Sample Compression Unleashed:
New Generalization Bounds for Real Valued Losses

Mathieu Bazinet
Université Laval
mathieu.bazinet.2@ulaval.ca

Valentina Zantedeschi
ServiceNow Research, Université Laval
vzantedeschi@gmail.com

Pascal Germain
Université Laval
pascal.germain@ift.ulaval.ca

Abstract

The sample compression theory provides generalization guarantees for predictors that can be fully defined using a subset of the training dataset and a (short) message string, generally defined as a binary sequence. Previous works provided generalization bounds for the zero-one loss, which is restrictive notably when applied to deep learning approaches. In this paper, we present a general framework for deriving new sample compression bounds that hold for real-valued unbounded losses. Using the Pick-To-Learn (P2L) meta-algorithm, which transforms the training method of any machine-learning predictor to yield sample-compressed predictors, we empirically demonstrate the tightness of the bounds and their versatility by evaluating them on random forests and multiple types of neural networks.

1 Introduction

Sample compression theory, introduced by [33], is based on the fundamental idea that "compressing implies learning" [12]. If it is possible to provably show that a learned model can be completely defined by a subset of the training dataset, then sample compression theory gives us generalization guarantees. The most well-known learning algorithms that comply with the sample compression framework are the support vector machine [5] and the perceptron [47, 41]; the relevant training subset being formed by the support vectors in the former case, and the points causing an update of the predictor in the latter case. More recently, [52] and [43] have introduced the first sample compression results for neural networks.

The sample compression theory is rich and multiple different approaches exist. For example, [2, 3, 12, 15, 21, 22, 23, 24, 42, 48] propose theoretical results relating the VC dimension [59] and the compression analysis. By relating the probability of *change of compression* to the true risk, [9, 43] express very tight guarantees for the consistent case, i.e., when the error on the training set is zero. Finally, [31, 36, 37, 38, 51] give computable risk certificates valid even in the non-consistent case.

In this paper, we build on the setting of [31], based on the binomial test-set bound of [29], which by definition is the tightest bound for the zero-one loss under the sole *i.i.d.* assumption. However, the use of the zero-one loss restricts its application to supervised classification problems. By leveraging proof techniques from the PAC-Bayesian literature, we extend the framework to real-valued losses and open the way to obtaining bounds directly for the cross-entropy loss [45] and unbounded losses [20, 10, 46], for example under the sub-Gaussian assumption. Finally, we train deep neural networks and random forests with the Pick-To-Learn meta-algorithm [43], an algorithm that modifies the training loop of a model to yield a sample-compressed predictor, and assess the tightness of our

Mathematics of Modern Machine Learning Workshop at NeurIPS 2024.

bounds in different settings. In the consistent case, our bounds are arbitrarily tight upper bounds on previous results restricted to the zero-one loss.

Of note, a major asset of our sample-compress bounds is that they do not depend on the number of learnable parameters. Two models of different sizes can achieve the same guarantees as long as they achieve the same empirical loss using the same amount of data. This lets us train large models such as DistilBERT and still achieve tight generalization bounds.

2 Background and Notation

We are interested in the supervised learning framework. Let $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ be a dataset of n datapoints, with each point sampled *i.i.d.* (independently and identically distributed) from an unknown distribution $\mathcal{D}$ over $\mathbb{R}^d \times \mathcal{Y}$. The targets are defined by the task at hand, with $\mathcal{Y} \in \{-1, +1\}$ for binary classification tasks and $\mathcal{Y} \subseteq \mathbb{R}$ for regression tasks. For the rest of this section, we focus on binary classification problems, but in Section 3, we study both classification and regression settings.

Let $\mathcal{H}$ be a family of predictors $h : \mathcal{X} \to \mathcal{Y}$. Let $A : \bigcup_{k=1}^{\infty} (\mathcal{X} \times \mathcal{Y})^k \to \mathcal{H}$ be an algorithm that takes a dataset S and returns a predictor $A(S)$. We consider the zero-one loss function $\ell^{0\text{-}1}(h, \boldsymbol{x}, y) = \mathbb{I}[h(\boldsymbol{x}) \neq y]$, with $\mathbb{I}[a] = 1$ if the predicate a is true and 0 otherwise. Then, the true risk of the hypothesis h is defined as

$$R_{\mathcal{D}}(h) \;=\; \underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathbb{P}} (h(\boldsymbol{x}) \neq y) \;=\; \underset{(\boldsymbol{x},y)\sim\mathcal{D}}{\mathbb{E}} \mathbb{I}[h(\boldsymbol{x}) \neq y]$$

and, for a realization $S \sim \mathcal{D}^n$, its empirical risk is defined as $\widehat{R}_S(h) = \frac{1}{n} \sum_{i=1}^n \mathbb{I}[h(\boldsymbol{x}_i) \neq y_i]$.

Since the distribution $\mathcal{D}$ is unknown, the true risk of a hypothesis cannot be computed. However, it can be upper bounded with high probability, using generalization bounds derived from statistical learning theories such as the sample compression theory.

2.1 Sample compression theory

Let $h = A(S)$ be the output of algorithm A applied to a dataset S. In order to obtain guarantees on the generalization performance of h using the sample compression theory, we need to be able to uniquely define h as a function (the reconstruction function) of a subset of S (the compression set) and a complementary sequence of information (the message).

The compression set $S_{\mathbf{i}}$ is defined using a vector of indices $\mathbf{i} = \big(i_1, i_2, \ldots, i_{|\mathbf{i}|}\big)$, where the indices are ordered such that $1 \leq i_1 < i_2 < \ldots < i_{|\mathbf{i}|} \leq n$. The vector $\mathbf{i}$ belongs in the set of all possible vectors composed of the natural numbers 1 through n, denoted $\mathcal{P}(n)$. Using this notation, $\mathbf{i}$ indicates the datapoints of S that are present in $S_{\mathbf{i}}$, as such

$$S_{\mathbf{i}} = \big\{(\boldsymbol{x}_{i_1}, y_{i_1}), \ldots, (\boldsymbol{x}_{i_{|\mathbf{i}|}}, y_{i_{|\mathbf{i}|}})\big\} \subseteq S\,.$$

Moreover, we define the complement vector $\mathbf{i}^c \in \mathcal{P}(n)$ such that $S_{\mathbf{i}^c} = S \setminus S_{\mathbf{i}}$ and $|\mathbf{i}^c| = n - |\mathbf{i}|$.

The message σ is chosen in a set $M(\mathbf{i})$, which contains all relevant messages associated to the compression set $\mathbf{i}$. The message is a complementary source of information and is generally defined as a binary sequence.

A predictor h is called a sample-compressed predictor if there exists a vector $\mathbf{i} \in \mathcal{P}(n)$ and (optionally) a message $\sigma \in M(\mathbf{i})$ such that $h = \mathcal{R}(S_{\mathbf{i}}, \sigma)$, where $\mathcal{R} : \bigcup_{m \leq n} (\mathcal{X} \times \mathcal{Y})^m \times \bigcup_{\mathbf{i} \in \mathcal{P}(n)} M(\mathbf{i}) \to \overline{\mathcal{H}}$ is a data-independent deterministic reconstruction function and $\overline{\mathcal{H}} \subseteq \mathcal{H}$ is a discrete set of sample-compressed predictors.

We define a distribution $P_{\overline{\mathcal{H}}}$ over $\overline{\mathcal{H}}$, such that $\sum_{h \in \overline{\mathcal{H}}} P_{\overline{\mathcal{H}}}(h) \leq 1$. As all sample-compressed predictors are uniquely defined using the indices vector and the message, we choose the distribution $P_{\overline{\mathcal{H}}}$ to be a product of two distributions $P_{\overline{\mathcal{H}}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)) = P_{\mathcal{P}(n)}(\mathbf{i}) P_{M(\mathbf{i})}(\sigma)$, with $P_{\mathcal{P}(n)}$ a distribution on $\mathcal{P}(n)$ and $P_{M(\mathbf{i})}$ a distribution on $M(\mathbf{i})$. Following previous works [e.g. 38], we require the distribution $P_{\mathcal{H}}$ to be data-independent, in order to avoid further assumptions. Without any information on the data, we generally set $P_{M(\mathbf{i})}$ to a uniform distribution. As for the distribution $P_{\mathcal{P}(n)}$, it is usually set to penalize larger compression sets [31, 37, 38]. For any size of compression set $|\mathbf{i}|$, there are

$\binom{n}{|\mathbf{i}|}$ different possible compression sets. We set the distribution $P_{\mathcal{P}(n)}(\mathbf{i})$ to be $\binom{n}{|\mathbf{i}|}^{-1}\zeta(|\mathbf{i}|)$, with $\zeta(m) = \frac{6}{\pi^2}(m+1)^{-2}$. This choice is discussed by [38].

We now present the sample compression bound of [31]. This result is derived using the binomial test-set bound of [29], which by definition is the tightest bound for the zero-one loss under the sole i.i.d. assumption.

Theorem 1 ([31], Theorem 1)**.** *For any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, for any family of set of messages $\{M(\mathbf{i}) \mid \mathbf{i} \in \mathcal{P}(n)\}$, for any deterministic reconstruction function $\mathcal{R}$ that outputs sample-compressed predictors $h \in \overline{\mathcal{H}}$ and for any $\delta \in (0,1]$, with probability at least $1-\delta$ over the draw of $S \sim \mathcal{D}^n$, we have*

$$\forall\, \mathbf{i} \in \mathcal{P}(n), \sigma \in M(\mathbf{i}) : R_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)) \le \overline{\text{Bin}}\left(\kappa, n, \binom{n}{|\mathbf{i}|}^{-1} \zeta(|\mathbf{i}|) P_{M(\mathbf{i})}(\sigma)\delta\right)$$

with $\kappa = n\widehat{R}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma))$ and

$$\overline{\text{Bin}}(k, m, \delta) = \sup_{r \in [0,1]} \left\{ \sum_{i=0}^{k} \binom{m}{i} r^i (1-r)^{m-i} \ge \delta \right\}.$$

This theorem can be applied to any family of sample-compressed predictors, such as the support vector machine, the perceptron, and the set covering machine [37]. To apply this theorem to neural networks, one must design a reconstruction function outputting neural networks. To this end, [52] propose to reparameterize a 2-layer LeakyReLU network in order to obtain "support vectors", which become the compression set of the reconstructed network. The following section presents a more general approach proposed by [43].

2.2 Pick-To-Learn

Conceptualized by [43], Pick-To-Learn (P2L, Algorithm 1) is a model-agnostic meta-algorithm that trains any model in such a way that it becomes a sample-compressed predictor. This algorithm is specifically designed for the generalization bound of [9], which holds only for sample compressed predictors in the *consistent case*, i.e., when $\widehat{R}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma))$=0.

Algorithm 1: Pick-To-Learn (P2L)

```
Initialize : $S_{\mathbf{i}} = \emptyset$
Initialize : $h_{\mathbf{i}} = h_0$
Initialize : $(\overline{\boldsymbol{x}}, \overline{y}) = \arg\max_{(\boldsymbol{x},y) \in S} \ell^{\text{x-e}}(h_0, \boldsymbol{x}, y)$
while $-\ln(0.5) \le \ell^{\text{x-e}}(h_{\mathbf{i}}, \overline{\boldsymbol{x}}, \overline{y})$ do
    $S_{\mathbf{i}} \leftarrow S_{\mathbf{i}} \cup \{(\overline{\boldsymbol{x}}, \overline{y})\}$
    $h_{\mathbf{i}} \leftarrow A(S_{\mathbf{i}})$
    $(\overline{\boldsymbol{x}}, \overline{y}) \leftarrow \arg\max_{(\boldsymbol{x},y) \in S_{\mathbf{i}^c}} \ell^{\text{x-e}}(h_{\mathbf{i}}, \boldsymbol{x}, y)$
end
return $h_{\mathbf{i}}$
```

To obtain sample-compressed predictors, P2L iteratively builds the compression set and trains the model on it. Starting with an initial predictor h_0, P2L tests the model on the whole dataset, picks the datapoint over which the model got the largest loss value, and adds it to the compression set. Then, using a learning algorithm A, P2L trains the model on the newly created compression set. The previous steps are repeated until the model achieves zero errors on the training set $S_{\mathbf{i}^c}$ (excluding the compression set datapoints), which is equivalent to stopping when the cross-entropy loss ($\ell^{\text{x-e}}$) becomes smaller than $-\ln(0.5)$.

Leveraging from the theoretical results of [9], [43] derived a theorem specifically for the P2L algorithm.

Theorem 2 ([43], Theorem 4.2)**.** *Let $h_{\mathbf{i}} = \mathcal{R}(S_{\mathbf{i}}, \emptyset)$ be the output of P2L. For any $\delta \in (0,1)$, with probability at least $1-\delta$ over the draw of $S \sim \mathcal{D}^n$, we have*

$$R_{\mathcal{D}}(h_{\mathbf{i}}) \le \overline{\varepsilon}(|\mathbf{i}|, \delta),$$

with

$$\Psi_{k,\delta}(\varepsilon) = \frac{\delta}{2N} \sum_{m=k}^{n-1} \frac{\binom{m}{k}}{\binom{n}{k}} (1-\varepsilon)^{-(n-m)} + \frac{\delta}{6N} \sum_{m=n+1}^{4N} \frac{\binom{m}{k}}{\binom{n}{k}} (1-\varepsilon)^{m-n}$$

and where, for $k = 0, 1, \ldots, n-1$, $\overline{\varepsilon}(k, \delta)$ is the unique solution to the equation $\Psi_{k,\delta}(\varepsilon) = 1$ in the interval $[\frac{k}{n}, 1]$, while $\overline{\varepsilon}(n, \delta) = 1$.

Note that the value of previous bound is completely determined by the size of the compression set. The faster P2L obtains zero errors, the better the bound will be.

