# Shortest Program Interpolation Learning

**Naren Sarayu Manoj** NSM@TTIC.EDU
*Toyota Technological Institute Chicago*

**Nathan Srebro** NATI@TTIC.EDU
*Toyota Technological Institute Chicago*

**Editors:** Gergely Neu and Lorenzo Rosasco

## Abstract

We prove that the Minimum Program Length learning rule exhibits tempered overfitting. We obtain tempered agnostic finite sample learning guarantees and characterize the asymptotic behavior in the presence of random label noise.

## 1. Introduction

We consider the learning rule MPL (minimum program length) which selects the predictor describable with minimal program length, in some fixed Turing-complete programming language, that fits the training set perfectly. That is, given training examples $x_1, \ldots, x_m$ with binary labels $y_1, \ldots, y_m \in \{0, 1\}$, MPL returns the shortest program $\mathsf{h}(\cdot)$ such that for every training example $x_i$, $\mathsf{h}(x_i)$ outputs its training label, $y_i$. Such minimum description length learning is well understood in the realizable setting [BEHW87] – if there exists some $\mathsf{h}^\star$ that is perfect on the source distribution, i.e. with zero population loss $L(\mathsf{h}^\star) = 0$, then $O(|\mathsf{h}^\star| / \varepsilon)$ samples are enough for MPL to have (expected) population loss at most $\varepsilon$, where $|\mathsf{h}^\star|$ is the description, or program, length of $\mathsf{h}^\star$.

However, to handle noisy situations, or compete with a short program $\mathsf{h}$ that might not be perfect, the standard wisdom is to balance training error against description length, e.g. employing the Structural Risk Minimization (SRM) principle. By minimizing the right combination of training error and description length (or a combination tuned through validation), such an SRM predictor can compete with any predictor $\mathsf{h}$, and using a training set of size $m$ has expected error at most [e.g. SB14][1]

$$\inf_{\mathsf{h}} \left( L(\mathsf{h}) + O\left( \frac{|\mathsf{h}|}{m} + \sqrt{L(\mathsf{h}) \cdot \frac{|\mathsf{h}|}{m}} \right) \right). \tag{1.1}$$

But following recent interest in benign overfitting and interpolation learning of noisy data [e.g. BHM18; BRT19; NDR20; BLLT20; MRSY20; HMRT; MNSBHS21; CL21, and many others], we ask: *what happens if we insist on interpolating (i.e. obtaining zero training error) and using the interpolating* MPL *rule?* Does MPL overfit benignly? Does it still enjoy the same guarantee (1.1) as SRM? Is it consistent like SRM, i.e., does it converge to the Bayes optimal predictor (as long as the Bayes optimal predictor has finite description)? Or is overfitting by MPL catastrophic, possibly

1. Shalev-Shwartz and Ben-David analyze SRM using Hoeffding's inequality, and thus present a guarantee with excess error $O(\sqrt{|\mathsf{h}| / m})$, while the tighter guarantee (1.1) is based on Bernstein's inequality. They call this application of the SRM principle "MDL", and thus use "MDL" to refer to the learning rule we refer to as "SRM". Others, such as Grünwald and Langford [GL07] refer to what we call SRM as "Occam Razor's Bound" (ORB), citing Blumer, Ehrenfeucht, Haussler, and Warmuth [BEHW87], although Blumer *et al* consider only the realizable case and thus the interpolating MDL rule.

yielding worthless predictions? Or perhaps tempered [as defined in MSAPBN22] with error worse than the optimally balanced SRM, but still better than random guessing? If so, can we bound the error of the interpolating MPL compared to the optimally balanced SRM? How much worse can it be compared to the SRM guarantee (1.1) ?

We show that MPL overfitting is not benign, with asymptotic error that could be worse than SRM. But we can bound this error away from $0.5$, as a simple fixed function of the Bayes error, depicted in Figure 1. With random label noise, we obtain a tight and precise characterization of the asymptotic error. Furthermore, we obtain an agnostic finite sample guarantee, which holds for any source distribution, without any realizability or specification assumptions, and tells us how well we compete with any competitor hypothesis (not necessarily the Bayes optimal). This contrasts with much of the existing work on benign overfitting which is distribution-specific, e.g. making Gaussianity assumptions on the data, and often assuming the model is well specified.

Our analysis essentially follows a uniform convergence approach, and decouples the analyses of the program length of MPL from that of the generalization error for short programs. In Section 3 we bound the minimum program length $|\mathsf{MPL}(S)|$ by proving an upper bound on the program needed to interpolate a noisy training set. Then in Section 4 we bound the expected error of any learning rule returning short programs in terms of the length of the program. Our learning guarantees, stated in Section 2, then follow immediately by combining the two.

**Minimum Description Length terminology and Rissanen's MDL model selection criterion** The minimum program (or description) length interpolator we study here should not be confused with the Minimum Description Length model selection criterion [Ris78], where a two-part code is typically considered, the first part describing a *probabilistic* model, and the second describing the data using the model. Grünwald and Langford [GL07] suggest using this model selection criterion also in a supervised learning setting, as a way of balancing the description length $|\mathsf{h}|$ (or almost equivalently, the prior probability of $\mathsf{h}$) with its empirical error $L_S(\mathsf{h})$. The resulting learning rule, which we refer to as $\mathsf{MDL}_2$, can be viewed as an alternative to SRM (which Grünwald and Langford refer to as "Occam's Razor Bound" (ORB)), yielding a different balance between $|\mathsf{h}|$ and $L_S(\mathsf{h})$, but generally not interpolating (i.e. not insisting on $L_S(\mathsf{h}) = 0$). Grünwald and Langford study $\mathsf{MDL}_2$ in the same agnostic supervised learning setting we employ, and show that it is inferior to SRM, does not yield (1.1), and in fact can be asymptotically inconsistent. However, their asymptotic inconsistency result is very different from ours in several important ways: **First** as we are interested in understanding the effect of overfitting and interpolation learning, we insist on perfect fit: we view $\mathsf{h} : x \mapsto y$ as defining a deterministic mapping and our "description" can be viewed as a one-part code. The interpolating learning rule we study is thus very different from the "balanced" $\mathsf{MDL}_2$ they study. **Second**, they obtain asymptotic inconsistency only in a non-well-specified setting. This is essential since $\mathsf{MDL}_2$ *is* consistent in a well specified setting. In contrast, we show the interpolating MPL is inconsistent *even with random classification noise*. **Third**, they show inconsistency for a very specific and artificial prior, corresponding to a very restrictive description language that is nothing like a Turing-complete language. It is easy to see that minimum description length (or max prior) interpolation has no chance of working with such a prior. Rather, we are interested in interpolation with short programs, i.e. with a Turing-complete description language, and show both lack of consistency in this setting, and more interestingly an upper bound (i.e. that interpolation is "tempered") which rests crucially on Turing completeness. In Appendix B we comment further on how our analysis relates to that of Grünwald and Langford.

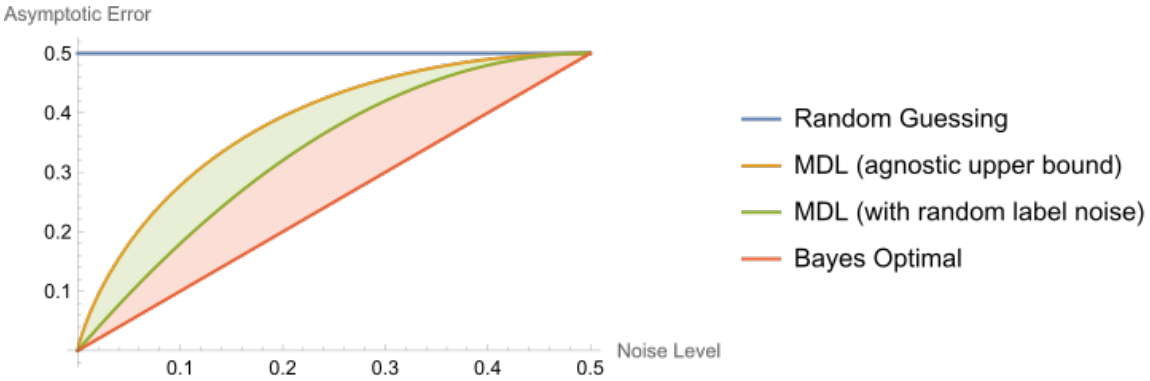


Figure 1: Behavior of interpolating MDL as a function of the noise level. Top curve: The function $\ell_{\mathsf{ag}}(L^\star)$, which provides an agnostic upper bound on the error of MPL, with a finite sample gurantee that approaches this curve; Contrast with the middle curve: the function $\ell_{\mathsf{samp}}(L^\star)$, which is the exact asymptotic error of MPL under random label noise.

**Notation** We write Bernoulli random variables with parameter $\alpha$ as $\mathsf{Ber}(\alpha)$. We use $H(X)$ to denote the entropy of random variable. We also write $H(\alpha)$ to denote the entropy of a $\mathsf{Ber}(\alpha)$-random variable. The Radon-Nikodym derivative between two distributions $p$ and $q$ is denoted $dp/dq$, and one can informally think of $dp(\cdot)$ as the probability density or mass function. We measure information in bits, and $\log$ is always base 2. The operation $\oplus$ denotes the XOR of two bits. For two random variables $A$ and $B$, we write $A \perp B$ to mean that $A$ and $B$ are independent.

