P}_0$ (when we take the average over θ_j, S_j and $\mathcal{A}(S)$). Thinking of this as a binary classification problem between $\mathcal{P}_{\theta_j}$ and $\mathcal{P}_0$ where ℓ samples from $\mathcal{P}_{\theta_j}$ are available, we can evaluate the error of the algorithm by

$$e_j := \frac{1}{2} \mathop{\mathbb{E}}_{\theta_{1:k}\sim\Psi^k, S_{\mathcal{X}}, h\leftarrow\mathcal{A}(S)} \left[\Pr_{X\sim\mathcal{P}_{\theta_j}}[h(X)=0] + \Pr_{X\sim\mathcal{P}_0}[h(X)=1] \right].$$

Using Assumption E.4, we obtain that

$$I(\mathcal{A}(S); S_j \mid \theta_j) \geq c_{\mathbf{P}}(1-2e_j) \cdot \mathrm{mem}_\ell(\mathbf{P}).$$

Note that

$$\sum_{j\in I_{n\#\ell}(S)} e_j = \mathop{\mathbb{E}}_{\theta_{1:k}\sim\Psi^k, S_{\mathcal{X}}} \left[\mathrm{errn}(\mathcal{A}, \theta_{1:k}, S, \ell)\right].$$

Adding up the contribution from different $j \in I_{n\#\ell}(S)$, we get

$$\sum_{j\in I_{n\#\ell}(S)} I(\mathcal{A}(S); S_j \mid \theta_j) \geq c_{\mathbf{P}} \cdot \mathop{\mathbb{E}}_{\theta_{1:k}\sim\Psi^k, S_{\mathcal{X}}} \left[(|I_{n\#\ell}(S)| - 2\cdot\mathrm{errn}(\mathcal{A}, \theta_{1:k}, S, \ell)) \cdot \mathrm{mem}_\ell(\mathbf{P})\right].$$

Finally, summing over all ℓ and taking the average over $\pi, i_1, \ldots, i_n$ concludes the proof.