3 A General Sample-Compress Bound

Let $\mathcal{H}$ be a family of predictors $h : \mathcal{X} \to \overline{\mathcal{Y}}$, where $\overline{\mathcal{Y}} \supseteq \mathcal{Y}$ is a convex hull of $\mathcal{Y}$. For example, $[-1, 1]$ is the convex hull of $\{-1, +1\}$. We consider a loss function $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. Then, the true risk of the hypothesis h is defined as $\mathcal{L}_{\mathcal{D}}(h) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\, \ell(h, \boldsymbol{x}, y)$ and, for a realization $S \sim \mathcal{D}^n$, its empirical risk is defined as $\widehat{\mathcal{L}}_S(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(h, \boldsymbol{x}_i, y_i)$. This setting is a generalization of the setting of Section 2. As Theorem 1 only holds for the zero-one loss, we need new results to extend the sample-compression theory to this setting.

To extend the work of [31] to real-valued losses, we introduce a *comparator function* $\Delta : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and provide a new result inspired by the general PAC-Bayes bound [17]. Theorem 3 presents a new general sample-compress bound that holds for any real-valued losses, extending the applicability of the sample-compression theory. The theorem is followed by a proof sketch highlighting the main steps, and the full proof is given in Appendix C.

Theorem 3. *For any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, for any family of set of messages $\{M(\mathbf{i}) \mid \mathbf{i} \in \mathcal{P}(n)\}$, for any deterministic reconstruction function $\mathcal{R}$ that outputs sample-compressed predictors $h \in \mathcal{H}$, for any loss $\ell : \mathcal{H} \times \mathcal{X} \times Y \to \mathbb{R}$, for any comparator function $\Delta : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and for any $\delta \in (0, 1]$, with probability at least $1 - \delta$ over the draw of $S \sim \mathcal{D}^n$, we have*

$$\forall \mathbf{i} \in \mathcal{P}(n), \sigma \in M(\mathbf{i}) :$$

$$\Delta\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma))\Big) \le \frac{1}{|\,\mathbf{i}^c\,|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{\mathcal{E}_{\Delta}(\mathbf{i}, \sigma)}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right]$$

with

$$\mathcal{E}_{\Delta}(\mathbf{i}, \sigma) = \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}}\ \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{|\mathbf{i}^c|}} e^{|\mathbf{i}^c|\Delta\big(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\big)}.$$

Proof Sketch. For all $\mathbf{i} \in \mathcal{P}(n)$, $\sigma \in M(\mathbf{i})$, $\epsilon > 0$, using Chernoff's bound with $t > 0$, we have

$$\begin{aligned}
&\mathop{\mathbb{P}}_{S\sim\mathcal{D}^n}\Big(\Delta\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma))\Big) > \epsilon\Big) && (1)\\
&\le e^{-t\epsilon} \mathop{\mathbb{E}}_{S\sim\mathcal{D}^n} e^{t\Delta\big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}},\sigma))\big)}\\
&= e^{-t\epsilon} \mathop{\mathbb{E}}_{S_{\mathbf{i}}\sim\mathcal{D}^n}\ \mathop{\mathbb{E}}_{S_{\mathbf{i}^c}\sim\mathcal{D}^n} e^{t\Delta\big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}},\sigma))\big)}
\end{aligned}$$

where the last equality requires *i.i.d.* datapoints. For any $\delta_{\mathbf{i}}^{\sigma} \in (0, 1]$, we define

$$\delta_{\mathbf{i}}^{\sigma} = e^{-t\epsilon} \mathop{\mathbb{E}}_{S_{\mathbf{i}}\sim\mathcal{D}^n}\ \mathop{\mathbb{E}}_{S_{\mathbf{i}^c}\sim\mathcal{D}^n} e^{t\Delta\big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}},\sigma))\big)} \qquad (2)$$

and solve for ϵ, using $t = n - |\mathbf{i}|$. The obtained solution is used to replace the ϵ in Eq. (1), which gives a bound valid with probability at most $\delta_{\mathbf{i}}^{\sigma}$ for every single predictor $\mathcal{R}(S_{\mathbf{i}}, \sigma)$. By setting $\delta_{\mathbf{i}}^{\sigma} = P_{\mathcal{P}(n)}(\mathbf{i})P_{M(\mathbf{i})}(\sigma)\delta$ and applying a union bound over all $\mathbf{i} \in \mathcal{P}(n)$, $\sigma \in M(\mathbf{i})$, the final result holds uniformly with probability δ for all predictors outputted by $\mathcal{R}$. □

Theorem 3 holds for any comparator function Δ such that $\mathcal{E}_{\Delta}$ is finite for any pair $(\mathbf{i}, \sigma)$. Although bounding $\mathcal{E}_{\Delta}$ can be challenging, it was extensively studied for convex functions in PAC-Bayesian theory [e.g., 40, 39, 10, 26]. We leverage this theory and present novel corollaries for the three most well-known comparators.

First of all, we present a bound using the comparator $\Delta_C(q, p) = -\ln\big(1 - p(1 - e^{-C})\big) - Cq$. The family of bounds $\{\Delta_C : C > 0\}$ is commonly referred to as "Catoni bounds" [11] in the PAC-Bayes literature.

Corollary 4. *In the setting of Theorem 3, for any $C > 0$, for a loss function $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to [0, 1]$, with probability at least $1 - \delta$ over the draw of $S \sim \mathcal{D}^n$, we have*

$$\forall\, \mathbf{i} \in \mathcal{P}(n), \sigma \in M(\mathbf{i}) :\ \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)) \le \frac{1 - \exp(-\epsilon_C(\mathbf{i}, \sigma, \delta))}{1 - e^{-C}}$$

with

$$\epsilon_C(\mathbf{i},\sigma,\delta) = C\,\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) + \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right].$$

For $0 \le q, p \le 1$, there exists $C^* = \arg\sup_{C>0}\Delta_C(q,p)$ such that Δ_{C^*} gives the tightest PAC-Bayesian bounds [16]. This result also holds true for Theorem 3, when restricted to proper, convex and lower semicontinuous comparator functions $\Delta : [0,1]\times[0,1]\to\mathbb{R}$. Unfortunately, the hyper-parameterized bounds hold for only one value of C, chosen prior to seeing S. With a union bound argument, we can consider multiple parameters C simultaneously, but there is no guarantee that C^* is in this set. To circumvent this problem, we can use the binary Kullback-Leibler divergence comparator function $\mathrm{kl}(q,p) = q\ln\frac{q}{p} + (1-q)\ln\frac{1-q}{1-p}$, which is equivalent to Δ_{C^*}, as per the following proposition.

Proposition 5 ([17], Proposition 2.1)**.** *For any* $0 \le q \le p < 1$*, we have* $\sup_{C\ge 0}\Delta_C(q,p) = \mathrm{kl}(q,p)$*.*

In practice, even with the term $1 = \mathcal{E}_{\Delta_C}(n) \le \mathcal{E}_{\mathrm{kl}}(n) = 2\sqrt{n}$, the kl bound will usually yield tighter bounds than Corollary 4, as the optimal value of C is unlikely to be selected before computing the bound. Moreover, the kl is known to be optimal for $[0,1]$-valued losses, as per the results of [26].

Corollary 6. *In the setting of Theorem 3, for a loss function* $\ell : \mathcal{H}\times\mathcal{X}\times\mathcal{Y}\to[0,1]$*, with probability at least* $1-\delta$ *over the draw of* $S\sim\mathcal{D}^n$*, we have*

$$\forall\mathbf{i}\in\mathcal{P}(n), \sigma\in M(\mathbf{i}) : \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) \le \mathrm{kl}^{-1}\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)), \epsilon_{\mathrm{kl}}(\mathbf{i},\sigma,\delta)\Big)$$

with $\mathrm{kl}^{-1}(q,\epsilon) = \arg\sup_{0\le p\le 1}\{\mathrm{kl}(q,p)\le\epsilon\}$ *and*

$$\epsilon_{\mathrm{kl}}(\mathbf{i},\sigma,\delta) = \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{2\sqrt{n-|\mathbf{i}|}}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right].$$

Both Corollary 4 and Corollary 6 hold for losses bounded in $[0,1]$. Using the linear function $\Delta_\lambda(q,p) = \lambda(p-q)$, we can extend this sample compression framework to unbounded losses provided that $\mathcal{E}_{\Delta_\lambda}$ is bounded. As an example, we present a result for sub-Gaussian losses [27].

Corollary 7. *In the setting of Theorem 3, for any* $\lambda > 0$*, with a* ς^2*-sub-Gaussian loss function* $\ell : \mathcal{H}\times\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$*, with probability at least* $1-\delta$ *over the draw of* $S\sim\mathcal{D}^n$*, we have*

$$\forall\mathbf{i}\in\mathcal{P}(n), \sigma\in M(\mathbf{i}) :$$
$$\mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) \le \widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) + \frac{\lambda\varsigma^2}{2} + \frac{1}{\lambda(n-|\mathbf{i}|)}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right].$$

This result could be extended to the hypothesis-dependent range condition of [20], any unbounded losses under model-dependent assumptions [10] or more general tail behaviors [46]. Note that this result encompasses bounded losses with a range of $[a,b]$, as they are sub-Gaussian with $\varsigma = \frac{b-a}{2}$.

3.1 Behavior in the consistent case

In this section, we present a new theoretical result that justifies the tightness of the bounds observed in Section 4. In this setting, predictors stop training when the empirical error reaches zero. Indeed, we show that when $\widehat{R}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) = 0$, our new bounds from Corollaries 4 and 6 are extremely tight upper bounds to the binomial bound of [31].

Theorem 8. *In the consistent case, i.e. when the training error is zero, Corollary 4 is arbitrarily close to the binomial tail inversion of Theorem 1. Moreover, Corollary 6 is a tight upper bound up to* *a constant $K(m,\delta)$ that decreases for m large enough and tends to 0 when m tends to ∞. Indeed,*

$$\begin{aligned}
\overline{\mathrm{Bin}}(0,m,\delta) =&1-\exp\left(\frac{-1}{m}\ln\frac{1}{\delta}\right) && (3)\\
=&\lim_{C\to\infty}\frac{1}{1-e^{-C}}\left[1-\exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right)\right] && (4)\\
=&\inf_{C>0}\frac{1}{1-e^{-C}}\left[1-\exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right)\right] && (5)\\
=&\mathrm{kl}^{-1}\left(0,\frac{1}{m}\ln\frac{1}{\delta}\right) && (6)\\
\leq&\mathrm{kl}^{-1}\left(0,\frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right). && (7)\\
=&\mathrm{kl}^{-1}\left(0,\frac{1}{m}\ln\frac{1}{\delta}\right)+K(m,\delta). && (8)
\end{aligned}$$

We prove the previous sequence of results in Appendix C.3. In the previous result, we relate the analytical form of $\overline{\mathrm{Bin}}(0,m,\delta)$ in Eq. (3) to the bound of Corollary 4. This relation is known to hold as an inequality for any $k\geq 0$, but we show that it is an equality for $k=0$. In Eq. (4), we show that the hyperparameterized distance is arbitrarily tight to the binomial tail inversion, as it approaches $\overline{\mathrm{Bin}}(0,m,\delta)$ when $C\to\infty$. We show in Eq. (5) that the minimal value of the bound is obtained when C tends to ∞. In Eq. (6), we relate the hyperparameterized distance to the Kullback-Leibler divergence, which we then upper bound in Eq. (7), achieving the result of Corollary 6. Finally, in Eq. (8), we demonstrate that Corollary 6 is a tight upper bound to the binomial tail inversion, up to a constant $K(m,\delta)$ that decreases when m is large and tends to 0 when $n\to\infty$. For example, with $\delta=0.01$, $K(m,\delta)$ is bounded by $K(7,0.01)\approx 0.11$, decreases for $m\geq 8$ and reaches $K(m,\delta)\leq 0.01$ at $m=357$.

4 Experiments

In this section, we show the versatility of our results by training different models using the P2L algorithm.[1] In Section 4.1, we train neural networks on binary classification problems and compare our new results to the pre-existing sample compression results. We empirically validate that our bounds are almost as tight as the binomial bound, all the while not suffering from the numerical optimization problem of Theorem 1 and being defined in the inconsistent case, where the P2L bound of Theorem 2 is undefined. In Section 4.2, we train CNNs on the MNIST dataset and present generalization bounds on the (bounded) cross-entropy loss. As no previous sample-compression bound is defined for real-valued losses, we compare our result to a PAC-Bayesian theorem. In Section 4.3, we use P2L to train decision forests on regression datasets and give generalization bounds on the root mean squared error (RMSE), an unbounded loss function, under the assumption that it is sub-Gaussian. Finally, in Section 4.4, we fine-tune DistilBERT, a 66M parameters language model, on a review polarity classification problem. We obtain tight bounds simultaneously on the zero-one loss and the cross-entropy loss, demonstrating that our new theorem is independent of the number of parameters of the model.

Each experiment is run five times with different seeds (with the exception of amazon polarity). In all tables, we present the mean and standard deviation of the metrics over five seeds. The datasets are separated into three parts: the training, validation and test set. The validation set is built using 10% of the training set. If the dataset doesn't have a built-in test set, we build a test set first, using 10% of the dataset, and then build the validation set. When computing the bounds, we use $\delta=0.01$. All baselines are trained on the whole dataset using stochastic gradient descent for 200 epochs or until the model achieves zero errors on the training set. The random forests [6] are trained on the whole dataset, with no modifications to the training algorithm. All the hyperparameters for all the experiments can be found in Appendix A.

Table 1: Results for the CNNs trained using P2L on the binary MNIST problems. The results displayed obtained the tightest P2L bound. All metrics presented are in percent (%), with the exception of $|\mathbf{i}|$.

Dataset	Validation error	Test error	kl bound	Binomial bound	P2L bound	$\lvert\mathbf{i}\rvert$	Baseline test error
MNIST08	0.33±0.17	0.25±0.10	5.05±0.16	5.00±0.16	1.04±0.04	92.0±3.6	0.22±0.05
MNIST17	0.20±0.08	0.38±0.16	4.33±0.21	4.29±0.21	0.86±0.05	84.0±5.2	0.17±0.03
MNIST23	0.39±0.12	0.27±0.10	8.20±0.34	8.15±0.34	1.86±0.09	175.6±9.5	0.16±0.05
MNIST49	0.82±0.11	0.77±0.17	10.52±0.37	10.47±0.37	2.53±0.11	237.0±11.0	0.44±0.07
MNIST56	0.46±0.12	0.47±0.15	6.29±0.22	6.24±0.22	1.35±0.06	117.0±5.2	0.30±0.08

Table 2: Results for the CNNs trained using P2L on the binary MNIST problems and stopped at the iteration with the minimum kl bound. The results displayed obtained the tightest kl bound. Metrics are in percents (%), except $|\mathbf{i}|$.