## 2. Formal Setup and Main Results

We consider learning based on $m$ i.i.d. samples $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \sim \mathcal{D}^m$ from a source distribution $\mathcal{D}(X, Y)$ over bit-strings $X$ and binary labels $Y \in \{0, 1\}$. A learning rule is a mapping $A : S \mapsto \mathsf{h}$ from training sets to predictors $\mathsf{h} : x \mapsto y$. To formalize the notion of program length, we can think of the predictors $\mathsf{h}$ as programs in some prefix-unambiguous Turing complete programming language, and we use $|\mathsf{h}|$ to denote program length in bits. We denote the training error as $L_S(\mathsf{h}) = \frac{1}{m}\mathbb{1}\{\mathsf{h}(x_i) \neq y_i\}$. We say $\mathsf{h}$ interpolates $S$ if $L_S(\mathsf{h}) = 0$ and that $A$ is an interpolating rule if $L_S(A(S)) = 0$ almost surely. With this notation, we have that $\mathsf{MPL}(S) = \arg\min_{L_S(\mathsf{h})=0} |\mathsf{h}|$. We denote the population error by $L(\mathsf{h}) = \Pr_{(X,Y)\sim\mathcal{D}}[\mathbb{1}\{\mathsf{h}(x_i) \neq y_i\}]$ and we use the same notation for the expected error of a learning rule: $L(A) = \mathbb{E}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S))]$.

In order to discuss interpolation learning, we must ensure it is always indeed possible to interpolate. This is the case if we never encounter the same instance $x$ with different labels, i.e.

$$\Pr\left[X = X' \wedge Y \neq Y'\right] = 0 \quad \text{for} \quad (X, Y), (X', Y') \sim \text{ i.i.d. } \mathcal{D}. \tag{2.1}$$

We will thus always assume (2.1). This is the case when $Y$ is a deterministic function of $X$. But we are particularly interested in noisy settings, in which case (2.1) holds, e.g., if $\mathcal{D}$ is non-atomic, i.e. $\Pr[X = X'] = 0$. In order to discuss non-atomic distributions over bit-strings, we will allow $X \in \{0, 1\}^{\mathbb{N}}$ to be an infinite[2] sequence of bits (e.g. the binary digits of a real number). The

2. To capture also finite bit strings $x \in \{0, 1\}^*$, we can think of padding $x$ with an infinite number of zeros.

programs[3] we learn will only be able to access a finite number of bits of $x$, and it will be useful for us to consider prefixes $x[:b]$ consisting of the first $b$ bits of $x$. Although we consider infinite bit sequences, we will need to bound how far we need to read in order to distinguish between instances. We formalize this notion through the following definition:

**Definition 2.1** *The* **disambiguation prefix length** $b(S)$ *of a sample* $S$ *is the minimal* $b$ *such that for all* $(x_i, y_i), (x_j, y_j) \in S$*, if* $x_i[:b] = x_j[:b]$*, then* $(x_i, y_i) = (x_j, y_j)$*. The* **quenched disambiguation prefix length** $\bar{b}(m)$ *of a distribution* $\mathcal{D}$ *for sample size* $m$ *is given by*

$$\log \bar{b}(m) \coloneqq \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[\log b(S)] \leq \log\left(\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[b(S)]\right)$$

With these definitions in hand, we are ready to state our main results.

**Theorem 2.2 (Agnostic)** *For any source distribution* $\mathcal{D}$ *with quenched disambiguation prefix length* $\bar{b}(m)$*, and any sample size* $m$*:*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L(\mathsf{MPL}(S))] \leq \inf_{\mathsf{h}}\left(\ell_{\mathsf{ag}}(L(\mathsf{h})) + O\left(\frac{|\mathsf{h}| + \log(m\cdot\bar{b}(m))}{m}\right)\right)$$

*where* $\ell_{\mathsf{ag}}(\alpha) \doteq 1 - 2^{-H(\alpha)} = 1 - \alpha^\alpha(1-\alpha)^{1-\alpha}$ *and* $\alpha < \ell_{\mathsf{ag}}(\alpha) < 0.5$ *for* $0 < \alpha < 0.5$*.*

For a "well specified" distribution, where the label noise is independent of $X$, we obtain a tighter and more precise guarantee:

**Theorem 2.3 (Random Label Noise)** *For any source distribution* $\mathcal{D}$ *where* $Y|X = \mathsf{h}^\star(X) \oplus \mathsf{Ber}(L^\star)$ *for some program* $\mathsf{h}^\star$ *and label noise* $L^\star$*, and any sample size* $m$*:*

$$\left|\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L(\mathsf{MPL}(S))] - \ell_{\mathsf{samp}}(L^\star)\right| \leq O\left(\frac{|\mathsf{h}^\star| + \log(m\cdot\bar{b}(m))}{m} + \sqrt{L^\star\cdot\frac{|\mathsf{h}^\star| + \log(m\cdot\bar{b}(m))}{m}}\right)$$

*where* $\ell_{\mathsf{samp}}(L^\star) \doteq 2L^\star(1-L^\star)$ *and* $L^\star < \ell_{\mathsf{samp}}(L^\star) < \ell_{\mathsf{ag}}(L^\star) < 0.5$ *for* $0 < L^\star < 0.5$*.*

Theorems 2.2 and 2.3 follow from plugging in Corollary 3.2 into Lemmas 4.1 and 4.2 (see Sections 3 and 4). The above Theorems hold for any finite number of samples and directly imply guarantees on the asymptotic error of MPL:

**Corollary 2.4** *For any source distribution* $\mathcal{D}$ *with quenched interpolation length* $\bar{b}(m) \leq 2^{o(m)}$ *and such that the Bayes predictor* $\mathsf{h}^\star(x) = \mathsf{Sign}(P(Y|x) - 0.5)$ *is computable, with Bayes error* $L^\star = L(\mathsf{h}^\star) < 0.5$ *then:*

$$\limsup_{m\to\infty}\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L(\mathsf{MPL}(S))] \leq \ell_{\mathsf{ag}}(L^\star) < 0.5$$

*And moreover, if the noise probability is independent of* $X$*, i.e.* $Y \perp X|\mathsf{h}^\star(X)$*, then more precisely:*

$$\lim_{m\to\infty}\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L(\mathsf{MPL}(S))] = \ell_{\mathsf{samp}}(L^\star) = 2L^\star(1-L^\star)$$

3. Formally, when discussing programs taking an infinite $x$ as input, we can think of a RAM computer which is allowed random access to bits of $x$, or a Turing Machine given access to $x$ on an infinite tape.

In Figure 1, we plot the general upper bound $\ell_{\mathsf{ag}}$ and the precise error for random label noise $\ell_{\mathsf{samp}}$. We see that even with random label noise, MPL overfitting is *not* benign, and MPL is not consistent. Nevertheless, regardless of the noise distribution, the asymptotic error can be non-trivially bounded as a function only of the Bayes error (or rather, the optimal error with a computable predictor), and without any dependence on any other property of the distribution, the predictor, or the noise.

**Bounding the quenched interpolation length.** We can bound $\bar{b}$ in terms of the min-entropy rate $H_{\min}(X[:b])/b$, where recall the min-entropy is defined as $H_{\min}(X) \doteq -\max_x \log P(X = x)$. For distributions uniform over $N$ outcomes, this is equal to $\log N$, which is also the Shannon entropy. But otherwise it can be smaller and captures the "worst case" randomness. The quenched interpolation length $\bar{b}(m)$ is roughly the length that ensures no collisions in a sample of size $m$, i.e., such that $m^2 \cdot \mathsf{Pr}\left[X[:b] = X'[:b]\right] \leq m^2 2^{-H_{\min}(X[:b])} \ll 1$, and so $H_{\min}(X[:\tilde{b}]) \approx O(\log m)$. If the bits of $X$ are uniform and independent, then the min-entropy rate is 1, $H_{\min}(X[:b]) = b$ and we have $\bar{b} = O(\log m)$. We can afford a much lower min-entropy rate. Any constant rate (e.g. when a small constant fraction of the bits are slightly bounded away from from being fixed conditioned on the previous bits), or arbitrary small polynomial rate $H_{\min}(X[:b]) = \Omega(b^\rho)$, still yields $\log \bar{b} = O(\log \log m)$. Even a logarithmically small min entropy rate, $H_{\min}(X[:b]) = \Omega(\log b)$ still ensures $\log \bar{b}(m) = O(\log m)$, and so we can ignore the dependence on $\bar{b}$ in our results. This happens, e.g., when differences between instances become increasingly sparse, with the entropy of the $i$th bit (conditioned on the previous bits) behaving like $1/i$. If the min-entropy rate is even lower, down to $H_{\min}(X[:b]) = \log \log b + \omega(1)$, we still have $\bar{b}(m) \leq 2^{o(m)}$ and the limits in Corollary 2.4 are still valid.

## 3. Constructing a Short Interpolating Predictor

Our goal in this section is to bound the length of a program that interpolates a noisy sample. In fact, we prove a deterministic worst-case bound on the program length needed to interpolate any given training set.

**Overview and intuition: How can we construct a short program interpolating a noisy sample?**

One approach is to memorize the sample $S$, or better yet, encode a good predictor $\mathsf{h}$ and then memorize all points in the sample that do not agree with $\mathsf{h}$. Such an interpolating predictor would generalize as well as $\mathsf{h}$ (since test examples will mostly not match the memorized examples). But is it the shortest? It would require storing all instances $x_i$ that do not agree with $\mathsf{h}$, and thus a description length of $L_S(\mathsf{h}) \cdot m \cdot \bar{b}(S)$.

The key is that we do not need to memorize the identities of the instances $x_i$ in the sample. We only need to remember the labels $y_i$, and so we can hope to prevent the description length from scaling linearly with $b$. To encode the information in the labels $y_i$, or rather their disagreement with $\mathsf{h}(x_i)$, we should need only $m \cdot H(L(\mathsf{h}))$ bits.

One approach to doing so is to hash the instances and store the labels (or disagreements) of the hash values. We could do this if our hash function has no collisions on $S$. The challenge in this approach is to determine how many bits are required to encode a hash function that is collision-free on $S$. Observe that such a function cannot be totally independent of $S$, since any hash function would have collisions on some $S$. Hence, any such hash function requires a description with super-constant length.