Dataset	Validation error	Test error	kl bound	Binomial bound	Train error	$\lvert\mathbf{i}\rvert$	Baseline test error
MNIST08	0.49±0.39	0.49±0.26	4.71±0.25	5.33±0.62	0.24±0.23	66.0±15.0	0.22±0.05
MNIST17	0.45±0.18	0.48±0.11	3.70±0.21	4.37±0.11	0.23±0.08	50.0±8.9	0.17±0.03
MNIST23	0.74±0.28	0.84±0.21	6.56±0.38	8.09±0.64	0.64±0.32	84.0±21.5	0.16±0.05
MNIST49	1.16±0.31	1.13±0.24	8.60±0.46	9.61±0.68	0.51±0.28	134.0±24.2	0.44±0.07
MNIST56	0.94±0.09	0.70±0.20	5.42±0.31	6.49±0.81	0.43±0.23	66.0±10.2	0.30±0.08

4.1 Binary MNIST

We create binary classification datasets by choosing two digits from the MNIST dataset [32], e.g., choosing all the datapoints labeled 0 and 8 to build the dataset MNIST08. We create five datasets: MNIST08, MNIST17, MNIST23, MNIST49 and MNIST56. Starting from randomly initialized neural networks, we train a MLP and a CNN using P2L on each dataset. More details are given in Appendix A.1.1.

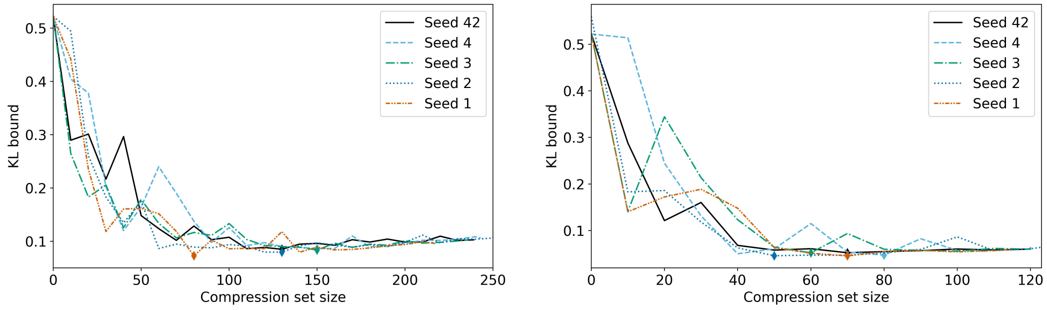
For all experiments in this section, we compute our proposed kl bound (Corollary 6), the binomial approximation bound of [31] (Corollary 9, in appendix) and the P2L bound of [43] (Theorem 2). We do not compute the binomial tail inversion of Theorem 1 as its optimization is very unstable. However, the binomial approximation is equivalent to Theorem 1 when $k = 0$, which corresponds to the consistent case reached by the P2L algorithm.

We present our results for the CNN in Table 1. All the results for the MLP can be found in Appendix A.1.1. The error on the training set is zero for all predictors returned by P2L. The results presented achieved the tightest P2L bound for each dataset. For reference, the reported "baseline test error" corresponds to the results of the best baseline model based on validation error. For both architectures, using P2L only incurs a slight increase of the test error compared to the baseline, whilst the model is trained on a very small percentage of the dataset, ranging from 0.7% to 3.4%. Finally, even though the P2L bound is much tighter than the proposed kl bound, our result is much more general, as it holds for any real-valued loss functions and in the non-consistent case. Moreover, our bounds hold uniformly over all iterations of the models trained using P2L. After training, one can use any checkpoint of the model and still obtain a valid bound, which gives control over a trade-off between the training error, the generalization bound and the validation error. In Fig. 1, we present the behavior of the bound throughout the P2L iterations. The minimal kl bound happens at about half the final number of iterations, leading to a smaller compression set and a tighter bound, as also reported in Table 2. In comparison to the previous results, the test error is about twice as high as the test error of the fully trained model (Table 1). However, the models were trained on very small portions of the dataset, with the model on MNIST17 being trained on 0.42% of the dataset and still achieving a test error of 0.48%. Finally, we observe that, in this setting, our new kl bound is much tighter than the binomial approximation of [31].

4.2 MNIST

We now train convolutional neural networks composed of two convolutional layers and two fully connected layers. We pre-train the model using stochastic gradient descent on a subset of the dataset

[1] Our code is available at `https://github.com/GRAAL-Research/pick-to-learn`.



(a) MNIST49

(b) MNIST56

Figure 1: Illustration of the behavior of the kl bound throughout P2L iterations for the five different seeds of the hyperparameter combination that achieved the minimal P2L bound on MNIST49 and MNIST56. We mark the minimal kl bound for each seed with a diamond (♦). The results for the other datasets can be found in Fig. 2.

Table 3: Cross-entropy loss achieved by the CNNs on MNIST. The results displayed obtained the smallest kl bound.

Model	Train loss	Test loss	kl bound	$\lvert\mathbf{i}\rvert$	Baseline test loss
P2L	0.0008±0.0006	0.0480±0.0073	0.7142±0.1773	275.20±82.46	0.0499±0.0108
PBB	0.0092±0.0005	0.0045±0.0004	0.0112±0.0005	-	

and then use P2L to fine-tune the model on the train set. The size of the pre-training subset is an hyperparameter. We use the same training setting as in Section 4.1 and use the extension of P2L that adds multiple datapoints to the compression set at a time, with batch size $R = 32$, as defined by Algorithm 2 of [43]. For comparison, we also train probabilistic neural networks (PNN) using the PAC-Bayes with Backprop (PBB) approach of [45]. They train the model by minimizing the PAC-Bayesian kl bound of Theorem 10. See Appendix A.1.2 for details.

For both our new sample-compression bounds and the PAC-Bayesian bound of [45], we compute the bounds on the zero-one loss and on a bounded version of the cross-entropy loss (see Appendix A.1.2). The probabilities outputted by the neural networks are restricted to be greater than 10^{-5}, effectively bounding the cross-entropy by $-\ln(10^{-5}) \approx 11.51$.

In Table 3, we report the bound values for the bounded cross-entropy loss (see Appendix for classification error). We observe that the PBB algorithm gives a tighter generalization bound than the one of P2L. This gap can be explained by the fact that PBB jointly optimizes the train error and the KL divergence, whilst we have almost no control on the minimization of the bound. Indeed, the heuristic of the P2L algorithm, which is to choose the datapoints over which the model incurs the greatest losses, doesn't give control on the trade-off between the decrease of the error and the increase of the complexity term. Moreover, for a large dataset, the binomial coefficient increases rapidly when the compression set size increases. However, using our bounds with the P2L algorithm has multiple advantages over the PBB algorithm. First of all, PBB needs to train twice as many parameters, as it fits both the mean and standard deviation of the distributions over the parameters. Secondly, computing the PAC-Bayesian bound necessitates a step of Monte Carlo sampling to determine the average error of the model. For 5000 steps of Monte Carlo sampling, the error over the dataset will be computed 5000 times, instead of only once with P2L. Finally, our bound doesn't take into account the number of parameters of the model, whilst the KL divergence in Theorem 10 is a sum of the KL divergence of the distribution of each parameter of the model.

4.3 Regression forests

In order to show the wide applicability of our bounds, we train decision forests on regression problems: Powerplant [58], Infrared [61], Airfoil [8], Parkinson [57] and Concrete [63]. These datasets range from a training set size of 827 to 7751 and range from a number of features of 4 to 33. To the best of

Table 4: Results for the decision forests trained using P2L. We report the RMSE achieved by the models and the generalization bounds on the RMSE.

Dataset	Train loss	Validation loss	Test loss	kl bound	Linear bound	$\lvert\mathbf{i}\rvert$	Baseline test loss	$\ell^{\max}$
Powerplant	5.23±2.23	5.23±2.18	5.37±2.33	11.08±5.04	12.79±5.91	29.20±17.81	3.59±0.13	90.6
Infrared	0.27±0.03	0.29±0.04	0.30±0.03	1.08±0.08	1.16±0.08	19.20±5.49	0.23±0.01	4.26
Airfoil	3.57±0.34	3.91±0.21	3.88±0.39	14.78±1.39	14.84±1.26	46.80±15.93	2.10±0.15	45.13
Parkinson	7.59±0.44	7.75±0.50	7.73±0.36	12.13±0.37	11.98±0.41	22.60±10.59	2.23±0.16	41.37
Concrete	8.59±1.10	8.74±0.79	8.78±1.07	30.18±1.61	31.15±1.43	26.20±7.28	4.70±0.36	90.63

Table 5: Results for the amazon polarity dataset. The results displayed for P2L obtained the lowest kl bound on the error, whilst the baseline was chosen by the lowest validation error.

Model	Error (%)					Cross-entropy loss				
	Train	Validation	Test	kl bound	Binomial bound	Train	Validation	Test	kl bound	$\lvert\mathbf{i}\rvert$
P2L	4.25	5.24	5.37	14.67	21.63	0.1180	0.1465	0.1486	0.9992	1472
Baseline	3.13	4.07	4.19	-	-	0.0902	0.1151	0.1156	-	-

our knowledge, no sample compression bounds exist for this setting. We adapt the P2L algorithm to this regression problem (see Algorithm 2 in appendix), which differs from the original one, designed only for classification problems where zero training error is achievable (consistent case). At each P2L iteration, we add a single datapoint to the compression set in order to train the random forest. The selected datapoint is the one with the largest root mean squared error (RMSE). Then, the trees are retrained completely on the compression set. As the minimal RMSE that can be achieved is dependent on the dataset, setting a predetermined threshold is not a suitable stopping criterion. Thus, we train the model until the validation loss has not decreased for a number of iterations. To compute the bounds, we need the loss to be either bounded or sub-Gaussian. As tree-based models predict the mean of the targets of each datapoint assigned to a leaf, their outputs are bounded by the extrema of the data. Thus, if we assume that the target space is bounded, the loss will be bounded. To compute the kl bound, we assume that the target space is bounded. The maximum value of the loss $\ell^{\max}$ on each dataset is reported in Table 4. To compute the linear bound, we assume that the loss is sub-Gaussian. We discuss in more details these assumptions and the way of defining the extrema in Appendix A.1.3.

We present the results in Table 4. The models are selected based on the smallest kl bound. We observe that the models trained with P2L are able to obtain competitive results with respect to the test error of our baseline, random forests trained on the whole dataset. We report these results in the column "baseline test loss" of Table 4, where the models were chosen by their validation loss. As the value of the bounds is always much smaller than $\ell^{\max}$, we can observe that our bounds are tight and non-vacuous. The generalization guarantees given by the bound using the linear function are competitive to the kl and are even tighter on the Parkinson dataset.

Experiments with regression trees can be found in Appendix A.1.3. Training trees using P2L lead to underfitted trees that were not competitive w.r.t. the baseline (see Table 11). To the best of our knowledge, these results are the first generalization bounds for regression trees.

4.4 Amazon polarity

Finally, we train DistilBERT [49] on the Amazon reviews polarity dataset [66]. Using P2L, we fine-tune the pretrained language model on 10% of the dataset, for a total of 360k datapoints, and evaluate the model on the test set, which comprises 400k datapoints. As the model and dataset are quite large, we run the experiments for each hyperparameter combination only once. We pre-train the model on half of the training dataset and then use P2L on the other half of the training set. We add 32 datapoints at a time in the compression set and early stop the training of the model if its validation loss has not decreased for 20 epochs. In this experiment, we study our new kl bound on the zero-one loss and on the bounded cross-entropy loss. Moreover, we compute the binomial approximation bound of Corollary 9. The P2L bound (Theorem 2) is invalid in this setting, as the model doesn't reach zero errors. The PAC-Bayesian bound of Theorem 10 could be computed on both metrics, but it would necessitate to train 132M parameters (twice the number of parameters of DistilBERT). Many new generalization bounds and approaches were presented for very large models [34, 35, 65, 54], such as large language models. However, most approaches are not suited for classification and regression, as they are derived for language modeling objectives.

We present the results in Table 5. First of all, we observe that training the model using P2L only incurs a loss of about a percent for the train, validation and test error. It achieves this error whilst being trained on about 1% of the dataset, as the compression set size is 1472 and the training set size is 144k. Both for the error and the cross-entropy loss, the bound is tight and non-vacuous. Our bound is much tighter than the binomial approximation bound, with a certificate of 14.67% for a train error of 4.25%. Despite the 66M parameters of DistilBERT, we are able to obtain tight generalization guarantees by simply changing the training loop of the model.

5 Conclusion

We developed novel generalization bounds for real-valued losses and sample-compressed predictors. These bounds leverage the comparator functions studied in the PAC-Bayes theory. We provide results for bounded and unbounded losses, under different assumptions. We empirically verified the tightness of the proposed bounds, showing that it is almost as tight as the binomial tail inversion, which, however, holds only for a less general setting. We trained neural networks with 66M parameters and obtained tight guarantees, without suffering from the cost of the number of parameters. This highlights an important asset of the sample compression framework: two models achieving the same empirical loss using the same amount of datapoints (compression set size) share the same guarantees (bound value), regardless of their size in terms of the number of trainable parameters.

In future works, we could leverage the possibility of having a message in the compression scheme, by training models such as the set covering machine [31] or decision trees [51], which both use binary sequences to specify how to reconstruct the model. Finally, although P2L is generally able to train good performing models, it is unclear that its sample selection heuristic is optimal for neural networks. Trying different heuristics, e.g., that optimize for sample diversity, could lead to improved performance and guarantees for the models.

Acknowledgements

Mathieu Bazinet is supported by a FRQNT B2X scholarship (343192). Pascal Germain is supported by the Canada CIFAR AI Chair Program and the NSERC Discovery grant RGPIN-2020-07223.

Disclosure of Interests

The authors have no competing interests relative to the content of this article.

References

[1] Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, CK Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Michael Suo, Phil Tillet, Eikan Wang, Xiaodong Wang, William Wen, Shunting Zhang, Xu Zhao, Keren Zhou, Richard Zou, Ajit Mathews, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM, 2024.

[2] Idan Attias, Steve Hanneke, Aryeh Kontorovich, and Menachem Sadigurschi. Agnostic sample compression schemes for regression. In *Forty-first International Conference on Machine Learning*, 2018.

[3] Shai Ben-David, Alex Bie, Clément L Canonne, Gautam Kamath, and Vikrant Singhal. Private distribution learning with public data: The view from sample compression. *Advances in Neural Information Processing Systems*, 36, 2024.

[4] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com.

[5] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[6] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[7] Thomas F. Brooks, D. Stuart Pope, and Michael A. Marcolini. Airfoil Self-Noise. UCI Machine Learning Repository, 1989. DOI: https://doi.org/10.24432/C5VW2C.

[8] Thomas F. Brooks, D. Stuart Pope, and Michael A. Marcolini. Airfoil self-noise and prediction. Technical report, 1989.

[9] Marco C Campi and Simone Garatti. Compression, generalization and learning. *Journal of Machine Learning Research*, 24(339):1–74, 2023.

[10] Ioar Casado, Luis A Ortega, Andrés R Masegosa, and Aritz Pérez. Pac-bayes-chernoff bounds for unbounded losses. *ArXiv preprint*, abs/2401.01148, 2024.

[11] Olivier Catoni. Pac-bayesian supervised classification: the thermodynamics of statistical learning. *ArXiv preprint*, abs/0712.0248, 2007.

[12] Ofir David, Shay Moran, and Amir Yehudayoff. Supervised learning through the lens of compression. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2784–2792, 2016.

[13] Gintare Karolina Dziugaite and Daniel M. Roy. Data-dependent pac-bayes priors via differential privacy. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8440–8450, 2018.

[14] William Falcon and The PyTorch Lightning team. PyTorch Lightning, 2019.

[15] Sally Floyd and Manfred Warmuth. Sample compression, learnability, and the vapnik-chervonenkis dimension. *Machine learning*, 21(3):269–304, 1995.

[16] Andrew Foong, Wessel Bruinsma, David Burt, and Richard Turner. How tight can pac-bayes be in the small data regime? *Advances in Neural Information Processing Systems*, 34:4093–4105, 2021.

[17] Pascal Germain, Alexandre Lacasse, François Laviolette, and Mario Marchand. Pac-bayesian learning of linear classifiers. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman, editors, *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 353–360. ACM, 2009.

[18] Pascal Germain, Alexandre Lacasse, François Laviolette, Mario Marchand, and Jean-Francis Roy. Risk bounds for the majority vote: From a pac-bayesian analysis to a learning algorithm. *The Journal of Machine Learning Research*, 16:787–860, 2015.

[19] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[20] Maxime Haddouche, Benjamin Guedj, Omar Rivasplata, and John Shawe-Taylor. Pac-bayes unleashed: Generalisation bounds with unbounded losses. *Entropy*, 23(10):1330, 2021.

[21] Steve Hanneke and Aryeh Kontorovich. Stable sample compression schemes: New applications and an optimal SVM margin bound. In *Algorithmic Learning Theory*, pages 697–721. PMLR, 2021.

[22] Steve Hanneke, Aryeh Kontorovich, and Menachem Sadigurschi. Efficient Conversion of Learners to Bounded Sample Compressors. *Proceedings of Machine Learning Research vol*, 75:1–21, 2018.

[23] Steve Hanneke, Aryeh Kontorovich, and Menachem Sadigurschi. Sample Compression for Real-Valued Learners. In *Proceedings of the 30th International Conference on Algorithmic Learning Theory*, pages 466–488. PMLR, 2019.

[24] Steve Hanneke, Shay Moran, and Waknine Tom. List sample compression and uniform convergence. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 2360–2388. PMLR, 2024.

[25] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[26] Fredrik Hellström and Benjamin Guedj. Comparing comparators in generalization bounds. In *International Conference on Artificial Intelligence and Statistics*, pages 73–81. PMLR, 2024.

[27] J. Kahane. Propriétés locales des fonctions à séries de fourier aléatoires. *Studia Mathematica*, 19(1):1–25, 1960.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[29] John Langford. Tutorial on practical prediction theory for classification. *Journal of machine learning research*, 6(3), 2005.

[30] John Langford and Matthias Seeger. *Bounds for averaging classifiers*. School of Computer Science, Carnegie Mellon University, 2001.

[31] François Laviolette, Mario Marchand, and Mohak Shah. Margin-Sparsity Trade-Off for the Set Covering Machine. In *Machine Learning: ECML 2005*, volume 3720, pages 206–217. Springer Berlin Heidelberg, 2005.

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

[33] Nick Littlestone and Manfred Warmuth. Relating data compression and learnability. 1986.

[34] Sanae Lotfi, Marc Anton Finzi, Yilun Kuang, Tim G. J. Rudner, Micah Goldblum, and Andrew Gordon Wilson. Non-vacuous generalization bounds for large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32801–32818. PMLR, 21–27 Jul 2024.

[35] Sanae Lotfi, Yilun Kuang, Brandon Amos, Micah Goldblum, Marc Finzi, and Andrew Gordon Wilson. Unlocking tokens as data points for generalization bounds on larger language models, 2024.

[36] Mario Marchand, Mohak Shah, John Shawe-Taylor, and Marina Sokolova. The set covering machine with data-dependent half-spaces. In Tom Fawcett and Nina Mishra, editors, *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pages 520–527. AAAI Press, 2003.

[37] Mario Marchand and John Shawe-Taylor. The set covering machine. *Journal of Machine Learning Research*, 3(4-5):723–746, 2002.

[38] Mario Marchand and Marina Sokolova. Learning with decision lists of data-dependent features. *Journal of Machine Learning Research*, 6(4), 2005.

[39] Andreas Maurer. A note on the pac bayesian theorem. *arXiv preprint cs/0411099*, 2004.

[40] David A McAllester. Some PAC-Bayesian theorems. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 230–234, 1998.

[41] Shay Moran, Ido Nachum, Itai Panasoff, and Amir Yehudayoff. On the perceptron's compression. In *Beyond the Horizon of Computability: 16th Conference on Computability in Europe, CiE 2020, Fisciano, Italy, June 29–July 3, 2020, Proceedings 16*, pages 310–325. Springer, 2020.

[42] Shay Moran and Amir Yehudayoff. Sample compression schemes for vc classes. *Journal of the ACM (JACM)*, 63(3):1–10, 2016.

[43] Dario Paccagnan, Marco Campi, and Simone Garatti. The pick-to-learn algorithm: Empowering compression for tight generalization bounds and improved post-training performance. *Advances in Neural Information Processing Systems*, 36, 2024.

[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[45] María Pérez-Ortiz, Omar Rivasplata, John Shawe-Taylor, and Csaba Szepesvári. Tighter risk certificates for neural networks. *Journal of Machine Learning Research*, 22(227):1–40, 2021.

[46] Borja Rodríguez-Gálvez, Ragnar Thobaben, and Mikael Skoglund. More pac-bayes bounds: From bounded losses, to losses with general tail behaviors, to anytime validity. *Journal of Machine Learning Research*, 25(110):1–43, 2024.

[47] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[48] Benjamin IP Rubinstein and J Hyam Rubinstein. A geometric approach to sample compression. *Journal of Machine Learning Research*, 13(4), 2012.

[49] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[50] Matthias Seeger. Pac-bayesian generalisation error bounds for gaussian process classification. *Journal of machine learning research*, 3(Oct):233–269, 2002.

[51] Mohak Shah. Sample compression bounds for decision trees. In Zoubin Ghahramani, editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 799–806. ACM, 2007.

[52] Christopher Snyder and Sriram Vishwanath. Sample compression, support vectors, and generalization in deep learning. *IEEE Journal on Selected Areas in Information Theory*, 1(1):106–120, 2020.

[53] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[54] Jingtong Su, Julia Kempe, and Karen Ullrich. Mission impossible: A statistical perspective on jailbreaking llms, 2024.

[55] Pnar Tfekci and Heysem Kaya. Combined Cycle Power Plant. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C5002N.

[56] Athanasios Tsanas and Max Little. Parkinsons Telemonitoring. UCI Machine Learning Repository, 2009. DOI: https://doi.org/10.24432/C5ZS3N.

[57] Athanasios Tsanas, Max Little, Patrick McSharry, and Lorraine Ramig. Accurate telemonitoring of parkinson's disease progression by non-invasive speech tests. *Nature Precedings*, pages 1–1, 2009.

[58] Pınar Tüfekci. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *International Journal of Electrical Power & Energy Systems*, 60:126–140, 2014.

[59] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

[60] Quanzeng Wang, Yangling Zhou, Pejman Ghassemi, Dwith Chenna, Michelle Chen, Jon Casamento, Joshua Pfefer, and David Mcbride. Facial and oral temperature data from a large set of human subject volunteers, 2023.

[61] Quanzeng Wang, Yangling Zhou, Pejman Ghassemi, David McBride, Jon P Casamento, and T Joshua Pfefer. Infrared thermography for measuring elevated body temperature: clinical accuracy, calibration, and evaluation. *Sensors*, 22(1):215, 2021.

[62] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Perric Cistac, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-Art Natural Language Processing. pages 38–45. Association for Computational Linguistics, October 2020.

[63] I-C Yeh. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete research*, 28(12):1797–1808, 1998.

[64] I-Cheng Yeh. Concrete Compressive Strength. UCI Machine Learning Repository, 1998. DOI: https://doi.org/10.24432/C5PK67.

[65] Oussama Zekri, Ambroise Odonnat, Abdelhakim Benechehab, Linus Bleistein, Nicolas Boullé, and Ievgen Redko. Large language models as markov chains, 2024.

[66] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

A Experiments

The experiments were run on two different devices. The experiments with PBB algorithm and the regression datasets were run on Python 3.12.2 on a computer with a NVIDIA GeForce RTX 4090. The experiments on MNIST were run on Python 3.12.3 on a computer with a NVIDIA GeForce RTX 2080 Ti. The libraries used for each environment can be found with the code. Notably, we use PyTorch [1] (BSD 3-Clause License), Lightning [14] (Apache 2.0 license), Weights and Biases [4] (MIT License), Scikit-Learn [44] (BSD 3-Clause License), NumPy [25] (NumPy license) and Transformer [62] (Apache 2.0 license). For all experiments, we run the code with the following seeds : $[1, 2, 3, 4, 42]$.

We give information on the datasets used in the experiments.

For the classification problems, we use the MNIST dataset [32] (MIT License) and the amazon polarity dataset [66] (Apache 2.0 License). All MNIST derived-dataset are composed of 784 real-valued features. For the multi-class classification problems on MNIST, we denote MNIST ($p\%$) to say that we pre-train the model on $p\%$ of the data, where p is a hyperparameter. For the Amazon polarity dataset, we chose 10% of the dataset to create a 360000 datapoints dataset. We then use 50% to pre-train the model and split the rest into a training and validation set. The datapoints are textual reviews and the labels are binary. The descriptions of the dataset are presented in Table 6.

Table 6: Description of the datasets used for classification problems.

Dataset	Pretrain set size	Train set size	Validation set size	Test set size
MNIST (10%)	6000	48000	6000	10000
MNIST (20%)	12000	42000	6000	10000
MNIST (50%)	30000	24000	6000	10000
MNIST08	0	10597	1177	1954
MNIST17	0	11707	1300	2163
MNIST23	0	10881	1208	2042
MNIST49	0	10612	1179	1991
MNIST56	0	10206	1133	1850
Amazon Polarity	180000	144000	36000	400000

For the regression problems, we train our models on five datasets : the *Combined Cycle Power Plant* [58, 55], the *Infrared Thermography Temperature* [61, 60], the *Airfoil Self-Noise* [8, 7], the *Parkinsons Telemonitoring* [57, 56], the *Concrete Compressive Strength* [63, 64]. The descriptions of the dataset are presented in Table 7. All datasets were chosen from the UCI dataset repository. Powerplant, Airfoil, Parkinson and Concrete are under the CC-BY 4.0 license. The Infrared dataset is under the CC0 license.

Table 7: Description of the datasets used for regression problems.

Dataset	Train set size	Validation set size	Test set size	Number of features
Powerplant	7751	861	956	4
Infrared	827	91	102	33
Airfoil	1218	135	150	5
Parkinson	4760	528	587	19
Concrete	835	92	103	8

A.1 Hyperparameter grids

In this section, we present the hyperparameter grids for all the experiments.

In all experiments, we use $\delta = 0.01$ and a batch size of 64. After each iteration of P2L, we train the model for 200 epochs or until the validation loss has not improved for three epochs.

A.1.1 Binary MNIST problems

For the binary MNIST problems, we used the following hyperparameters.

- Model type : [MLP, CNN]
- Dropout probability : [0.1, 0.2]
- Training learning rate : $[1\mathrm{e}-2, 1\mathrm{e}-3, 5\mathrm{e}-3, 1\mathrm{e}-4]$

The MLP is composed of three hidden fully connected layers of 600 neurons and the CNN is composed of two convolutional layers and two fully connected layers. We use ReLU activations [19], dropout layers [53] and the Adam optimizer [28] with the default parameters $\beta = (0.9, 0.999)$.

At each iteration, the P2L algorithm adds one datapoint to the compression set.

For the baselines, we train the same models with the same hyperparameters for 200 epochs or until the model achieves zero errors on the training set.

In the following tables, we present the results for the MLP, both trained fully using P2L and early-stopped, respectively in Table 8 and in Table 9. Moreover, in Fig. 2, we present the results not present in Fig. 1.

Table 8: Results for the MLPs trained using P2L on the binary MNIST problems. The results displayed obtained the tightest P2L bound. All metrics presented are in percents (%), with the exception of $|\mathbf{i}|$.