We take a more direct approach. We ask how difficult it would be to find and describe a "hash function" mapping instances to single bits such that the output values on the sample are exactly what we need them to be. Consider using a "random" binary function $\mathsf{hash}(x)$ where $\mathsf{hash}(x) \sim (\mathsf{Ber}\left(L(\mathsf{h})\right))$. Such a random function will interpolate with probability roughly $L(\mathsf{h})^{L(\mathsf{h})m}(1-L(\mathsf{h}))^{(1-L(\mathsf{h}))m} = 2^{-mH(L(\mathsf{h}))}$. If we use a pseudo-random generator with seed length $\gg mH(L(\mathsf{h}))$, one of the $\gg 2^{mH(\alpha)}$ "random" functions, corresponding to some specific seed value, should hopefully interpolate. We can then describe this function through its corresponding seed.

But how can we guarantee that some seed would work? To match the above probability calculation to the output of a pseudo-random generator (PRG), we need a PRG that generates $N$ bits that are $m$-way independent and marginally $\mathsf{Ber}\,(\alpha)$ using a seed of length $mH(\alpha)+O(\log m+\log\log N)$ (we need to generate $N = 2^{\bar{b}}$ bits, for each possible input $x$). We are not aware of any explicit PRG allowing this. Instead, the approach we take is to prove such a PRG must exist (Lemma 3.3) and then describe it as "the lexicographically first such PRG." This is a perfectly valid and precise description that can be encoded as a constant length program.

Notice that unlike the expensive instance memorization approach, the random hash predictor will not generalize as well as $\mathsf{h}$. The output $\mathsf{hash}(x)$ will have the same bias $L(\mathsf{h})$ on test instances, leading to a test error of $2L(\mathsf{h})(1-L(\mathsf{h}))$ (we make a mistake either if $\mathsf{h}$ does and we didn't correct it, or if $\mathsf{h}$ didn't make a mistake but we accidentally corrected it). In Section 4, we show through Lemma 4.2 that the MPL predictor indeed behaves this way.

**Formal results.** We establish a worst-case (deterministic) bound on the program length needed to memorize any labels (which we can think of as noise), in terms of the the bias of the labels. We then use this to describe a short program that interpolates the disagreement vs. a reference predictor on a random training set.

**Theorem 3.1** *Let $S = \{(x_i, y_i)$, for $i \in [m]\}$, where $x_i \in \{0,1\}^b$, $y_i \in \{0,1\}$, and the $x_i$ are pairwise distinct. Then, there exists a program $\mathsf{h}$ of length*

$$|\mathsf{h}| = m \cdot H\left(\frac{\sum_i y_i}{m}\right) + 3\log m + \log b + O(1)$$

*such that for all $(x_i, y_i) \in S$, we have $\mathsf{h}(x_i) = y_i$.*

For any program $\mathsf{h}$, we can apply Theorem 3.1 to the "labels" $y_i \oplus \mathsf{h}(x_i)$ to obtain the following:

**Corollary 3.2** *For $S \sim \mathcal{D}^m$ with quenched interpolation length $\bar{b}$, we have*

$$\mathop{\mathbb{E}}_{S}\left[|\mathsf{MPL}(S)|\right] \leq \min_{\textit{programs } \mathsf{h}} \left\{|\mathsf{h}| + m \cdot H(L(\mathsf{h})) + O\left(\log m + \log \bar{b}(m)\right)\right\}.$$

**Proof** For any program $\mathsf{h}$ and any $S$, let $\widetilde{\mathsf{h}_S}$ be the short program ensured by Theorem 3.1 for $\widetilde{S} = \left\{\left(x_i[:\bar{b}(S)]\,,\, y_i \oplus \mathsf{h}(x_i)\right)\right\}$. If $\widetilde{S}$ has repeated points, we remove them—Lemma A.6 in the Appendix shows that removing duplicates can only reduce $mH(\sum y_i/m)$, and so also the guaranteed length. The program $\mathsf{h}_S(x) = \mathsf{h}(x) \oplus \widetilde{\mathsf{h}_S}(x[:\bar{b}(S)])$ interpolates $S$ and is of length

$$|\mathsf{h}| + \left|\widetilde{\mathsf{h}_S}\right| + \log(\bar{b}(S)) + O(1) \leq |\mathsf{h}| + mH(L_S(\mathsf{h})) + O(\log m + \log \bar{b}(S)).$$

Taking an expectation over $S$ and recalling $\mathbb{E}[H(L_S(\mathsf{h}))] \leq H(\mathbb{E}[L_S(\mathsf{h})]) = H(L(\mathsf{h}))$ yields Corollary 3.2. ■

The key ingredient to proving Theorem 3.1 is a PRG based on a short seed length that can be used to generate "random" binary function $\mathsf{hash}(x)$ with $\mathsf{hash}(x) \sim \mathsf{Ber}(\alpha)$, where $\alpha = \sum_i y_i/m$. To make this precise, we consider a family of hash functions, indexed by a seed of length $r$, or in other words a seeded hash function of the form $\mathsf{hash}(\mathsf{seed}, x)$, where we will show that for every $S$, there exists a seed such that $x \mapsto \mathsf{hash}(\mathsf{seed}, x)$ interpolates $S$. In Lemma 3.3, we show that such a seeded hash function exist and bound the required seed length.

**Lemma 3.3** *For all $m, b \in \mathbb{N}$ and all $k \leq m$, and for*

$$r = m \cdot H\left(k/m\right) + \log m + \log b + 1$$

*there exists a function* $\mathsf{hash} : \{0,1\}^r \times \{0,1\}^b \to \{0,1\}$ *such that for all distinct $x_1, \ldots, x_m \in \{0,1\}^b$ and all $y_1, \ldots, y_m \in \{0,1\}$ with $\sum_i y_i = k$, there exists* $\mathsf{seed} \in \{0,1\}^r$*, such that for all $i$,* $\mathsf{hash}(\mathsf{seed}, x_i) = y_i$.

**Proof** To show existence, we use the probabilistic method. Specifically, we will show that a random function $G : \{0,1\}^r \times \{0,1\}^b \to \{0,1\}$ has positive probability of satisfying the Lemma requirements.

Let $\alpha = 1/m \cdot \sum_{i=1}^m y_i$. Choose $G$ at random by setting $G(\mathsf{seed}, x) = 1$ with probability $\alpha$ independently over all choices of seed and $x$. We will say that $G$ *fails* if there exists some $S = \{(x_i, y_i),\ \text{for } i \in [m]\}$ (with $x_i \neq x_j$ and $\sum_i y_i = k$) for which there is no corresponding seed such that interpolates $S$, i.e. s.t. $\forall_i G(\mathsf{seed}, x_i) = y_i$.

For a fixed $S$ and seed, the probability seed interpolates $S$ is exactly $\alpha^{\alpha m}(1-\alpha)^{(1-\alpha)m} = 2^{-m \cdot H(\alpha)}$, and so the probability there is no seed that interpolates $S$ is $\left(1 - 2^{mH(\alpha)}\right)^{(2^r)}$. Taking a union bound over all choices of $S$ yields

$$\begin{aligned}
\Pr_G[G \text{ fails}] &= \Pr_G[\text{there exists } S \text{ on which } G \text{ fails}] \leq \sum_S \Pr_G[\text{there is no seed that interpolates } S] \\
&= \sum_S \left(1 - 2^{-m \cdot H(\alpha)}\right)^{2^r} = \binom{m}{k}\binom{2^b}{m} \cdot \left(1 - 2^{-m \cdot H(\alpha)}\right)^{2^r} \\
&< \exp\left(m \ln(2) + mb \ln(2) - 2^{-m \cdot H(\alpha)} \cdot 2^r\right) < \exp(0) = 1 \qquad (3.1)
\end{aligned}$$

where in the last inequality we plugged in the prescribed seedlength $r$. Thus, $\Pr_G[G \text{ fails}] > 0$, and there exists at least one function $G$ that satisfies Lemma 3.3. ■

Lemma 3.3 establishes the *existence* of such a seeded hash function. However, to actually use it in a short program, we not only need the seed to be short, but also the description of the function hash to be short. Lemma 3.3 does *not* provide such a description as it is non-constructive, and we are not aware of any explicit construction. Fortunately, as we are not concerned with runtime, we can describe hash through a (short and explicit) program that enumerates over all $2^{2^{r+b}}$ possible functions and picks the first one lexicographically. Using this "explicit" short programmatic description of hash, we finish the proof of Theorem 3.1.

**Proof** [Theorem 3.1] Let GenHash be a program that takes as input three integers $(m, k, b)$, calculates $r$ based on them as defined in Lemma 3.3, enumerates over all functions $G : \{0,1\}^r \times \{0,1\}^b \to \{0,1\}$, and returns the lexicographically first function that satisfies Lemma 3.3. The size of the output of GenHash, which is a huge lookup table, depends on its inputs, but the function description itself is fixed, with fixed length $|\mathsf{GenHash}|$ (e.g. $|\mathsf{GenHash}| < 1000$ in compressed Python or C++ with standard libraries). Our program for interpolating $S$ is

$$\mathsf{h}_S(x) = \mathsf{GenHash}(m, k, \mathsf{bitlength}(x))(\mathsf{seed}, x) \tag{3.2}$$

where $k = \sum_i y_i$ and seed is the seed the interpolates $S$ using the lexicographically first function that satisfies Lemma 3.3. This seed is hard-coded into the program. The description length of program is thus $|\mathsf{GenHash}| + |m| + |k| + |\mathsf{seed}| + O(1) = r + 2\log m + O(1) = mH(m/k) + 3\log m + \log b + O(1)$ (where here $|a|$ is the description length of $a$). ■

**Tightness of dependence on Disambiguation Prefix Length.** One might wonder whether it is possible to avoid, or improve, the dependence on $b$ in Theorem 3.1 and thus on $\bar{b}$ in Corollary 3.2. Unfortunately, this is not possible. To see this, for any $b$, we will construct a sample $S = \{(x_1, y_1), (x_2, y_2)\}$, with $x_1, x_2 \in \{0,1\}^b$ that cannot be interpolated using any program of length less than $\log b$. We will do so by associating for every $x \in \{0,1\}^b$, a vector $\phi(x) \in \{0,1\}^N$ consisting of the output of running each of the $N < b$ programs of length $< \log b$ on $x$. I.e. $\phi(x)[i] = \mathsf{h}_i(x)$, where $\mathsf{h}_i$ is the lexicographical $i$th program (and we can set $\phi(x)[i] = 0$ if the program doesn't stop and output a valid value). There are $2^b$ different $x$s, but only $2^N < 2^b$ possible $\phi(x)$. Hence, there must be two inputs $x_1 \neq x_2$ with $\phi(x_1) = \phi(x_2)$, i.e. such that no short program can distinguish between them. The sample $S = \{(x_1, 0), (x_2, 1)\}$ can thus not be interpolated by any program of length less than $\log b$.