Dataset	Validation error	Test error	kl bound	Binomial bound	P2L bound	$\vert\mathbf{i}\vert$	Baseline test error
MNIST08	0.41±0.14	0.40±0.08	6.56±0.30	6.51±0.30	1.42±0.08	128.2±7.4	0.34±0.07
MNIST17	0.37±0.14	0.47±0.17	4.93±0.27	4.89±0.27	1.01±0.07	99.0±7.0	0.33±0.08
MNIST23	0.87±0.24	0.58±0.12	12.21±0.29	12.17±0.29	3.06±0.09	296.6±9.4	0.36±0.14
MNIST49	1.19±0.33	1.04±0.10	14.41±0.05	14.37±0.05	3.78±0.02	361.4±1.9	0.96±0.14
MNIST56	0.68±0.17	0.65±0.05	10.35±0.31	10.30±0.31	2.48±0.09	223.0±8.9	0.59±0.15

Table 9: Results for the MLPs trained using P2L on the binary MNIST problems and stopped at the iteration with the minimum kl bound. The results displayed obtained the tightest kl bound. All metrics presented are in percents (%), with the exception of $|\mathbf{i}|$.

Dataset	Validation error	Test error	kl bound	Binomial bound	Train error	$\vert\mathbf{i}\vert$	Baseline test error
MNIST08	1.11±0.52	1.04±0.67	5.46±0.53	7.77±1.64	0.85±0.71	59.2±34.4	0.34±0.07
MNIST17	0.88±0.39	0.80±0.29	4.02±0.36	5.49±0.77	0.50±0.26	44.0±15.0	0.33±0.08
MNIST23	1.93±0.49	1.59±0.43	10.86±0.19	13.23±0.74	1.27±0.41	146.0±25.8	0.36±0.14
MNIST49	2.28±0.53	2.07±0.58	13.14±0.32	15.08±0.99	1.22±0.47	202.0±30.6	0.96±0.14
MNIST56	1.97±0.53	1.88±0.44	8.85±0.58	11.78±1.44	1.38±0.61	92.0±27.9	0.59±0.15

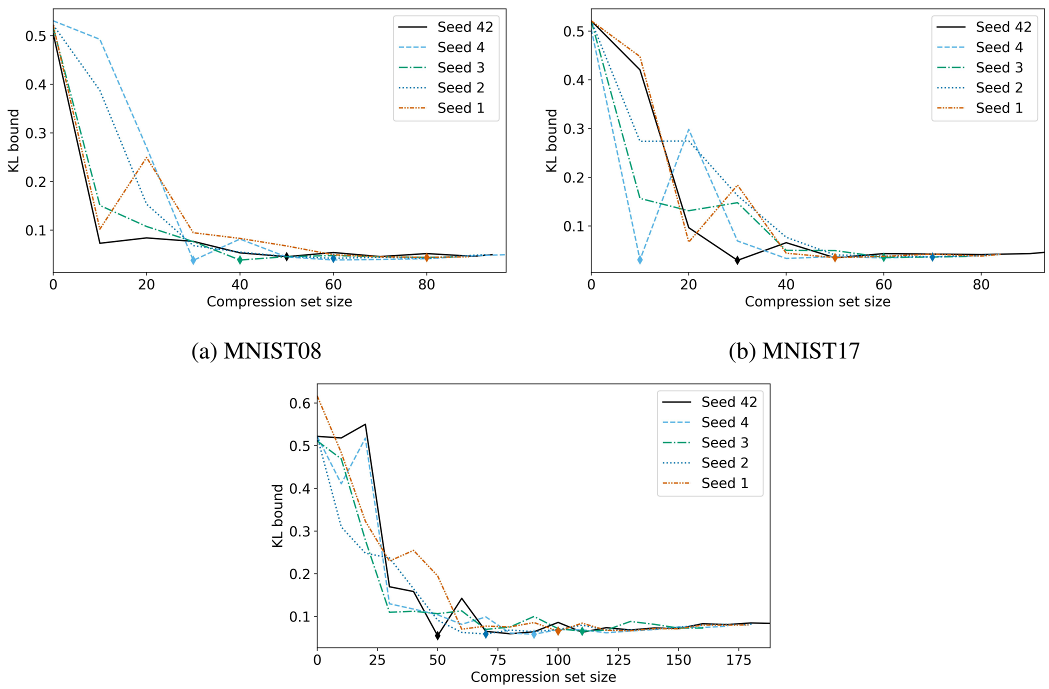
A.1.2 MNIST problems

We train a convolutional neural network over the 10-class MNIST dataset with the following hyperparameters.

- Size of pretraining set : $[10\%, 20\%, 50\%]$
- Pretraining epochs : [50, 100]
- Pretraining learning rate : $[1\mathrm{e}-2, 1\mathrm{e}-3, 1\mathrm{e}-4]$
- Dropout probability : [0.1, 0.2]
- Training learning rate : $[1\mathrm{e}-3, 5\mathrm{e}-3, 1\mathrm{e}-4]$

At each iteration, the P2L algorithm adds 32 datapoints to the compression set. To compute bounds for the cross-entropy loss, we clamp the log-probabilities to be greater or equal than $\ln\left(10^{-5}\right)$ [45, 13], as follows :

$$\ell(h, \boldsymbol{x}, y) = -\max\left(\ln\left(10^{-5}\right), \ln\left(\frac{\exp(h(\boldsymbol{x})_y)}{\sum_{c=1}^{C}\exp(h(\boldsymbol{x})_c)}\right)\right),$$



(a) MNIST08

(b) MNIST17

(c) MNIST23

Figure 2: Illustration of the behavior of the kl bound throughout P2L iterations for the five different seeds of the hyperparameter combination that achieved the minimal P2L bound. We mark the minimal kl bound for each seed with a diamond (⧫).

Table 10: Train metrics on MNIST (risk 01) (in percents)

Model	Train error	Test Error	kl bound	Binomial bound	Compression set size	Baseline test error
P2L	0.0±0.0	1.06±0.10	6.15± 1.51	6.13±1.51	275.20± 82.46	0.0108±0.0010
PBB	1.67 ± 0.07	1.05±0.05	1.94±0.07	-	-	

where $h(\boldsymbol{x}) = (h(\boldsymbol{x})_1, \dots, h(\boldsymbol{x})_C)$ is the output of the neural network and C is the number of classes. The loss then takes values between $[0, -\ln(10^{-5})]$. We will use the same bounded cross-entropy loss for the following experiments.

For the baseline, we train the same model with the same hyperparameters for 200 epochs or until the model achieves zero errors on the training set.

For the PAC-Bayes with Backprop (PBB) algorithm, we used the GitHub repository associated to the article of [45]. We used the hyperparameter grid proposed by the article, with the exception of the dropout, which we kept similar to the other experiments. We used $\delta = \delta' = 0.01$ to compute Theorem 10. We used $m = 5000$ Monte Carlo sampling instead of the value $m = 150000$ found in the code, as it takes several hours to run.

- Scale parameter of the prior distribution : $[0.1, 0.05, 0.04, 0.03, 0.02, 0.01, 0.005]$
- Training learning rate : $[1e-3, 5e-3, 1e-2]$
- Pre-training learning rate : $[1e-3, 5e-3, 1e-2]$
- Momentum : $[0.95, 0.99]$
- Dropout probability : $[0.1, 0.2]$

A.1.3 Regression problems

We trained decision trees and forests on the datasets, using P2L to train the forests on one datapoint at a time. We trained the models until their validation loss hasn't decreased for 10 or 20 epochs. We summarize this idea in Algorithm 2. We denote the RMSE as $\ell^{\text{RMSE}}(h, \boldsymbol{x}, y) = \sqrt{(h(\boldsymbol{x}) - y)^2}$ and the empirical risk on the dataset

$$\mathcal{L}_S^{\text{RMSE}}(h) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(h(\boldsymbol{x}_i) - y_i)^2}.$$

For tree-based models, we chose h_0 to simply output zeroes for all entries. We use COUNTER and $\widehat{\mathcal{L}}_{\text{BEST}}$ as variables to stop the training when the loss hasn't decreased for T epochs.

Algorithm 2: Pick-To-Learn for regression problems

```
Input      : $T$, the number of iterations before stopping.
Initialize : $S_{\mathbf{i}} = \emptyset$.
Initialize : $h_{\mathbf{i}} = h_0$.
Initialize : $\mathcal{L}_{\text{BEST}} = \infty$.
Initialize : $\texttt{COUNTER} = 0$.
Initialize : $(\overline{\boldsymbol{x}}, \overline{y}) = \arg\max_{(\boldsymbol{x},y)\in S} \ell^{\text{RMSE}}(h_0, \boldsymbol{x}, y)$
while $\texttt{COUNTER} \leq T$ do
    $S_{\mathbf{i}} \leftarrow S_{\mathbf{i}} \cup \{(\overline{\boldsymbol{x}}, \overline{y})\}$
    $h_{\mathbf{i}} \leftarrow A(S_{\mathbf{i}})$
    $(\overline{\boldsymbol{x}}, \overline{y}) \leftarrow \arg\max_{(\boldsymbol{x},y)\in S_{\mathbf{i}^c}} \ell^{\text{RMSE}}(h_{\mathbf{i}}, \boldsymbol{x}, y)$
    if $\mathcal{L}_{S_{\mathbf{i}^c}}^{RMSE}(h_{\mathbf{i}}) < \mathcal{L}_{BEST}$ then
        $\mathcal{L}_{\text{BEST}} \leftarrow \mathcal{L}_{S_{\mathbf{i}^c}}^{\text{RMSE}}(h_{\mathbf{i}})$
        $\texttt{COUNTER} \leftarrow 0$
    else
        $\texttt{COUNTER} \leftarrow \texttt{COUNTER} + 1$
    end
end
return $h_{\mathbf{i}}$
```

We now present the hyperparameter grid.

- Maximum depth of the trees : $[5, 10]$
- Minimum samples to split : $[2, 3, 4]$
- Minimum samples to create a leaf : $[1, 2, 3]$
- Cost-Complexity pruning parameter : $[0.0, 0.05, 0.1, 0.2, 0.5, 1, 2]$
- Number of epochs before stopping : $[10, 20]$

For the decision forests, we choose the number of estimators in $[50, 100]$. For the baselines, we train the same model with the same hyperparameters on the whole dataset.

The results for the decision trees can be found in Table 11. Using only P2L to train the trees leads to underfitted trees, as the model is not complex enough to use only a few datapoints to train a complete model.

When computing the bounds, we need to bound the loss, as the RMSE is not bounded. However, the regression trees cannot predict a value bigger (respectively smaller) than the biggest (respectively smallest) target value found in the dataset. Thus, the loss is bounded by the biggest and smallest target values found in the dataset S. To compute the kl bound, we add an assumption on the data-generating distribution $\mathcal{D}$, which is that the target values of $\mathcal{D}$ are bounded by :

$$\min_{(\boldsymbol{x},y)\sim S} y - p\left(\max_{(\boldsymbol{x},y)\sim S} y - \min_{(\boldsymbol{x},y)\sim S} y\right) \leq \min_{(\boldsymbol{x},y)\sim \mathcal{D}} y \leq \max_{(\boldsymbol{x},y)\sim \mathcal{D}} y \leq \max_{(\boldsymbol{x},y)\sim S} y + p\left(\max_{(\boldsymbol{x},y)\sim S} y - \min_{(\boldsymbol{x},y)\sim S} y\right)$$

where $p \in [0\%, 100\%]$. For the experiments, we choose $p = 10\%$. If $\min_{(\boldsymbol{x},y)\sim S} y = 0$, then we simply lower bound by 0.

To compute the linear bound, we assume that the distribution is ς^2-sub-Gaussian with $\varsigma = \frac{1}{2}\big(\max_{(\boldsymbol{x},y)\sim S} y - \min_{(\boldsymbol{x},y)\sim S} y\big)$.

These assumptions are restrictive and we would rather have no assumption on the data-generating distribution. In some settings, we can remove the assumption on the dataset by having expert knowledge on the distribution. For example, if you predict a probability, the target domain is $[0, 1]$.

We report the lower bounds, minimum values in the training set, maximum values in the training set and upper bounds for each dataset in Table 12.

Table 11: Results for the decision trees trained using P2L. We report the RMSE achieved by the models and the generalization bounds on the RMSE.

Dataset	Train loss	Validation loss	Test loss	kl bound	Linear bound	$\lvert\mathbf{i}\rvert$	Baseline test loss	$\ell^{\max}$
Powerplant	11.66±3.00	11.83±3.12	12.00±3.04	23.40±1.59	24.15±1.54	72.80±38.32	4.07±0.13	90.6
Infrared	0.77±0.11	0.80±0.08	0.76±0.13	1.63±0.14	1.55±0.11	12.80±0.40	0.27±0.03	4.26
Airfoil	11.02±1.71	10.89±1.38	11.10±1.93	18.87±1.97	18.14±1.75	12.20±0.40	3.01±0.19	45.13
Parkinson	14.75±1.86	14.53±2.19	14.90±2.22	18.86±1.95	18.28±1.86	12.00±0.00	3.20±0.15	41.37
Concrete	26.44±1.99	27.40±1.85	27.08±1.68	46.56±1.74	45.00±1.72	14.00±1.26	6.22±0.91	90.63

Table 12: Minimum and maximum values used to compute the bound on regression problems.

Dataset	Lower bound	Minimum value	Maximum value	Upper bound
Powerplant	412.71	420.26	495.76	503.31
Infrared	35.40	35.75	39.3	39.66
Airfoil	99.62	103.38	140.99	144.75
Parkinson	1.59	5.04	39.51	42.96
Concrete	0	2.33	82.6	90.63

A.1.4 Amazon Polarity

We trained DistilBERT on the Amazon Reviews Polarity dataset. We use a subset of 10% of the real dataset, amounting to 360000 datapoints. 180000 are used for pretraining the model, 144000 for training and 36000 for validation. We use the given test set of 400000 datapoints. Using P2L, we add 32 datapoints at a time to the compression set and stop the training when the validation loss hasn't decreased in 20 iterations. For both P2L and the baseline, which was trained for 200 epochs or until it reached 0 errors on the training dataset, we use the following hyperparameter grid :

- Number of pretraining epochs : $[2, 5]$
- Pretraining learning rate : 2e-5
- Dropout probability : $[0.1, 0.2]$
- Training learning rate : [1e-6, 1e-7, 1e-8]

For this model, we trained using only one seed.