## 4. Generalization

After establishing in Section 3 an upper bounds on the length of $\mathsf{MPL}(S)$, we will now prove Theorems 2.2 and 2.3 by combining these with guarantees on the generalization error of any learning rule outputting a short program.

**Agnostic guarantees and proof of Theorem 2.2.** The following generalization guarantee in terms of program length is a a tight version of the standard description-length based guarantee:

**Lemma 4.1** *For any distribution $\mathcal{D}$, any interpolating learning rule $A$, and any sample size $m$:*

$$-\log\left(1 - \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[L(A(S))\right]\right) \leq \frac{I(S; A(S))}{m} \leq \frac{\mathbb{E}\left[|A(S)|\right]}{m}.$$

We can obtain a high probability version of the Lemma 4.1, in terms of $\sup|A(S)|$, using a union bound over all short programs. This is also a special case of the PAC-Bayes Bound [McA03], noting that $D_{\mathsf{KL}}(0 \parallel \beta) = -\log(1-\beta)$. Raginsky, Rakhlin, Tsao, Wu, and Xu [RRTWX16], Russo and Zou [RZ19], and Xu and Raginsky [XR17] obtain similar (and more general) bounds, but bound $L(A)^2$ rather than $-\log(1 - L(A))$ on the left-hand side. Since in our case the right-hand-side will not vanish, this distinction is significant—a tight bound, even up to constant factors, is essential for obtaining tempered overfitting guarantees.

For completeness, we provide a proof of Lemma 4.1, which we will also use as a basis for a more refined analysis in Lemma 4.2. The proof captures the following information argument: Suppose that the error rate of $A(S)$ outside $S$ is very different from its error rate inside $S$. Then, one could apply $A(S)$ to a large number of samples from $\mathcal{D}$ and gain information about which samples were more likely to be in $S$. In other words, this yields a lower bound on $I(S; A(S))$.

**Proof** [Proof of Lemma 4.1] Without loss of generality, we will assume that $A$ is symmetric over the training examples–otherwise, consider a randomized rule that first randomly permutes that training examples and then applies $A$ to this permuted training set. The Distribution over training sets and

We denote $\mathsf{U} = A(S)$. Observe that $\mathsf{U}$ is a random variable. We have

$$\mathbb{E}\left[|\mathsf{U}|\right] \geq H(\mathsf{U}) \geq I(S;\mathsf{U}) \geq \sum_i I((X_i, Y_i); \mathsf{U}) = mI((X_1, Y_1); \mathsf{U}) \tag{4.1}$$

where the first inequality is Shannon's source coding bound and the third inequality is due to the independence of $(X_i, Y_i)$ (Lemma A.3 in the Appendix). In the third inequality we assumed, without loss of generality, that $A$ is symmetric over the training examples—otherwise, consider a rule $\mathsf{U} = A(\pi(S))$ that first randomly permutes $S$ and then applies $A$ to the random permutation. Since $S$ and a random permutation of $S$ are identically distributed, the distribution of $\mathsf{U}$, and thus its expected error, are also unchanged.

To analyze $I((X_1, Y_1); \mathsf{U})$, we rely on the variational bound $I(A; B) \geq \mathop{\mathbb{E}}_{A,B\sim p}\left[\log \frac{dq(A|B)}{dp_A(A)}\right]$, where $p(A, B)$ is the true joint distribution, with marginal $p_A$, and $q(A|B)$ is any proposed conditional distribution (Lemma A.2). In our case, we use the proposal distribution $q(X_1, Y_1|\mathsf{U})$ defined by

$$dq(x, y|\mathsf{u}) = \tfrac{1}{Z_\mathsf{u}} \mathbb{1}\{\mathsf{u}(x) = y\}\, dp(x, y) \tag{4.2}$$

where $p(x, y)$ is the true marginal over $X_1, Y_1$ (i.e. the source distribution $\mathcal{D}$), and $Z_\mathsf{u} = \mathop{\mathbb{E}}_{X,Y\sim p}[\mathbb{1}\{\mathsf{u}(X) = Y\}] = 1 - L(\mathsf{u})$. This proposal distribution amounts to bounding the mutual information by the information $\mathsf{U}$ tells us about $(X_1, Y_1)$ by telling us that $(X_1, Y_1)$ satisfies $\mathsf{U}(X) = Y$ (since $\mathsf{U} = A(S)$ interpolates the training points). We now calculate.

$$\begin{aligned} I((X_1, Y_1); \mathsf{U}) &\geq \mathop{\mathbb{E}}_S\left[\log\left(\frac{dq(X_1, Y_1|\mathsf{U})}{dp(X_1, Y_1)}\right)\right] = \mathop{\mathbb{E}}_S\left[\log\left(\frac{\mathbb{1}\{\mathsf{U}(X_1) = Y_1\}}{Z_\mathsf{U}}\right)\right] \\ &= \mathop{\mathbb{E}}_S\left[\log\frac{1}{Z_\mathsf{U}}\right] = \mathop{\mathbb{E}}_S\left[-\log\left(1 - L(\mathsf{U})\right)\right] \geq -\log\left(1 - \mathop{\mathbb{E}}_S\left[L(\mathsf{U})\right]\right) \end{aligned} \tag{4.3}$$

Recalling that $\mathsf{U} = A(S)$, and combining (4.1) and (4.3) we obtain the statement of Lemma 4.1. ■

*Proof of Theorem 2.2.* Plugging Corollary 3.2 into Lemma 4.1. we have for any $\mathsf{h}$,

$$\mathbb{E}\left[L(\mathsf{MPL}(S))\right] \leq 1 - 2^{-H(L(\mathsf{h})) - \left(|\mathsf{h}| + O\left(\log m + \log \bar{b}(m)\right)\right)/m}. \tag{4.4}$$

As $m \to \infty$, the right hand side converges to $\ell_{\mathsf{ag}}(L(\mathsf{h})) = 1 - 2^{-H(L(\mathsf{h}))}$. We now use the following inequality (proved as Lemma A.4 in the Appendix) to bound the convergence rate to $\ell_{\mathsf{ag}}(L(\mathsf{h}))$:

$$\text{For } C \geq 0 \text{ and } 0 \leq \alpha \leq 1, \quad 1 - 2^{-H(\alpha)-C} \leq 1 - 2^{-H(\alpha)} + C \qquad \text{(Lemma A.4)}$$

Combining (4.4) and (Lemma A.4) yields Theorem 2.2. ■

**Tightness of agnostic generalization guarantee.** Before continuing, we note that Lemma 4.1 is tight, and we cannot hope to get a better guarantee solely in terms of program length. To see this, consider a distribution $\mathcal{D}$ where $X$ is i.i.d. uniform bits, and $Y = 0$. Although this is a very easy distribution to interpolate, consider, for any $0 < \alpha \leq 0.5$, an interpolating learning rule $A$ that searches for a random function $\mathsf{hash}(x)$ that interpolates the data, where $\mathsf{hash}(x) \sim \mathsf{Ber}(\alpha)$ independently for all $x$. Using arguments similar to the proof of Theorem 3.1, we can calculate that the probability of such a function interpolating the data is $(1-\alpha)^m$, and we can therefore encode such a function using $\mathbb{E}[|A(S)|] = O(m\log(1-\alpha) + \log m)$ bits. For large $m$, the right-hand-side of Lemma 4.1 is therefore $\log(1-\alpha) + o(1)$. This is tight since $L(A) = \alpha = 1 - 2^{\log(1-\alpha)}$.

**Random label noise and proof of Theorem 2.3.** Although Lemma 4.1 is worst-case optimal, we show a tighter generalization guarantee for well specified distributions with independent label noise:

**Lemma 4.2** *For any source distribution $\mathcal{D}$ such that $Y|X = \mathsf{h}^\star(X) \oplus \mathsf{Ber}(L^\star)$, any learning rule $A(S)$ returning an interpolating program, and any sample size $m$, we have*

$$D_{\mathsf{KL}}\left(L^\star \,\middle\|\, \mathop{\mathbb{E}}_S\left[\Pr_{X\sim\mathcal{D}}\left[A(S)(X) \neq \mathsf{h}^\star(X)\right]\right]\right) \leq \frac{\mathop{\mathbb{E}}_S\left[|A(S)|\right] - m\cdot H\left(L^\star\right)}{m} \tag{A}$$

*and therefore*

$$|L(A) - 2L^\star(1-L^\star)| \leq O\left(\frac{\mathop{\mathbb{E}}_S\left[|A(S)|\right] - m\cdot H(L^\star)}{m} + \sqrt{L^\star\cdot\frac{\mathop{\mathbb{E}}_S\left[|A(S)|\right] - m\cdot H(L^\star)}{m}}\right). \tag{B}$$

The proof is again information-theoretic based on the following intuition: The agreement rate of $A(S)$ with $\mathsf{h}^\star$ inside $S$ is exactly $L^\star$. If the agreement rate outside $S$ differs significantly, we can use it to construct a predictor for which $x$s are in $S$. Hence, the output of $A(S)$ has information about the $X_i$s. But $A(S)$ needs at least $mH(L^\star)$ bits of information just for encoding the noise on the labels, and so if its description length is not much more than $mH(L^\star)$, it cannot also contain information about which $x$s are in $S$ (i.e. it does not have enough information capacity for also memorizing anything about the $X_i$s).