B Theoretical results from the literature

Corollary 9 ([31], Corollary 1)**.** *For any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, for any set of messages $\{M(\mathbf{i}) \,\forall\, \mathbf{i} \in I\}$, for any deterministic reconstruction function $\mathcal{R}$ that outputs sample-compressed predictors $h \in \mathcal{H}$ and for any $\delta \in (0, 1]$, with probability at least $1 - \delta$ over the draw of $S \sim \mathcal{D}^n$, we have*

$$\forall\, \mathbf{i} \in I, \forall \sigma \in M(\mathbf{i}) \,:\, R_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)) \leq 1 - \exp\left(\frac{-1}{n - |\mathbf{i}| - \kappa}\left[\ln\binom{n - |\mathbf{i}|}{\kappa} + \ln\left(\frac{\binom{n}{|\mathbf{i}|}}{\zeta(|\mathbf{i}|) P_{M(\mathbf{i})}(\sigma)\delta}\right)\right]\right),$$

with $\kappa = nR_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma))$.

Theorem 10 ([45])**.** *For any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, for any set $\mathcal{H}$ of predictors $h : \mathcal{X} \to \mathcal{Y}$, for any loss $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to [0, 1]$, for any dataset-independent prior distribution $\mathcal{P}$ on $\mathcal{H}$, for any $\delta, \delta' \in (0, 1]$, with probability at least $1 - \delta - \delta'$ over the draw of $S \sim \mathcal{D}^n$ and a set of m predictors $h_1, \ldots, h_m \sim \mathcal{Q}_S$, where $\mathcal{Q}_S$ is a dataset-dependent posterior distribution over $\mathcal{H}$, we have*

$$\mathop{\mathbb{E}}_{h\sim\mathcal{Q}} \mathcal{L}_{\mathcal{D}}(h) \leq \mathrm{kl}^{-1}\left(\mathrm{kl}^{-1}\left(\frac{1}{m}\sum_{i=1}^{m} \widehat{\mathcal{L}}_S(h_i), \frac{1}{m}\log\frac{2}{\delta'}\right), \frac{1}{n}\left[\mathrm{KL}(\mathcal{Q}\,||\,\mathcal{P}) + \ln\left(\frac{2\sqrt{n}}{\delta}\right)\right]\right).$$

C Proofs

C.1 Proof of the main result

Theorem 3. *For any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, for any family of set of messages $\{M(\mathbf{i}) \,|\, \mathbf{i} \in \mathcal{P}(n)\}$, for any deterministic reconstruction function $\mathcal{R}$ that outputs sample-compressed predictors $h \in \mathcal{H}$, for any loss $\ell : \mathcal{H} \times \mathcal{X} \times Y \to \mathbb{R}$, for any comparator function $\Delta : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and for any $\delta \in (0, 1]$, with probability at least $1 - \delta$ over the draw of $S \sim \mathcal{D}^n$, we have*

$$\forall \mathbf{i} \in \mathcal{P}(n), \sigma \in M(\mathbf{i}) :$$

$$\Delta\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma))\Big) \leq \frac{1}{|\,\mathbf{i}^c\,|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{\mathcal{E}_\Delta(\mathbf{i}, \sigma)}{\zeta(|\mathbf{i}|) P_{M(\mathbf{i})}(\sigma)\delta}\right)\right]$$

with

$$\mathcal{E}_\Delta(\mathbf{i}, \sigma) = \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \; \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{|\mathbf{i}^c|}} e^{|\mathbf{i}^c|\Delta\big(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\big)}.$$

Before proving Theorem 3, we need to restate Chernoff's bound in a way that will be useful to prove Theorem 3.

Lemma 11 (Chernoff's bound)**.** *For $t > 0$ and X a random variable :*

$$\mathbb{P}\left(X \leq \frac{1}{t}\left[\ln \mathbb{E}\, e^{tX} + \ln\frac{1}{\delta}\right]\right) \geq 1 - \delta.$$

Proof of Lemma 11. Chernoff's bound states that for a random variable X, any $t > 0$ and $\epsilon > 0$, we have :

$$\mathbb{P}\,(X > \epsilon) \leq e^{-t\epsilon}\, \mathbb{E}\, e^{tX}$$

By choosing $\delta = e^{-t\epsilon}\, \mathbb{E}\, e^{tX}$, we have :

$$\begin{aligned} &\delta = e^{-t\epsilon}\, \mathbb{E}\, e^{tX} \\ \Longleftrightarrow\ & e^{t\epsilon} = \frac{1}{\delta}\, \mathbb{E}\, e^{tX} \\ \Longleftrightarrow\ & t\epsilon = \ln\frac{1}{\delta}\, \mathbb{E}\, e^{tX} \\ \Longleftrightarrow\ & \epsilon = \frac{1}{t}\left[\ln \mathbb{E}\, e^{tX} + \ln\frac{1}{\delta}\right] \end{aligned}$$

Thus, we have :

$$\mathbb{P}\left(X > \frac{1}{t}\left[\ln \mathbb{E}\, e^{tX} + \ln\frac{1}{\delta}\right]\right) \leq \delta.$$

□

Proof of Theorem 3. We start by defining a sample-compressed set. Given a dataset $S \sim \mathcal{D}^n$ and $\mathcal{H}$ a predictor set, we consider the following subset of $\mathcal{H}$, that contains only sample-compressed predictors :

$$\widehat{\mathcal{H}}_S := \{\mathcal{R}(S_{\mathbf{i}}, \sigma) |\, \mathbf{i} \in I, \sigma \in M(\mathbf{i})\} \subseteq \mathcal{H}\,.$$

For any vector of indices $\mathbf{i} \in I$ and a message $\sigma \in M(\mathbf{i})$, when given a dataset S, we fully define a predictor $\mathcal{R}(S_{\mathbf{i}}, \sigma) \in \widehat{\mathcal{H}}_S$. For a specific pair $(\mathbf{i}, \sigma)$, let's study the value of $\Delta\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma))\Big)$, a realization of a random variable of mean

$$\mathop{\mathbb{E}}_{T\sim\mathcal{D}^n} \Delta\Big(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}}, \sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}}, \sigma))\Big) = \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} \Delta\Big(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}}, \sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}}, \sigma))\Big).$$

With $\delta_{\mathbf{i}}^{\sigma} \in (0, 1)$ and $t > 0$, using Chernoff's bound as stated in Lemma 11, we have :

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^n}\left(\Delta\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(h_{\mathbf{i}}^{\sigma}), \mathcal{L}_{\mathcal{D}}(h_{\mathbf{i}}^{\sigma})\Big) \leq \frac{1}{t}\left[\ln \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} e^{t\Delta\left(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)} + \ln\frac{1}{\delta_{\mathbf{i}}^{\sigma}}\right]\right)$$
$$\geq 1 - \delta_{\mathbf{i}}^{\sigma}$$

Using the union bound, we get a bound that is valid for all pairs $(\mathbf{i}, \sigma)$ simultaneously,

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^n}\left(\forall\, \mathbf{i} \in I, \sigma \in M(\mathbf{i}) : \Delta\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(h_{\mathbf{i}}^{\sigma}), \mathcal{L}_{\mathcal{D}}(h_{\mathbf{i}}^{\sigma})\Big) \leq \frac{1}{t}\left[\ln \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} e^{t\Delta\left(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)} + \ln\frac{1}{\delta_{\mathbf{i}}^{\sigma}}\right]\right)$$
$$\geq 1 - \sum_{\mathbf{i}\in I}\sum_{\sigma\in M(\mathbf{i})} \delta_{\mathbf{i}}^{\sigma}. \tag{9}$$

To obtain a valid bound, we will need to either compute or bound the following term :

$$\mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} e^{t\Delta\left(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)}.$$

As the set T is a realization of a $n - |\mathbf{i}|$ datapoints, choosing $t = n - |\mathbf{i}|$ will generally be the best value to make sure that this term is bounded. Indeed, this is a requirement for the proofs of multiple results in the PAC-Bayesian theory. Thus, we will simply always choose $t = n - |\mathbf{i}|$. With this choice made, we denote :

$$\mathcal{E}_{\Delta}(\mathbf{i}, \sigma) = \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} e^{(n-|\mathbf{i}|)\Delta\left(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)),\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)}.$$

We finish the proof by choosing the value of $\delta_{\mathbf{i}}^{\sigma}$, which needs to be defined independently of S. Choose a set $I_m = \{\mathbf{i} \in I : |\mathbf{i}| = m\}$. As we have no information on which $\mathbf{i} \in I_m$ will give the best results, we define a uniform distribution over all $\binom{n}{m}$ vectors in I_m, which gives a weight of $\binom{n}{m}^{-1} \forall\, \mathbf{i} \in I_m$. Now, we generally consider multiple sizes of compression set

$$I = \bigcup_{k=0}^{M} I_k,$$

so we need to define a probability distribution over each set I_k. We could simply choose $\frac{1}{M+1}$, but the probabilities would tend very fast to zero when we consider a large number M of compression set sizes. It is a better choice, as discussed by [38] in Section 5.2, to choose :

$$\zeta(m) = \frac{6}{\pi^2(m+1)^2}, \quad \text{with } \sum_{m=0}^{\infty}\zeta(m) = 1.$$

For any compression set $S_{\mathbf{i}}$, we define a probability distribution $P_{M(\mathbf{i})}$ over $M(\mathbf{i})$ such that $\sum_{\sigma\in M(\mathbf{i})} P_{M(\mathbf{i})}(\sigma) \leq 1$.

Thus, for a $\delta \in (0, 1)$, we define $\delta_{\mathbf{i}}^{\sigma} = \begin{pmatrix} n \\ |\mathbf{i}| \end{pmatrix}^{-1} \zeta(|\mathbf{i}|) P_{M(\mathbf{i})}(\sigma)\delta$ and

$$\sum_{\mathbf{i}\in I}\sum_{\sigma\in M(\mathbf{i})} \delta_{\mathbf{i}} = \sum_{\mathbf{i}\in I}\sum_{\sigma\in M(\mathbf{i})} \begin{pmatrix} n \\ |\mathbf{i}| \end{pmatrix}^{-1} \zeta(|\mathbf{i}|) P_{M(\mathbf{i})}(\sigma)\delta$$

$$
\begin{aligned}
&\le \sum_{\mathbf{i} \in I} \binom{n}{|\mathbf{i}|}^{-1} \zeta(|\mathbf{i}|)\delta \\
&= \sum_{m=1}^{M} \sum_{\mathbf{i} \in I_m} \binom{n}{|\mathbf{i}|}^{-1} \zeta(|\mathbf{i}|)\delta \\
&= \sum_{m=1}^{M} \zeta(|\mathbf{i}|)\delta \\
&\le \delta.
\end{aligned}
$$

Thus, we have $1 - \sum_{\mathbf{i} \in I} \sum_{\sigma \in M(\mathbf{i})} \delta_{\mathbf{i}}^{\sigma} \ge 1 - \delta$. We replace $\delta_{\mathbf{i}}^{\sigma}$ by $\binom{n}{|\mathbf{i}|}^{-1} \zeta(|\mathbf{i}|) P_{M(\mathbf{i})}(\sigma)\delta$ in Equation 9 to finish the proof.

□

C.2 Corollaries to the main result

To prove most corollaries, we are going to need the following lemma :

Lemma 12 ([39], [18]). *Let X be any random variable with values in $[0,1]$ and expectation $\mu = \mathbb{E}(X)$. Denote X the vector containing the results of n independent realizations of X. Then, consider a Bernoulli random variable X' ($\{0,1\}$-valued) of probability of success μ. Denote $X' \in \{0,1\}^n$ the vector containing the results of n independent realizations of X'.*

If function $g : [0,1]^n \to \mathbb{R}$ is convex, then

$$\mathbb{E}[g(X)] \le \mathbb{E}[g(X')].$$

We now prove our first corollary.

Corollary 4. *In the setting of Theorem 3, for any $C > 0$, for a loss function $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to [0,1]$, with probability at least $1 - \delta$ over the draw of $S \sim \mathcal{D}^n$, we have*

$$\forall\, \mathbf{i} \in \mathcal{P}(n), \sigma \in M(\mathbf{i}) :\ \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)) \le \frac{1 - \exp(-\epsilon_C(\mathbf{i}, \sigma, \delta))}{1 - e^{-C}}$$

with

$$\epsilon_C(\mathbf{i}, \sigma, \delta) = C\, \widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}}, \sigma)) + \frac{1}{n - |\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|) P_{M(\mathbf{i})}(\sigma)\delta}\right)\right].$$

Proof of Corollary 4. To prove this corollary, we need to do two things : bound $\mathcal{E}_{\Delta_C}$ and rearrange the terms. Both of these things were already proven multiple time in the PAC-Bayes theory, so we simply restate their proofs.

We start by bounding $\mathcal{E}_{\Delta_C}$. We follow the proof of [18]. Let us introduce a random variable $X_{\mathbf{i}}^{\sigma}$ that follows a binomial distribution of m trials with a probability of success $\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}}, \sigma))$, denoted $B(m, \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}}, \sigma)))$. We use Lemma 12 with $g(\cdot) = e^{m\Delta_C(\cdot, \mathcal{L}_{\mathcal{D}}(h))}$.

$$
\begin{aligned}
\mathcal{E}_{\Delta_C}(\mathbf{i}, \sigma) &= \mathop{\mathbb{E}}_{T_{\mathbf{i}} \sim \mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c} \sim \mathcal{D}^{n-|\mathbf{i}|}} e^{(n-|\mathbf{i}|)\Delta_C\left(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)} \\
&\le \mathop{\mathbb{E}}_{T_{\mathbf{i}} \sim \mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{X_{\mathbf{i}}^{\sigma} \sim B(m, \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma)))} e^{(n-|\mathbf{i}|)\Delta_C\left(\frac{1}{n-|\mathbf{i}|} X_{\mathbf{i}}^{\sigma}, \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)} \\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}} \sim \mathcal{D}^{|\mathbf{i}|}} \sum_{k=0}^{n-|\mathbf{i}|} \mathop{\mathbb{P}}_{X_{\mathbf{i}}^{\sigma} \sim B(m, \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma)))} (X_{\mathbf{i}}^{\sigma} = k) e^{(n-|\mathbf{i}|)\Delta_C\left(\frac{k}{n-|\mathbf{i}|}, \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)} \\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}} \sim \mathcal{D}^{|\mathbf{i}|}} \sum_{k=0}^{n-|\mathbf{i}|} \binom{n - |\mathbf{i}|}{k} (\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}}, \sigma)))^k (1 - \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}}, \sigma)))^{n-|\mathbf{i}|-k} e^{(n-|\mathbf{i}|)\Delta_C\left(\frac{k}{n-|\mathbf{i}|}, \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)}
\end{aligned}
$$

$$\begin{aligned}
&\leq \mathop{\mathbb{E}}_{T_{\mathbf{i}} \sim \mathcal{D}^{|\mathbf{i}|}} \sup_{r \in [0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^k (1-r)^{n-|\mathbf{i}|-k} e^{(n-|\mathbf{i}|)\Delta_C\left(\frac{k}{n-|\mathbf{i}|}, r\right)} \right] \\
&= \sup_{r \in [0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^k (1-r)^{n-|\mathbf{i}|-k} e^{(n-|\mathbf{i}|)\Delta_C\left(\frac{k}{n-|\mathbf{i}|}, r\right)} \right] \\
&= \sup_{r \in [0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^k (1-r)^{n-|\mathbf{i}|-k} \frac{e^{-Ck}}{[1-(1-e^{-C})r]^{n-|\mathbf{i}|}} \right] \\
&= \sup_{r \in [0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} \left(re^{-C}\right)^k (1-r)^{n-|\mathbf{i}|-k} \frac{1}{[1-(1-e^{-C})r]^{n-|\mathbf{i}|}} \right] \\
&= \sup_{r \in [0,1]} \left[\frac{[re^{-C} + (1-r)]^{n-|\mathbf{i}|}}{[1-(1-e^{-C})r]^{n-|\mathbf{i}|}} \right] = \sup_{r \in [0,1]} [1] = 1.
\end{aligned}$$

where the last lign is derived using binomial theorem.

Now, we rearrange the terms:

$$\begin{aligned}
\Delta\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(R(\mathbf{i},\sigma)), \mathcal{L}_D(R(\mathbf{i},\sigma))\Big) &\leq \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right] \\
-\ln\big(1-\mathcal{L}_D(R(\mathbf{i},\sigma))(1-e^{-C})\big) - C\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(R(\mathbf{i},\sigma)) &\leq \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right] \\
\ln\big(1-\mathcal{L}_D(R(\mathbf{i},\sigma))(1-e^{-C})\big) &\geq -C\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(R(\mathbf{i},\sigma)) - \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right] \\
1-\mathcal{L}_D(R(\mathbf{i},\sigma))(1-e^{-C}) &\geq \exp\left(-C\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(R(\mathbf{i},\sigma)) - \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right]\right) \\
\mathcal{L}_D(R(\mathbf{i},\sigma))(1-e^{-C}) &\leq 1-\exp\left(-C\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(R(\mathbf{i},\sigma)) - \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right]\right) \\
\mathcal{L}_D(R(\mathbf{i},\sigma)) &\leq \frac{1}{1-e^{-C}}\left[1-\exp\left(-C\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(R(\mathbf{i},\sigma)) - \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right]\right)\right]
\end{aligned}$$

□

Corollary 6. *In the setting of Theorem 3, for a loss function* $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to [0,1]$*, with probability at least* $1-\delta$ *over the draw of* $S \sim \mathcal{D}^n$*, we have*

$$\forall \mathbf{i} \in \mathcal{P}(n), \sigma \in M(\mathbf{i}) : \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) \leq \mathrm{kl}^{-1}\Big(\widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)), \epsilon_{\mathrm{kl}}(\mathbf{i},\sigma,\delta)\Big)$$

with $\mathrm{kl}^{-1}(q,\epsilon) = \arg\sup_{0 \leq p \leq 1}\{\mathrm{kl}(q,p) \leq \epsilon\}$ *and*

$$\epsilon_{\mathrm{kl}}(\mathbf{i},\sigma,\delta) = \frac{1}{n-|\mathbf{i}|}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{2\sqrt{n-|\mathbf{i}|}}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right].$$

Proof. To prove this corollary, we need to bound $\mathcal{E}_{\mathrm{kl}}$. This was first proven by [30, 50] and then improved by [39]. We restate the proof for completeness.

We use Lemma 12 with $g(\cdot) = e^{m\mathrm{kl}(\cdot, \mathcal{L}_{\mathcal{D}}(h))}$.

$$\mathcal{E}_{\mathrm{kl}}(\mathbf{i},\sigma) = \mathop{\mathbb{E}}_{T_{\mathbf{i}} \sim \mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c} \sim \mathcal{D}^{n-|\mathbf{i}|}} e^{(n-|\mathbf{i}|)\mathrm{kl}\left(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)}$$

$$
\begin{aligned}
&\leq \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{X_{\mathbf{i}}^{\sigma}\sim B(m,\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma)))} e^{(n-|\mathbf{i}|)\mathrm{kl}\left(\frac{1}{n-|\mathbf{i}|}X_{\mathbf{i}}^{\sigma},\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)}\\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \sum_{k=0}^{n-|\mathbf{i}|} \mathop{\mathbb{P}}_{X_{\mathbf{i}}^{\sigma}\sim B(m,\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma)))}(X_{\mathbf{i}}^{\sigma}=k) e^{(n-|\mathbf{i}|)\mathrm{kl}\left(\frac{k}{n-|\mathbf{i}|},\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)}\\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma)))^{k}(1-\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma)))^{n-|\mathbf{i}|-k} e^{(n-|\mathbf{i}|)\mathrm{kl}\left(\frac{k}{n-|\mathbf{i}|},\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)}\\
&\leq \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \sup_{r\in[0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^{k}(1-r)^{n-|\mathbf{i}|-k} e^{(n-|\mathbf{i}|)\mathrm{kl}\left(\frac{k}{n-|\mathbf{i}|},r\right)}\right]\\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^{k}(1-r)^{n-|\mathbf{i}|-k} e^{(n-|\mathbf{i}|)\mathrm{kl}\left(\frac{k}{n-|\mathbf{i}|},r\right)}\right]\\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^{k}(1-r)^{n-|\mathbf{i}|-k} \times e^{(n-|\mathbf{i}|)\left(\frac{k}{n-|\mathbf{i}|}\ln\left(\frac{k}{n-|\mathbf{i}|}\cdot\frac{1}{r}\right)+(1-\frac{k}{n-|\mathbf{i}|})\ln\left((1-\frac{k}{n-|\mathbf{i}|})\cdot\frac{1}{1-r}\right)\right)}\right]\\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^{k}(1-r)^{n-|\mathbf{i}|-k} \times e^{k\ln\left(\frac{k}{n-|\mathbf{i}|}\cdot\frac{1}{r}\right)+(n-|\mathbf{i}|-k)\ln\left((1-\frac{k}{n-|\mathbf{i}|})\cdot\frac{1}{1-r}\right)}\right]\\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^{k}(1-r)^{n-|\mathbf{i}|-k} \times e^{\ln\left(\frac{k}{n-|\mathbf{i}|}\right)^{k}+\ln\left(\frac{1}{r}\right)^{k}+\ln\left(1-\frac{k}{n-|\mathbf{i}|}\right)^{n-|\mathbf{i}|-k}+\ln\left(\frac{1}{1-r}\right)^{n-|\mathbf{i}|-k}}\right]\\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} (r)^{k}(1-r)^{n-|\mathbf{i}|-k} \times \frac{1}{(r)^{k}(1-r)^{n-|\mathbf{i}|-k}}\left(\frac{k}{n-|\mathbf{i}|}\right)^{k}\left(1-\frac{k}{n-|\mathbf{i}|}\right)^{n-|\mathbf{i}|-k}\right]\\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{n-|\mathbf{i}|} \binom{n-|\mathbf{i}|}{k} \left(\frac{k}{n-|\mathbf{i}|}\right)^{k}\left(1-\frac{k}{n-|\mathbf{i}|}\right)^{n-|\mathbf{i}|-k}\right]\\
&\leq e^{\frac{1}{12(n-|\mathbf{i}|)}}\sqrt{\frac{\pi(n-|\mathbf{i}|)}{2}}+2\\
&\leq 2\sqrt{n-|\mathbf{i}|}.
\end{aligned}
$$

The last two inequalities were proven by [39]. The last inequality holds only for $n-|\mathbf{i}|\geq 8$, but it can be verified that $\mathcal{E}_{\mathrm{kl}}(\mathbf{i},\sigma)\leq 2\sqrt{n-|\mathbf{i}|}$ holds for $n-|\mathbf{i}|\geq 1$. □

Corollary 7. *In the setting of Theorem 3, for any $\lambda > 0$, with a ς^2-sub-Gaussian loss function $\ell : \mathcal{H}\times\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$, with probability at least $1-\delta$ over the draw of $S\sim\mathcal{D}^n$, we have*

$$
\begin{aligned}
&\forall \mathbf{i}\in\mathcal{P}(n), \sigma\in M(\mathbf{i}):\\
&\quad \mathcal{L}_{\mathcal{D}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) \leq \widehat{\mathcal{L}}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\sigma)) + \frac{\lambda\varsigma^2}{2} + \frac{1}{\lambda(n-|\mathbf{i}|)}\left[\log\binom{n}{|\mathbf{i}|} + \log\left(\frac{1}{\zeta(|\mathbf{i}|)P_{M(\mathbf{i})}(\sigma)\delta}\right)\right].
\end{aligned}
$$

Proof. We assume that the loss ℓ is ς^2-sub-Gaussian, which is defined as:

$$
\mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \exp\left[\lambda\left(\ell(h,\boldsymbol{x},y) - \mathop{\mathbb{E}}_{(\boldsymbol{x}',y')\sim\mathcal{D}}\ell(h,\boldsymbol{x}',y')\right)\right] \leq \exp\left(\frac{\lambda^2\varsigma^2}{2}\right).
$$

Then, we have

$$
\mathcal{E}_{\Delta_\lambda}(\mathbf{i},\sigma) = \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} \exp\left[(n-|\mathbf{i}|)\Delta_\lambda\left(\widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma)), \mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\right)\right]
$$

$$
\begin{aligned}
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} \exp\left[(n-|\mathbf{i}|)\lambda\Big(\mathcal{L}_{\mathcal{D}}(\mathcal{R}(T_{\mathbf{i}},\sigma)) - \widehat{\mathcal{L}}_{T_{\mathbf{i}^c}}(\mathcal{R}(T_{\mathbf{i}},\sigma))\Big)\right] \\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} \exp\left[(n-|\mathbf{i}|)\lambda\left(\mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x},y) - \frac{1}{n-|\mathbf{i}|}\sum_{i=1}^{n-|\mathbf{i}|} \ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x}_i,y_i)\right)\right] \\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} \exp\left[\lambda\sum_{i=1}^{n-|\mathbf{i}|}\left(\mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x},y) - \ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x}_i,y_i)\right)\right] \\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} \exp\left[-\lambda\sum_{i=1}^{n-|\mathbf{i}|}\left(\ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x}_i,y_i) - \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x},y)\right)\right] \\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \mathop{\mathbb{E}}_{T_{\mathbf{i}^c}\sim\mathcal{D}^{n-|\mathbf{i}|}} \prod_{i=1}^{n-|\mathbf{i}|} \exp\left[-\lambda\Big(\ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x}_i,y_i) - \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x},y)\Big)\right] \\
&= \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \prod_{i=1}^{n-|\mathbf{i}|} \mathop{\mathbb{E}}_{(\boldsymbol{x}_i,y_i)\sim\mathcal{D}} \exp\left[-\lambda\Big(\ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x}_i,y_i) - \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \ell(\mathcal{R}(T_{\mathbf{i}},\sigma),\boldsymbol{x},y)\Big)\right] \qquad (10) \\
&\le \mathop{\mathbb{E}}_{T_{\mathbf{i}}\sim\mathcal{D}^{|\mathbf{i}|}} \prod_{i=1}^{n-|\mathbf{i}|} \exp\left(\frac{\lambda^2\varsigma^2}{2}\right) \qquad (11) \\
&= \exp\left(\frac{(n-|\mathbf{i}|)\lambda^2\varsigma^2}{2}\right).
\end{aligned}
$$

In Equation 10, we use the *i.i.d.* assumption. In Equation 11, we use the ς^2-sub-Gaussian assumption.