**Proof** [Proof of Lemma 4.2] Denoting $\mathsf{U} = A(S)$ as before, and again assuming symmetry without loss of generality, we have

$$\mathbb{E}[|\mathsf{U}|] \geq mI(X_1, Y_1; \mathsf{U}) = m(X_1; \mathsf{U}) + mI(Y_1; \mathsf{U}|X_1) \tag{4.5}$$

where the inequality is the same as in the proof of Lemma 4.1. We evaluate:

$$I(Y_1; \mathsf{U}|X_1) = H(Y_1|X_1) - H(Y_1|\mathsf{U}, X_1) = H(L^\star) - 0 = H(L^\star) \tag{4.6}$$

where in the second equality, the first term follows since $Y_1|X_1 \sim \mathsf{Ber}(L^\star)$ based on the noise model, and the second is because $Y_1 = \mathsf{U}(X_1)$ is a deterministic function of $\mathsf{U}, X_1$.

In order to bound $I(X_1; \mathsf{U})$, it will be convenient to define $\tilde{\mathsf{U}}$, which is a deterministic function of $\mathsf{U}$ (and hence also a random variable) with $\tilde{\mathsf{U}}(x) = \mathsf{U}(x) \oplus \mathsf{h}^\star(x)$ (recall $\mathsf{h}^\star$ is fixed and deterministic here). We will also denote $\widetilde{L} = \mathbb{E}[\Pr[\mathsf{U}(X) \neq \mathsf{h}^\star(X)]] = \mathbb{E}\left[\tilde{\mathsf{U}}(X)\right]$ the disagreement probability

we want to bound. Now, to bound $I(X_1;\mathsf{U})$, we will use the same variational bound, this time with the proposal distribution:

$$dq_{X|\mathsf{U}}(x|\mathsf{u}) = \frac{1}{Z_{\mathsf{u}}} \frac{p_{\mathsf{Ber}(L^\star)}(\tilde{\mathsf{u}}(x))}{p_{\mathsf{Ber}(\widetilde{L})}(\tilde{\mathsf{u}}(x))} dp(x) \tag{4.7}$$

where $p_{\mathsf{Ber}(\alpha)}(0) = 1-\alpha, p_{\mathsf{Ber}(\alpha)}(1) = \alpha$ is the Bernoulli probability mass function, and again $p(x)$ is the true (population) marginal. This proposal distribution is the best we can do solely in terms of $\tilde{\mathsf{u}}(x)$, since we know that inside $S$ we have $\tilde{\mathsf{U}}(X_1) = \mathsf{U}(X_1) \oplus \mathsf{h}^\star(X_1) = Y_1 \oplus \mathsf{h}^\star(X_1) \sim \mathsf{Ber}\left(L^\star\right)$ while for a random $X$, $\tilde{\mathsf{U}}(X) = \mathsf{U}(X) \oplus \mathsf{h}^\star(X) \sim \mathsf{Ber}\left(\widetilde{L}\right)$, by definition of $\widetilde{L}$. We calculate the partition function.

$$Z_{\mathsf{u}} = \mathop{\mathbb{E}}_{X\sim p}\left[\frac{p_{\mathsf{Ber}(L^\star)}\left(\tilde{\mathsf{u}}(X)\right)}{p_{\mathsf{Ber}(\widetilde{L})}\left(\tilde{\mathsf{u}}(X)\right)}\right] = \mathop{\mathsf{Pr}}_{X}\left[\tilde{\mathsf{u}}(X)=1\right] \cdot \frac{L^\star}{\widetilde{L}} + \mathop{\mathsf{Pr}}_{X}\left[\tilde{\mathsf{u}}(X)=0\right] \cdot \frac{1-L^\star}{1-\widetilde{L}} \tag{4.8}$$

Taking an expectation over $\mathsf{U}$, we have $\mathbb{E}\left[Z_{\mathsf{U}}\right] = \frac{L^\star}{\widetilde{L}} \cdot \widetilde{L} + \frac{1-L^\star}{1-\widetilde{L}} \cdot (1-\widetilde{L}) = 1$. Applying the variational bound (Lemma A.2) we have:

$$\begin{aligned} I(X_1;\mathsf{U}) &\geq \mathop{\mathbb{E}}_{X_1,\mathsf{U}}\left[\log\left(\frac{dq(X_1|\mathsf{U})}{dp(X_1)}\right)\right] = \mathop{\mathbb{E}}_{X_1,\mathsf{U}}\left[\log\left(\frac{p_{\mathsf{Ber}(L^\star)}(\tilde{\mathsf{U}}(X_1))}{p_{\mathsf{Ber}(\widetilde{L})}(\tilde{\mathsf{U}}(X_1))} \cdot \frac{1}{Z_{\mathsf{U}}}\right)\right] \\ &\geq \mathop{\mathbb{E}}_{\tilde{\mathsf{U}}(X_1)\sim\mathsf{Ber}(L^\star)}\left[\log\left(\frac{p_{\mathsf{Ber}(L^\star)}(\tilde{\mathsf{U}}(X_i))}{p_{\mathsf{Ber}(\widetilde{L})}(\tilde{\mathsf{U}}(X_i))}\right)\right] - \log\left(\mathop{\mathbb{E}}_{\mathsf{U}}\left[Z_{\mathsf{U}}\right]\right) = D_{\mathsf{KL}}\left(L^\star \,\middle\|\, \widetilde{L}\right) \end{aligned} \tag{4.9}$$

Where the inequality is due to Jensen on the second term, and we then use $\mathbb{E}\left[Z_{\mathsf{U}}\right] = 1$.

Plugging in (4.9) and (4.6) into (4.5) yields part (A) of the Lemma. To obtain part (B), we first use the inequality $|\beta-\alpha| \leq 2D_{\mathsf{KL}}(\alpha \parallel \beta) + \sqrt{2\alpha D_{\mathsf{KL}}(\alpha \parallel \beta)}$ (Lemma A.5) to obtain $\left|\widetilde{L} - L^\star\right| \leq 2R + \sqrt{2L^\star R}$, where $R$ is the right hand side of part (A). And since $L(A) = \widetilde{L}(1-L^\star) + (1-\widetilde{L})L^\star$, we have $|L(A) - 2L^\star(1-L^\star)| = (1-2L^\star)\left|\widetilde{L} - \widetilde{L}\right| \leq \left|\widetilde{L} - \widetilde{L}\right|$. Combining the two inequalities yields part (B). ■

*Proof of Theorem 2.3.* Plugging in Corollary 3.2 into Part (B) of Lemma 4.2 yields Theorem 2.3. ■

## 5. Tightness and Discussion

For MPL interpolation in the presence of random label noise, we provide a precise characterization of the effect of overfitting. In this case, unlike the optimally tuned SRM, which converges to the Bayes optimal predictor, the interpolating MPL predictor will converge to sampling from the posterior, yielding up to twice the Bayes error. This is similar to the behavior of a 1-nearest-neighbor rule (although the actual predictions will of course be very different), the observed behavior of certain neural networks [NB20], and perhaps kernels [MSAPBN22]. This is a "tempered" behavior, where for any non-trivial Bayes error $0 \leq L^\star \leq 0.5$, the limiting MPL error $L^\star < \ell_{\mathsf{samp}}(L^\star) < 0.5$ is strictly worse than Bayes, but still provides an edge over random guessing.

In the more general agnostic case, we give only an upper bound, depicted in Figure 1. Although strictly worse than the sampling behavior with random label noise, this behavior is still tempered (Corollary 2.4): if some computable function has non-trivial error $L(\mathsf{h}) < 0.5$, the optimally tuned SRM will converge to at most this error, and MPL might suffer due to overfitting, but we will still yield (as $m \to \infty$) an edge over random guessing and error at most $\ell_{\mathsf{ag}}(L(\mathsf{h})) < 0.5$.

**A uniform convergence approach.** Although we use an information-theoretic approach in our generalization proofs, the proofs essentially rely on a uniform guarantee over all short programs. In particular, they hold for *any* interpolation rule, not only MPL. The connection to MPL is only realized by plugging in the program length we can ensure for MPL. This is similar in spirit to the uniform convergence of interpolator arguments of Koehler, Zhou, Sutherland, and Srebro [KZSS21], which separately bound the norm of the min-norm predictor, and then analyze uniform convergence over the appropriate norm ball.

**Tempered overfitting with finite samples.** An important feature of our results is that we do not look only at the asymptotic behavior, but ask also about the effect of overfitting with a finite number of samples. This allows us to draw a comparison against the finite-sample agnostic SRM guarantee (1.1). In particular, with finite $m$, the competitor $\mathsf{h}$ with which we want to compete (i.e. the one minimizing the right hand side of (1.1)) might be different and depend on $m$. Indeed, our finite sample agnostic guarantee (Theorem 2.2) shows that we can compete with the $m$-dependent $\mathsf{h}$ with which SRM competes, with a "tempered" effect on the error. This is similar in spirit to the study of how minimum norm interpolation can adapt the approximation error to the sample complexity as recently studied by Misiakiewicz [Mis22] and Xiao, Pennington, Misiakiewicz, Hu, and Lu [XPMHL22].

**Tightness of agnostic guarantee.** One might ask whether our agnostic upper bound is tight and whether it is possible to identify its exact behavior.

First, we point out that MPL might yield limiting error anywhere between the Bayes error and the error of the sampling predictor, i.e. anywhere in the red region between the Bayes optimal line and sampling curve in Figure 1. To see this, consider a source distribution where $X[1] \sim \mathsf{Ber}(\alpha)$, the remaining bits of $X$ are i.i.d.$\mathsf{Ber}(0.5)$, and $Y = X[2]$ if $X[1] = 0$, but $Y = \mathsf{Ber}(\beta)$ if $X[1] = 1$. It is easy to verify that $L^\star = \alpha\beta$ while $L(\mathsf{MPL}) \overset{m\to\infty}{\to} L_{\mathsf{MPL}} = 2\alpha\beta(1-\beta)$, which allows us to get any $0 \leq L^\star \leq L_{\mathsf{MPL}} \leq \ell_{\mathsf{samp}}(L^\star) \leq 0.5$ by varying $\alpha$ and $\beta$. This is the same sampling behavior and same asymptotic error that will be reaches by other sampling-type over-fitting predictors, such as 1-nearest-neighbor.

We do not know whether there are source distributions for which MPL will yield errors above the sampling curve $\ell_{\mathsf{samp}}$ (the green region in Figure 1), or whether the difference between $\ell_{\mathsf{samp}}$ and $\ell_{\mathsf{ag}}$ is due to a looseness in Theorem 2.2. In Sections 3 and Section 4 we argued that the description length bound in Corollary 3.2 and the generalization bound in terms of program length in Lemma 4.1 are tight. This implies our proof technique, which separately asks what length programs we need to consider and then uses what is essentially a uniform generalization guarantee for all short programs, cannot improve beyond Theorem 2.2 (in the agnostic case). But although this proof technique cannot be improved, it is possible that by analyzing specific properties of the MPL, it is possible to significantly strengthen 2.2, perhaps replacing $\ell_{\mathsf{ag}}$ with $\ell_{\mathsf{samp}}$ also in the agnostic case, and we leave this as an open question.

It is useful to note that if the posterior $\eta(x) = P(Y = 1|x)$ is computable, MPL should also converge to a sampling classifier and yield limiting error $L(\mathsf{MPL}) \overset{m\to\infty}{\to} L_{\mathsf{MPL}} \leq \ell_{\mathsf{samp}}(L^*)$ where $L^\star$ is the Bayes error. In fact, we suspect it is possible to generalize Theorem 2.3 to show

$$\mathop{\mathbb{E}}_{X}\left[D_{\mathsf{KL}}\left(\eta(X)\,\middle\|\,\Pr_{S}\left[\mathsf{MPL}(S)(X)=1\right]\right)\right] \leq \frac{|\eta| + O(\log m + \log \bar{b}(m))}{m}, \tag{5.1}$$

where $|\eta|$ is the description length of the (computable) posterior $\eta$. This is a more general situation than random label noise added to a computable Bayes optimal predictor, where $\eta(x) = L^\star + \mathsf{h}^\star(1 - 2L^\star)$. The scenario where MPL might yield error above $\ell_{\mathsf{samp}}(L(\mathsf{h}^\star))$, is thus when the Bayes predictor $\mathsf{h}^\star(x) = \mathsf{Sign}(\eta(X) - 0.5)$ is computable, but the posterior $\eta(x)$ itself is not. Even without getting to non-computability, we can consider a situation where the Bayes optimal predictor has a very short description, but the posterior requires a much longer program, and ask whether this would result in large gaps between the optimally balanced SRM and the interpolating MPL.

**Different notions of description length and different inductive bias.** We considered minimum description length interpolation learning in the Turing or Kolmogorov sense, i.e. by minimizing program length. This is arguably the most general notion of description length if we would like the learned predictor to actually be computable.

Still, one can instead think more abstractly of logical descriptions that allow quantifiers over infinite domains. Our results hold also in these more general settings, or any other notion that subsumes or extend Turing computation. More specifically, the only descriptive power we require is that we can describe "lexicographically first function satisfying Lemma 3.3."

Alternatively, one might consider more limited notions of description, e.g. limiting to only programs with short runtime. Specifically, one can consider the learning rule[4] MinRuntime that selects the program with the minimal (worst case) runtime that interpolates $S$. Or almost equivalently (up to some polynomial relationship), we can further restrict the set of functions to neural networks and study the learning rule MinNetwork, which returns the neural network[5] with the minimal number of edges that interpolates the training set. Our analysis does not apply to MinRuntime or MinNetwork since the short program we construct has a doubly-exponential runtime. An explicit and efficiently computable pseudo-random generator, generating $N$ bits that are (approximately) $m$-way independent and marginally $\mathsf{Ber}(\alpha)$ using a seed length of $m \cdot H(\alpha) + O(\log m + \log\log N)$ (or even a worse dependence on $N$), would extend our results to min-runtime or min-size-neural-network interpolation.

More generally, our analysis can be viewed as providing a sufficient condition on an inductive bias $c(\mathsf{h})$ such that minimum-$c(\mathsf{h})$ interpolation exhibits tempered overfitting. Roughly speaking, as long as the inductive bias allows us to encode "random function" with capacity (i.e. the capacity of the sublevel set of $c(\cdot)$ containing these random functions) not much larger than the capacity of the set of such "random functions", it should be the case that minimum-$c(\mathsf{h})$ interpolation is tempered in the sense of Theorems 2.2 and 2.3.

4. While still abstract, the learning rule MinRuntime is more useful as a reference universal rule, since we want our predictor to not merely be computable, but also be tractable with reasonable runtime [Val84]. Additionally, MinRuntime $\in$ NP, and for all we know might be poly-time computable, unlike MPL which is uncomputable.
5. More formally, we fix the activation function, e.g. to ReLU activation, and search over all architecture graphs and all edge weights.

**Tightness of dependence on the Disambiguation Prefix Length.** Another open technical question is whether the mild dependence on the quenched disambiguation prefix length in Theorems 2.2 and 2.3 is necessary. Again, we argue that it is necessary for bounding the description length, and so for our proof technique. But the examples which require long programs due to the differences between instances being hidden in far-away and hard-to-describe bits, do not show these long programs do not generalize well. We do not know and leave it open whether the dependence on $\overline{b}$ in Theorems 2.2 and 2.3 is necessary, or whether different techniques and specific analysis of the MPL can avoid these.

**Summary.** With the growing interest in noisy interpolation learning, and obtaining an understanding and characterization of the "benignness" and/or harm of overfitting, we find it instructive to consider what is perhaps the most basic and fundamental learning principal, with roots going back to the first discussions of machine learning and inductive inference [Sol60]. We hope that our study will help direct the search for the fundamental principles of what "makes" overfitting benign or catastrophic. We would also like to see our tempered finite sample agnostic guarantee (Theorem 2.2) as a template for studying how overfitting compares with the optimally balanced approached (the SRM guarantee of (1.1) in our case), instead of focusing on comparing the asymptotic behavior and seeking consistency, which is frequently less relevant for learning.

### Acknowledgments

We thank David McAllester for discussions on MPL learning, Madhur Tulsiani and Roei Tell for discussions on random number generators, Xiaohan Zhu for pointing out an inaccuracy in the original proof of Lemmas 4.1 and 4.2, Andrew Barron for discussions on contrasting our work with risk bounds for the MDL model selection criteria, and an anonymous reviewer for pointing out the work of Grünwald and Langford [GL07]. We are especially grateful to Alexander Razborov for pointing out the argument in the lower bound in terms $b$ (end of Section 3), which led to the proof approach for Theorem 3.1 using enumeration as a method of description.

This research was supported in part by the NSF-Simons Collaboration on the Mathematics of Deep Learning and an NSF-Tripod-supported Institute for Data Economics and Algorithms. NSM was funded in part by a United States NSF Graduate Research Fellowship.

## References

[BLLT20] Peter L. Bartlett, Philip M. Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

[BHM18] Mikhail Belkin, Daniel J Hsu, and Partha Mitra. Overfitting or perfect fitting? Risk bounds for classification and regression rules that interpolate. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[BRT19] Mikhail Belkin, Alexander Rakhlin, and Alexandre B. Tsybakov. Does data interpolation contradict statistical optimality? In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[BEHW87] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Occam's razor. *Information processing letters*, 24(6):377–380, 1987.

[CL21] Niladri S. Chatterji and Philip M. Long. Finite-sample analysis of interpolating linear classifiers in the overparameterized regime. *Journal of Machine Learning Research*, 22(129):1–30, 2021.

[CT06] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. A Wiley-Interscience publication. Wiley, 2006.

[GL07] Peter Grünwald and John Langford. Suboptimal behavior of bayes and mdl in classification under misspecification. *Machine Learning*, 66:119–149, 2007.

[HMRT] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J. Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *The Annals of Statistics*, 50(2).

[KZSS21] Frederic Koehler, Lijia Zhou, Danica J Sutherland, and Nathan Srebro. Uniform convergence of interpolators: gaussian width, norm bounds and benign overfitting. *Advances in Neural Information Processing Systems*, 34:20657–20668, 2021.

[MSAPBN22] Neil Mallinar, James Simon, Amirhesam Abedsoltan, Parthe Pandit, Misha Belkin, and Preetum Nakkiran. Benign, tempered, or catastrophic: toward a refined taxonomy of overfitting. In *Advances in Neural Information Processing Systems*, 2022.

[McA03] David McAllester. Simplified PAC-bayesian margin bounds. In *Learning Theory and Kernel Machines*, pages 203–215. Springer, 2003.

[Mis22] Theodor Misiakiewicz. Spectrum of inner-product kernel matrices in the polynomial regime and multiple descent phenomenon in kernel ridge regression. *arXiv preprint arXiv:2204.10425*, 2022.

[MRSY20] Andrea Montanari, Feng Ruan, Youngtak Sohn, and Jun Yan. The generalization error of max-margin linear classifiers: high-dimensional asymptotics in the overparametrized regime. *Preprint, arXiv:1911.01544*, 2020.

[MNSBHS21] Vidya Muthukumar, Adhyyan Narang, Vignesh Subramanian, Mikhail Belkin, Daniel Hsu, and Anant Sahai. Classification vs regression in overparameterized regimes: does the loss function matter? *Journal of Machine Learning Research*, 22(222):1–69, 2021.

[NB20] Preetum Nakkiran and Yamini Bansal. Distributional generalization: a new kind of generalization. *arXiv preprint arXiv:2009.08092*, 2020.

[NDR20] Jeffrey Negrea, Gintare Karolina Dziugaite, and Daniel M. Roy. In defense of uniform convergence: generalization via derandomization with an application to interpolating predictors. In *International Conference on Machine Learning*, 2020.

[RRTWX16] Maxim Raginsky, Alexander Rakhlin, Matthew Tsao, Yihong Wu, and Aolin Xu. Information-theoretic analysis of stability and bias of learning algorithms. In *2016 IEEE Information Theory Workshop (ITW)*, pages 26–30. IEEE, 2016.