We replace the comparator function in Theorem 3 and bound the cumulant generating function $\mathcal{E}_{\Delta_\lambda}$ to finish the proof. □

C.3 Behavior with zero error

Theorem 8. *In the consistent case, i.e. when the training error is zero, Corollary 4 is arbitrarily close to the binomial tail inversion of Theorem 1. Moreover, Corollary 6 is a tight upper bound up to a constant $K(m,\delta)$ that decreases for m large enough and tends to 0 when m tends to ∞. Indeed,*

$$
\begin{aligned}
\overline{\mathrm{Bin}}(0,m,\delta) &= 1-\exp\left(\frac{-1}{m}\ln\frac{1}{\delta}\right) && (3)\\
&= \lim_{C\to\infty}\frac{1}{1-e^{-C}}\left[1-\exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right)\right] && (4)\\
&= \inf_{C>0}\frac{1}{1-e^{-C}}\left[1-\exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right)\right] && (5)\\
&= \mathrm{kl}^{-1}\left(0,\frac{1}{m}\ln\frac{1}{\delta}\right) && (6)\\
&\le \mathrm{kl}^{-1}\left(0,\frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right). && (7)\\
&= \mathrm{kl}^{-1}\left(0,\frac{1}{m}\ln\frac{1}{\delta}\right)+K(m,\delta). && (8)
\end{aligned}
$$

Before proving the result, we remind the reader of the definition of the binomial tail:

$$
\mathrm{Bin}(k,m,r)=\sum_{i=0}^{k}\binom{m}{i}r^i(1-r)^{m-i}
$$

and the binomial tail inversion :

$$\overline{\text{Bin}}(k, m, \delta) = \underset{r \in [0,1]}{\arg\sup} \left\{ \text{Bin}(k, m, r) \geq \delta \right\}$$

Proof of Eq. (3).

$$\begin{aligned}
\text{Bin}(0, m, r) &= \sum_{i=0}^{0} \binom{m}{i} r^i (1-r)^{m-i} \\
&= \binom{m}{0} r^0 (1-r)^{m-0} \\
&= 1 \cdot 1 \cdot (1-r)^m \\
&= (1-r)^m.
\end{aligned}$$

Thus, we have :

$$\begin{aligned}
\overline{\text{Bin}}(0, m, \delta) &= \underset{r \in [0,1]}{\arg\sup} \left\{ \text{Bin}(0, m, r) \geq \delta \right\} \\
&= \underset{r \in [0,1]}{\arg\sup} \left\{ (1-r)^m \geq \delta \right\} \\
&= \underset{r \in [0,1]}{\arg\sup} \left\{ 1 - r \geq \delta^{\frac{1}{m}} \right\} \\
&= \underset{r \in [0,1]}{\arg\sup} \left\{ 1 - \delta^{\frac{1}{m}} \geq r \right\} \\
&= 1 - \delta^{\frac{1}{m}} \\
&= 1 - \exp\left(\ln\left(\delta^{\frac{1}{m}} \right) \right) \\
&= 1 - \exp\left(-\frac{1}{m} \ln\left(\frac{1}{\delta} \right) \right)
\end{aligned}$$

□

Proof of Eq. (4). We want to show that :

$$1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) = \lim_{C \to \infty} \frac{1}{1 - e^{-C}} \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right]$$

We simply take the limit.

$$\begin{aligned}
\lim_{C \to \infty} \frac{1}{1 - e^{-C}} \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right] &= \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right] \lim_{C \to \infty} \frac{1}{1 - e^{-C}} \\
&= \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right] \cdot 1 \\
&= 1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right)
\end{aligned}$$

□

Proof of Eq. (5). We want to prove that :

$$\lim_{C \to \infty} \frac{1}{1 - e^{-C}} \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right] = \inf_{C > 0} \frac{1}{1 - e^{-C}} \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right].$$

First of all, we know that :

$$\inf_{C > 0} \frac{1}{1 - e^{-C}} \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right] = \left[1 - \exp\left(-\frac{1}{m} \ln \frac{1}{\delta} \right) \right] \inf_{C > 0} \frac{1}{1 - e^{-C}}$$

We show that $f(C) = \frac{1}{1-e^{-C}}$ is decreasing on $[0, \infty)$ and has a minimum at $C = \infty$.

We take its derivative :

$$\begin{aligned}
f'(C) &= \frac{\mathrm{d}}{\mathrm{d}C}\frac{1}{1-e^{-C}} = \frac{\mathrm{d}}{\mathrm{d}C}\left(1-e^{-C}\right)^{-1} \\
&= \frac{-1}{\left(1-e^{-C}\right)^2}\frac{\mathrm{d}}{\mathrm{d}C}\left(1-e^{-C}\right) \\
&= \frac{-1}{\left(1-e^{-C}\right)^2}e^{-C} \\
&= (-1)\frac{e^{-C}}{\left(1-e^{-C}\right)^2}
\end{aligned}$$

The derivative of f is always negative, so f is decreasing. Thus, it will have no minimum on $\mathbb{R}_{>0}$. Indeed, if we try to find a value of $C \in \mathbb{R}_{>0}$ such that $f'(C) = 0$, we have

$$\begin{aligned}
(-1)\frac{e^{-C}}{\left(1-e^{-C}\right)^2} &= 0 \\
\iff e^{-C} &= 0.
\end{aligned}$$

There is no $C \in \mathbb{R}_{>0}$ such that $e^{-C} = 0$, however we know that $\lim_{C\to\infty} e^{-C} = 0$. This finishes the proof. □

Proof of Eq. (6). We want to show :

$$1 - \exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right) = \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right)$$

The function $\mathrm{kl}(0,p)$ is monotonically increasing and $\mathrm{kl}(0,1) = \infty$. Thus, there exists a value $p^* = \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right)$ such that $\mathrm{kl}(0,p^*) = \frac{1}{m}\ln\frac{1}{\delta}$.

$$\begin{aligned}
& p^* = \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right) \\
\iff & \mathrm{kl}(0,p^*) = \frac{1}{m}\ln\frac{1}{\delta} \\
\iff & 0\ln\frac{0}{p^*} + (1)\ln\frac{1}{1-p^*} = \frac{1}{m}\ln\frac{1}{\delta} \qquad (12) \\
\iff & \ln\frac{1}{1-p^*} = \frac{1}{m}\ln\frac{1}{\delta} \\
\iff & \frac{1}{1-p^*} = \exp\left(\frac{1}{m}\ln\frac{1}{\delta}\right) \\
\iff & p^* = 1 - \exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right)
\end{aligned}$$

In Eq. (12), we use the convention of the PAC-Bayes theory that $0\ln 0 = 0$. This is used to define the kl when $q \in \{0,1\}$ or $p \in \{0,1\}$. □

Proof of Eq. (7). We want to prove that

$$\mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right) \leq \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right).$$

The function $\mathrm{kl}(0,p)$ is monotonically increasing and $\mathrm{kl}(0,1) = \infty$. Thus, there exists a value $p^* = \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right)$ such that $\mathrm{kl}(0,p^*) = \frac{1}{m}\ln\frac{1}{\delta}$. Moreover, there exists a value

$p^\dagger = \mathrm{kl}^{-1}\Big(0, \frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\Big)$ such that $\mathrm{kl}(0, p^\dagger) = \frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}$. As $\mathrm{kl}(0, p)$ is monotonically increasing and

$$\mathrm{kl}(0, p^*) = \frac{1}{m}\ln\frac{1}{\delta} \leq \frac{1}{m}\ln\frac{2\sqrt{m}}{\delta} = \mathrm{kl}(0, p^\dagger),$$

then $p^* \leq p^\dagger$. □

Proof of Eq. (8). We want to prove that

$$\mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right) = \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right) + K(m,\delta)$$

and that $K(m,\delta)$ decreases for m large enough and tends to 0 when $m \to \infty$. To do so, we will use Eq. (6), restated here for ease of reading

$$1 - \exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right) = \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right).$$

We start by defining the constant as the gap between the two terms.

$$\begin{aligned}
&\mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right) - \mathrm{kl}^{-1}\left(0, \frac{1}{m}\ln\frac{1}{\delta}\right)\\
=&\left[1 - \exp\left(-\frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right)\right] - \left[1 - \exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right)\right]\\
=&\exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right) - \exp\left(-\frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right)\\
=:&K(m,\delta)
\end{aligned}$$

We now prove that this constant is decreasing for m large enough. To do so, we start by computing the derivative of K, for $m > 0$.

We have :

$$\begin{aligned}
\frac{\partial K(m,\delta)}{\partial m} &= \frac{1}{2}\frac{\exp\left(-\frac{1}{m}\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right) - 2\ln(\delta)\exp\left(\frac{1}{m}\ln(\delta)\right) - 2\ln\left(\frac{2\sqrt{m}}{\delta}\right)\exp\left(-\frac{1}{m}\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right)}{m^2}\\
&= \frac{1}{2}\frac{\left(1 - 2\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right)\exp\left(-\frac{1}{m}\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right) + 2\ln\left(\frac{1}{\delta}\right)\exp\left(-\frac{1}{m}\ln\left(\frac{1}{\delta}\right)\right)}{m^2}
\end{aligned}$$

The first term is always negative, as we have

$$1 - 2\ln\left(\frac{2\sqrt{m}}{\delta}\right) \leq 0 \iff \frac{\delta^2}{4}e \leq m$$

and $\frac{\delta^2}{4}e$ is always smaller than one. All the other terms will always be positive.

We focus only on the numerator, as the denominator will always be greater than 0. We wish to find the values of m such that the derivative of $K(m,\delta)$ is negative, showing that $K(m,\delta)$ is decreasing.

$$\begin{aligned}
&\left(1 - 2\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right)\exp\left(-\frac{1}{m}\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right) + 2\ln\left(\frac{1}{\delta}\right)\exp\left(-\frac{1}{m}\ln\left(\frac{1}{\delta}\right)\right) \leq 0\\
\iff &\left(1 - 2\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right)\exp\left(-\frac{1}{m}\ln\left(2\sqrt{m}\right)\right)\exp\left(-\frac{1}{m}\ln\left(\frac{1}{\delta}\right)\right) + 2\ln\left(\frac{1}{\delta}\right)\exp\left(-\frac{1}{m}\ln\left(\frac{1}{\delta}\right)\right) \leq 0\\
\iff &\left(1 - 2\ln\left(\frac{2\sqrt{m}}{\delta}\right)\right)\exp\left(-\frac{1}{m}\ln\left(2\sqrt{m}\right)\right) + 2\ln\left(\frac{1}{\delta}\right) \leq 0\\
\iff &2\ln\left(\frac{1}{\delta}\right) \leq \left(2\ln\left(\frac{2\sqrt{m}}{\delta}\right) - 1\right)\exp\left(-\frac{1}{m}\ln\left(2\sqrt{m}\right)\right)\\
\iff &2\ln\left(\frac{1}{\delta}\right) \leq \left(2\ln\left(\frac{1}{\delta}\right) + \ln\left(2\sqrt{m}\right) - 1\right)\exp\left(-\frac{1}{m}\ln\left(2\sqrt{m}\right)\right)
\end{aligned}$$

As the exponential term tends to one when m tends to ∞, for m large enough, the inequality will always hold and $K(m, \delta)$ will be decreasing. To the best of our knowledge, there exists no analytical solution to this inequality. However, we can easily compute m^* the solution of $\frac{\partial K(m,\delta)}{\partial m} = 0$. Then, we know that for any $m \geq m^*$, $K(m, \delta)$ will be decreasing.[2]

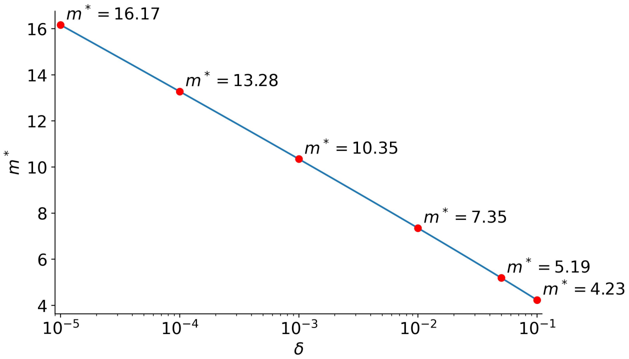


Figure 3: Illustration of the behavior of the solution m^* for different values of δ.

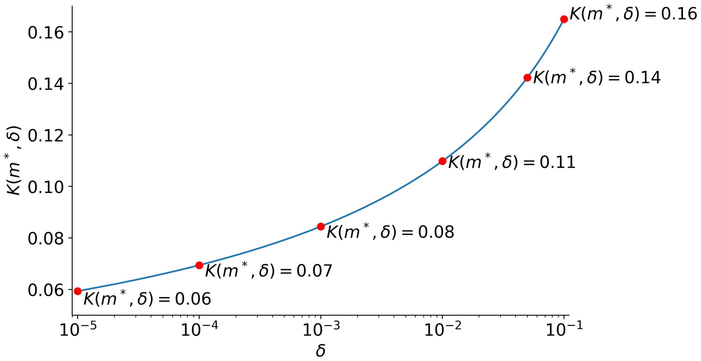


Figure 4: Illustration of the behavior of $K(m^*, \delta)$ for different values of δ.

Finally, we show that this constant $K(m, \delta)$ tends to 0 when m tends to ∞, as such

$$\begin{aligned}
\lim_{n\to\infty} K(n, \delta) &= \lim_{n\to\infty} \exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right) - \exp\left(-\frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right) \\
&= \lim_{n\to\infty} \exp\left(-\frac{1}{m}\ln\frac{1}{\delta}\right) - \lim_{n\to\infty} \exp\left(-\frac{1}{m}\ln\frac{2\sqrt{m}}{\delta}\right) \\
&= 1 - 1 = 0.
\end{aligned}$$

□

To provide an intuition of the magnitude of $K(m, \delta)$, hence of the tightness of the upper bound of the binomial tail inversion, in Fig. 5 we report the values of $K(m, \delta)$ for different sizes of datasets m and the commonly chosen $\delta = 0.01$. In particular, we highlight the values corresponding to the training set sizes used in our experiments, as reported in Table 6. Note however, that when computing the bounds in practice, the gap will be $K(m - |\mathbf{i}|, \delta)$ as we will need to consider the size of the compression set.

[2] In Fig. 3, we present the value of m^* for multiple values of δ. In Fig. 4, we present the maximum value of $K(m, \delta)$, which is achieved at m^*, for multiple values of δ.

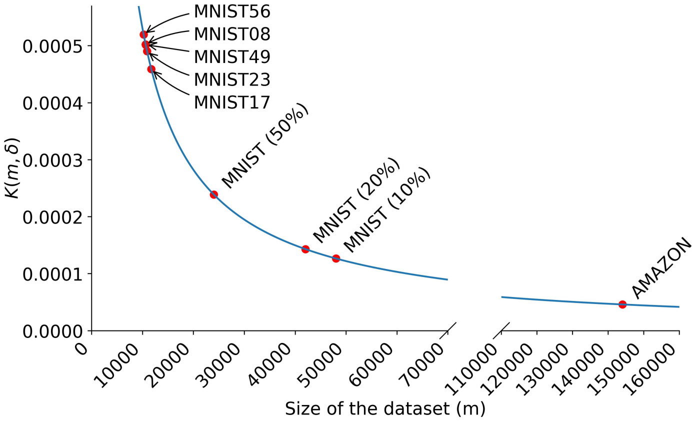


Figure 5: Illustration of the behavior of $K(m, \delta)$ for different sizes of datasets, when $\delta = 0.01$. We highlight the values corresponding to the datasets used for classification.