[Ris78] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[RZ19] Daniel Russo and James Zou. How much does your data exploration overfit? controlling bias via information usage. *IEEE Transactions on Information Theory*, 66(1):302–323, 2019.

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[Sol60] Ray J Solomonoff. A preliminary report on a general theory of inductive inference. In 1960.

[Val84] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[XPMHL22] Lechao Xiao, Jeffrey Pennington, Theodor Misiakiewicz, Hong Hu, and Yue Lu. Precise learning curves and higher-order scalings for dot-product kernel regression. In *Advances in Neural Information Processing Systems*, 2022.

[XR17] Aolin Xu and Maxim Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. *Advances in Neural Information Processing Systems*, 30, 2017.

## Appendix A. Identities and Inequalities from Information Theory

We present and either cite or prove several identities and inequalities we use in our proofs.

**Lemma A.1 (Chain Rule of Mutual Information; see p. 42 of [CT06])** *For any random variables $A_1, A_2$, and $B$,*

$$I((A_1, A_2); B) = I(A_2; B|A_1) + I(A_1; B).$$

**Lemma A.2** *Let $A$ and $B$ be any two random variables with associated marginal distributions $p_A$, $p_B$, and joint $p_{A,B}$. Let $q_{A|B}$ be any conditional distribution (i.e. such that for any $b$, $q_{A|B}(\cdot, b)$ is a normalized non-negative measure). Then:*

$$I(A; B) \geq \mathop{\mathbb{E}}_{A,B\sim p_{A,B}} \left[\log\left(\frac{dq_{A|B}(A|B)}{dp_A(A)}\right)\right]$$

**Proof** The proof essentially uses the chain rule for KL-divergence. We write

$$\begin{aligned}
I(A; B) = D_{\mathsf{KL}}\left(p_{A|B} \,\middle\|\, p_A\right) &= \mathop{\mathbb{E}}_{A,B\sim p_{A,B}} \left[\log\left(\frac{dp_{A|B}(A|B)}{dp_A(A)}\right)\right] && \text{(A.1)}\\
&= \mathop{\mathbb{E}}_{A,B\sim p_{A,B}} \left[\log\left(\frac{dp_{A|B}(A|B)}{dp_A(A)} \cdot \frac{dq_{A|B}(A|B)}{dq_{A|B}(A|B)}\right)\right] && \text{(A.2)}\\
&= \mathop{\mathbb{E}}_{A,B\sim p_{A,B}} \left[\log\left(\frac{dq_{A|B}(A|B)}{dp_A(A)}\right)\right] + \mathop{\mathbb{E}}_{A,B\sim p_{A,B}} \left[\log\left(\frac{dp_{A|B}(A|B)}{dq_{A|B}(A|B)}\right)\right] && \text{(A.3)}\\
&= \mathop{\mathbb{E}}_{A,B\sim p_{A,B}} \left[\log\left(\frac{dq_{A|B}(A|B)}{dp_A(A)}\right)\right] + \mathop{\mathbb{E}}_{B\sim p_B} \left[D_{\mathsf{KL}}\left(p_{A|B} \,\middle\|\, q_{A|B}\right)\right] && \text{(A.4)}\\
&\geq \mathop{\mathbb{E}}_{A,B\sim p_{A,B}} \left[\log\left(\frac{dq_{A|B}(A|B)}{dp_A(A)}\right)\right] && \text{(A.5)}
\end{aligned}$$

where the inequality follows from the non-negativity of the KL divergence. ■

**Lemma A.3** *Let $A_1, A_2, B$ be random variables where $A_1$ and $A_2$ are independent. Then*

$$I((A_1, A_2); B) \geq I(A_1; B) + I(A_2; B).$$

**Proof** We use Lemma A.2 with the conditional distribution $q_{A_1,A_2|B} = p_{A_1|B} \cdot p_{A_2|B}$ below.

$$\begin{aligned}
I((A_1, A_2); B) &\geq \mathop{\mathbb{E}}_{A_1,A_2,B} \left[\log\left(\frac{dp_{A_1|B}(A_1|B) \cdot dp_{A_2|B}(A_2|B)}{dp_{A_1,A_2}(A_1, A_2)}\right)\right] && \text{(A.6)}\\
&= \mathop{\mathbb{E}}_{A_1,B} \left[\log\left(\frac{dp_{A_1|B}(A_1|B)}{dp_{A_1}(A_1)}\right)\right] + \mathop{\mathbb{E}}_{A_2,B} \left[\log\left(\frac{dp_{A_2|B}(A_2|B)}{dp_{A_2}(A_2)}\right)\right] && \text{(A.7)}\\
&= I(A_1; B) + I(A_2; B)
\end{aligned}$$

This concludes the proof of Lemma A.3. ■

**Lemma A.4** *For $C \geq 0$ and $0 \leq \alpha \leq 1$, $1 - 2^{-H(\alpha)-C} \leq \ell_{\mathsf{ag}}(\alpha) + C$.*

**Proof** We first prove that for all $\alpha, \beta \in (0,1)$ such that $\beta \geq 1 - 2^{-H(\alpha)}$, we have

$$\log\left(\frac{1}{1-\beta}\right) - H(\alpha) \geq \beta - \left(1 - 2^{-H(\alpha)}\right). \tag{A.8}$$

Let $g(a) \coloneqq -\log(1-a) - a$. Notice that the derivative of $g(a)$ is $g'(a) = -1 + (\ln(2) - a\ln(2))^{-1}$.

First, we show that for all $a \in (0,1)$, we have $g(a) \geq 0$. We do so by showing that $g(0) = 0$ and that $g(a)$ is increasing on $a \in (0,1)$. It is easy to see that equality is achieved at $a = 0$, so it is enough to show that $g'(a) \geq 0$ for all $a \geq 0$. This follows immediately since $\ln(2) < 0$.

Next, we analyze $g(\beta) - g(1 - 2^{-H(\alpha)})$. Since $g(\cdot)$ is nonnegative and increasing, and since we assume $\beta \geq 1 - 2^{-H(\alpha)}$, we have $g(\beta) - g(1 - 2^{-H(\alpha)}) \geq 0$. Inequality A.8 follows from expanding the definition of $g(\cdot)$ and rearranging.

We now turn to proving the statement of Lemma A.4. Set $\beta = 1 - 2^{-H(\alpha)-C}$ and notice that

$$1 - 2^{-H(\alpha)-C} = \beta \leq \log\left(\frac{1}{1-\beta}\right) - H(\alpha) + \ell_{\mathsf{ag}}(\alpha) = \left(1 - 2^{-H(\alpha)}\right) + C = \ell_{\mathsf{ag}}(\alpha) + C$$

which, after rearranging, recovers the statement of Lemma A.4. ■

**Lemma A.5 (Following McAllester [McA03], page 4)** *Let $\alpha, \beta \in [0,1]$. Then*

$$|\beta - \alpha| \leq \sqrt{2\alpha D_{\mathsf{KL}}(\alpha \parallel \beta)} + 2D_{\mathsf{KL}}(\alpha \parallel \beta).$$

**Proof** We handle the cases $\beta \geq \alpha$ and $\beta \leq \alpha$ separately. We first, consider $\beta \geq \alpha$ and show an upper bound on $\beta$. We will show

$$D_{\mathsf{KL}}(\alpha \parallel \beta) - \frac{(\beta-\alpha)^2}{2\beta} \geq g_\alpha^{(1)}(\beta) \coloneqq D_{\mathsf{KL}}(\alpha \parallel \beta) - \frac{(\beta-\alpha)^2}{(2\ln 2)\,\beta} \geq 0. \tag{A.9}$$

To show (A.9), note that $g_\alpha^{(1)}(\alpha) = 0$ and

$$\forall \beta \geq \alpha \qquad \frac{\partial}{\partial\beta} g_\alpha^{(1)}(\beta) = \frac{1}{\ln 2}\left(\frac{\beta-\alpha}{\beta(1-\beta)} + \frac{\beta-\alpha}{\beta} + \frac{(\beta-\alpha)^2}{2\beta^2}\right) \geq 0. \tag{A.10}$$

Rearranging and left-most expression in (A.9), we obtain the quadratic inequality

$$0 \geq \beta^2 - 2\beta\left(\alpha + D_{\mathsf{KL}}(\alpha \parallel \beta)\right) + \alpha^2. \tag{A.11}$$

We solve (A.11) for $\beta$.

$$\beta \leq \left(\alpha + D_{\mathsf{KL}}(\alpha \parallel \beta)\right) + \sqrt{\left(\alpha + D_{\mathsf{KL}}(\alpha \parallel \beta)\right)^2 - \alpha^2} \tag{A.12}$$

$$= \alpha + D_{\mathsf{KL}}(\alpha \parallel \beta) + \sqrt{2\alpha D_{\mathsf{KL}}(\alpha \parallel \beta) + D_{\mathsf{KL}}(\alpha \parallel \beta)^2} \leq \sqrt{2\alpha D_{\mathsf{KL}}(\alpha \parallel \beta)} + 2D_{\mathsf{KL}}(\alpha \parallel \beta) \tag{A.13}$$

This yields the desired upper bound on $|\beta - \alpha|$ when $\beta \geq \alpha$.

For $\beta \leq \alpha$, we will show

$$g_\alpha^{(2)}(\beta) \coloneqq D_{\mathsf{KL}}(\alpha \parallel \beta) - \frac{(\alpha-\beta)^2}{2\alpha\,(1-\alpha)} \geq 0. \tag{A.14}$$

To show (A.14), we similarly note that $g_\alpha^{(2)}(\alpha) = 0$ and

$$\forall \beta \leq \alpha \qquad \frac{\partial}{\partial \beta} g_\alpha^{(2)}(\beta) = \frac{1}{\ln 2}\left((\alpha-\beta)\left(\frac{1}{\alpha\,(1-\alpha)} - \frac{1}{\beta\,(1-\beta)}\right)\right) \leq 0. \tag{A.15}$$

Now solving A.14 for $\beta$ yields

$$\beta \geq \alpha - \sqrt{2\alpha(1-\alpha) D_{\mathsf{KL}}(\alpha \parallel \beta)} \geq \alpha - \sqrt{2\alpha D_{\mathsf{KL}}(\alpha \parallel \beta)} \tag{A.16}$$

$$\geq \alpha - \sqrt{2\alpha D_{\mathsf{KL}}(\alpha \parallel \beta)} - 2D_{\mathsf{KL}}(\alpha \parallel \beta). \tag{A.17}$$

We now have the lower bound $\beta - \alpha$, which upper bounds $|\beta - \alpha|$ when $\beta \leq \alpha$. We combine this with the case where $\beta \geq \alpha$ to obtain the desired upper bound on $|\beta - \alpha|$ in all cases. ■

**Interpolating training sets with repeated elements.** In Lemma A.6, we show that removing $k^+$ repeated examples with $y_i = 1$,and $k^-$ repeated examples with $y_i = 0$ only reduces the program length guaranteed by Theorem 3.1. Hence, even if the sample has repeated samples, the guarantee from Theorem 3.1 still holds.

**Lemma A.6** *For any $K^-, k^+ \geq 0$ and $m > k^+ + k^-$*

$$\left(m - (k^+ + k^-)\right) H\left(\frac{m - k^+}{m - (k^+ + k^-)}\right) \leq mH\left(\frac{k}{m}\right).$$

**Proof** It is enough to prove the below for any two positive integers $a \leq b$.

$$b \cdot H\left(\frac{a}{b}\right) \leq (b+1) \cdot H\left(\frac{a}{b+1}\right) \tag{A.18}$$

$$b \cdot H\left(\frac{a}{b}\right) \leq (b+1) \cdot H\left(\frac{a+1}{b+1}\right) \tag{A.19}$$

For Inequality A.18, we take the derivative of the function $f_1(a,b) \coloneqq b \cdot H(a/b)$ with respect to $b$ and show that it is always nonnegative. Indeed, the derivative of $f_1(a,b)$ with respect to $b$ is $\log\left(b/(b-a)\right) > 0$. For Inequality A.19, we use $H(x) = H(1-x)$ and Inequality A.18 to write

$$b \cdot H\left(\frac{a}{b}\right) = b \cdot H\left(\frac{b-a}{b}\right) \leq (b+1) \cdot H\left(\frac{b-a}{b+1}\right) = (b+1) \cdot H\left(\frac{a+1}{b+1}\right).$$

This completes the proof of Lemma A.6. ■

## Appendix B. Contrasting to the Inconsistency Result of Grünwald and Langford [GL07]

Grünwald and Langford [GL07] study, among other learning rules inspired by probabilistic modeling, the learning rule $\mathsf{MDL}_2$ defined as

$$\mathsf{MDL}_2(S) \coloneqq \operatorname*{argmin}_{\mathsf{h}\in\mathcal{H}} \quad -\log p(\mathsf{h}) + \log \binom{m}{m \cdot L_S(\mathsf{h})}, \tag{B.1}$$

where $p$ is a discrete "prior" over a countable hypothesis class $\mathcal{H}$ such that $\sum_{\mathsf{h}\in\mathcal{H}} p(\mathsf{h}) = 1$ (as presented by Grünwald and Langford), or $\sum_{\mathsf{h}\in\mathcal{H}} p(\mathsf{h}) \leq 1$ more generally. This can be interpreted as applying the Minimum Description Length model selection criteria (using a standard two-part code) to a conditional probability model $Y|X$ given by parameters $(\mathsf{h}, \alpha)$ where $\Pr[Y = \mathsf{h}(X)|X] = 1 - \alpha$. Instead of a prior $p(\cdot)$, we can (almost) equivalently think of some prefix-unambiguous description language $d : \mathcal{H} \to \{0,1\}^*$, with $p(\mathsf{h}) \coloneqq 2^{-|d(\mathsf{h})|}$ where Kraft's inequality ensures $\sum_{\mathsf{h}\in\mathcal{H}} p(\mathsf{h}) \leq 1$. In our paper we are specifically concerned with descriptions in some Turing complete programming language, but description-length (or equivalently, prior-based) learning rules can be studied with respect to any descriptions (e.g. using decision trees, formulas, linear predictors, or arbitrary artificial descriptions/prior).

Grünwald and Langford show a specific hypothesis class $\mathcal{H}$, prior $p(\cdot)$ and source distribution $\mathcal{D}$ such that $\mathsf{MDL}_2$ is inconsistent, i.e.where $\lim_{m\to\infty} L(\mathsf{MDL}_2) > \inf_{\mathsf{h}\in\mathcal{H}} L(\mathsf{h})$. They contrast this with SRM (which they refer to as ORB), which yields (1.1) and is thus consistent. Our Theorem 2.3 also establishes inconsistency when $L^\star > 0$. But as mentioned in the introduction these inconsistency results are quite different, and here we examine the differences in detail.

**Interpolation.** The two inconsistency results refer to different learning rules, as $\mathsf{MPL}$ is an interpolating rule for which $L_S(\mathsf{MPL}(S)) = 0$ almost surely. On the other hand, $\mathsf{MDL}_2$ balances between the prior, or description length, and the training error, and does not generally interpolate. Our interest here is specifically in interpolating learning.

**Turing-completeness of the priors.** Grünwald and Langford study $\mathsf{MDL}_2$ with respect to a prior over some specific artificial hypothesis class $\mathcal{H}$. The hypothesis class $\mathcal{H}$ they consider is extremely restrictive and specific and has nothing to do with computable functions. Viewing their prior as corresponding to a description language, this description language is extremely far from being Turing complete.

It is easy to see that the interpolating $\mathsf{MDL}$ learning rule

$$\mathsf{MDL}_0(S) \coloneqq \arg\min_{\mathsf{h}\in\mathcal{H}} -\log p(\mathsf{h}) \quad \text{s.t. } L_S(\mathsf{h}) = 0 \tag{B.2}$$

is not consistent for arbitrary hypothesis classes. Consider, for example the hypothesis class $\mathcal{H} = \{\mathsf{h}_i : x \mapsto x[i] \mid i \in \mathbb{N}\}$, and a source distribution where bits of $X$ are i.i.d. uniform, and $Y = X[1] \oplus \mathsf{Ber}(\alpha)$ for any $0 < \alpha < 0.5$. We have that $L(\mathsf{MDL}_0) \overset{m\to\infty}{\to} 0.5 > \inf_{\mathsf{h}\in\mathcal{H}} L(\mathsf{h}) = L(\mathsf{h}_1) = \alpha$.

But our interest here is *not* in arbitrary hypothesis classes, priors or description languages, but rather in interpolation with short programs, i.e. with respect to a Turing complete description language. Theorem 2.3 establishes inconsistency even for a Turing-complete description language, that is for the learning rule $\mathsf{MPL}$, which is an instantiation of $\mathsf{MDL}_0$ for a Turing-complete description

language. Furthermore, and perhaps even more significantly, we show an upper bound on the generalization error of MPL, which rests crucially on Turing completeness (as attested by the example above) and always improves over random guessing. We do not know whether $\mathsf{MDL}_2$ is consistent for a Turing complete description language, and Grünwald and Langford [GL07] do not shed light on this question.

**Well-specification.** Grünwald and Langford show inconsistency only in a misspecified setting, i.e., when the noise with respect to the optimal predictor in their hypothesis class is *not* independent of the input datapoint $X$. This misspecification is crucial, since with random label noise, i.e. when $Y = \mathsf{h}^\star(X) \oplus \mathsf{Ber}(\alpha)$ for some $\mathsf{h}^\star \in \mathcal{H}$, they state $\mathsf{MDL}_2$ *is* consistent. In contrast, Theorem 2.3 establishes inconsistency of the interpolating MPL rule even with random classification noise.

On the positive side, Theorem 2.2 establishes an upper bound on the generalization error of MPL, regardless of the noise model which always (as long as $L^\star < 0.5$) improves over random guessing. Grünwald and Langford provide an agnostic upper bound for a related learning rule (Bayesian averaging), though not for $\mathsf{MDL}_2$, but in contrast to our agnostic guarantee for MPL, their upper bound ensures better-than-random-guessing prediction only when $L^\star < 0.11$.

**Source of the inconsistency – bias versus variance.** To further understand how the two inconsistency results are rather different and stem from different causes, it is instructive to consider a form of bias-variance decomposition and study the "mean" predictor: for a learning rule $A$, source distribution $\mathcal{D}$ and sample size $m$, consider the predictor obtained by taking an expectation over the training set. Since each predictor, on each test point $x$, returns either 0 or 1, we will get a real-value prediction, which is the probability, over the training set, of predicting 1 on test point $x$. We consider majority vote prediction, i.e. predicting by thresholding this probability at $1/2$.

$$\overline{A}(x) = \mathbb{1}\left\{ \Pr_{S\sim\mathcal{D}^m}\left[A(S)(x) = 1\right] \geq 0.5 \right\} \tag{B.3}$$

We can then think of the "bias" of $A$ as the error $L(\overline{A})$ of the mean predictor.

For MPL with random label noise, i.e. when $Y = \mathsf{h}^\star(X) \oplus \mathsf{Ber}(L^\star)$ with $L^\star < 0.5$ we have that $\overline{\mathsf{MPL}}(x) \overset{m\to\infty}{\to} \mathsf{h}^\star$. There is no bias, and the inconsistency stems from non diminishing "variance." That is, $\Pr_{S\sim\mathcal{D}^m}\left[\mathsf{MPL}(S) = \overline{\mathsf{MPL}}\right]$ remains bounded away from zero even as $m \to \infty$. On the other hand, one can verify that in the inconsistency example for $\mathsf{MDL}_2$ that Grünwald and Langford provide (in a non-well-specified setting), the "variance" vanishes (that is, $\Pr_{S\sim\mathcal{D}^m}\left[\mathsf{MDL}_2(S)(x) = \overline{\mathsf{MDL}_2}(x)\right] \to 0$). However, the *bias* does not vanish and we have that

$$\liminf_{m\to\infty} L(\overline{\mathsf{MDL}_2}) > 0.$$