Proceedings of Machine Learning Research vol 178:1–25, 2022 35th Annual Conference on Learning Theory

Monotone Learning

Olivier Bousquet OBOUSQUET@GOOGLE.COM
Google, Zürich

Amit Daniely AMIT.DANIELY@MAIL.HUJI.AC.IL
School of Computer Science and Engineering, The Hebrew University and Google Research

Haim Kaplan HAIMK@TAU.AC.IL
Blavatnik school of computer science, Tel Aviv University and Google Research

Yishay Mansour MANSOUR.YISHAY@GMAIL.COM
Blavatnik school of computer science, Tel Aviv University and Google Research

Shay Moran SMORAN@TECHNION.AC.IL
Departments of Mathematics and Computer Science, Technion
Google Research, Tel-Aviv

Uri Stemmer U@URI.CO.IL
Blavatnik school of computer science, Tel Aviv University and Google Research

Editors: Po-Ling Loh and Maxim Raginsky

Abstract

The amount of training-data is one of the key factors which determines the generalization capacity of learning algorithms. Intuitively, one expects the error rate to decrease as the amount of training-data increases. Perhaps surprisingly, natural attempts to formalize this intuition give rise to interesting and challenging mathematical questions. For example, in their classical book on pattern recognition, Devroye, Györfi, and Lugosi (1996) ask whether there exists a monotone Bayes-consistent algorithm. This question remained open for over 25 years, until recently Pestov (2021) resolved it for binary classification, using an intricate construction of a monotone Bayes-consistent algorithm.

We derive a general result in multiclass classification, showing that *every* learning algorithm A can be transformed to a monotone one with similar performance. Further, the transformation is efficient and only uses a black-box oracle access to A. This demonstrates that one can provably avoid non-monotonic behaviour without compromising performance, thus answering questions asked by Devroye, Györfi, and Lugosi (1996), Viering, Mey, and Loog (2019), Viering and Loog (2021), and by Mhammedi (2021).

Our general transformation readily implies monotone learners in a variety of contexts: for example, Pestov's result follows by applying it on *any* Bayes-consistent algorithm (e.g., k-Nearest-Neighbours). In fact, our transformation extends Pestov's result to classification tasks with an *arbitrary* number of labels. This is in contrast with Pestov's work which is tailored to binary classification.

In addition, we provide *uniform bounds* on the error of the monotone algorithm. This makes our transformation applicable in distribution-free settings. For example, in PAC learning it implies that *every* learnable class admits a monotone PAC learner. This resolves questions asked by Viering, Mey, and Loog (2019); Viering and Loog (2021); Mhammedi (2021).

Keywords: Learning curve, Monotonicity, Bayes consistency, PAC learning

1. Introduction

In this work we study the following fundamental question. Pick some standard learning algorithm A, and consider training it for some natural task using a data-set of n examples.

Does feeding A with more training data *provably* reduces its population loss?

E.g., would increasing the number of examples from n to $n+1$ improve its loss? How about $2n$? Can one guarantee improvement in this case? What about 2^n, or even 2^{2^n}? Would that be sufficient? Can one at least assure that the loss will not deteriorate?

Intuitively, the answer should be yes: indeed, the more often we face a certain task, the better we typically get at solving it. This basic intuition is reflected in many works in theoretical and applied machine learning. For example, Shalev-Shwartz and Ben-David (2014) assert in their book that the learning curve starts decreasing when the number of examples surpasses the VC dimension (page 153); Duda, Hart, and Stork (2001) state in their book that for real-world problems the learning curve is monotone (Subsection 9.6.7). Similar statements are made by a variety of other works, a partial list includes Gu, Hu, and Liu (2001); Tax and Duin (2008); Weiss and Battistin (2014). We refer the reader to the thorough survey by Viering and Loog (2021) for an extensive discussion about monotone and non-monotone learning curves (Section 6).

On the other hand, one might argue that in order to successfully learn complex functions, the algorithm must dedicate time and resources to exploring larger and larger sets of hypotheses, and consequently exhibit a non-monotone behaviour. For example, any Bayes consistent learning algorithm must consider arbitrarily complex hypotheses (since it is able to approximate arbitrary functions). Indeed, one can show that consistent algorithms such as *Nearest-Neighbors* can demonstrate such non-monotone behaviour (Devroye, Györfi, and Lugosi, 1996). This intuition is reflected in Chapter 6 in the book by Devroye, Györfi, and Lugosi (1996) in which it is conjectured that no Bayes-consistent rules can be monotone (Problem 6.16).

These intuitive considerations inspire a host of theoretical questions. Is it really the case that "more data = better generalization"? Is it at least the case for natural algorithms and natural learning tasks? Perhaps it is too much to expect that the addition of a single example will lead to better performance, but maybe if one doubles the training-set then better performance is guaranteed? Can we at least guarantee that the performance does not deteriorate?

Notation. We focus on multiclass classification with respect to the zero/one loss and use standard learning theoretic notation (see, e.g., Shalev-Shwartz and Ben-David (2014)). Let X be a set called the domain and let Y denote the label-set. We assume that Y is finite, w.l.o.g $Y = [k] = \{0, 1, \ldots, k-1\}$. For a set Z, let $Z^\star := \cup_{n=0}^{\infty} Z^n$ denote the space of all finite sequences with elements from Z. An *hypothesis (or classifier)* is a function $h : X \to Y$. An *example* is a pair $z = (x, y) \in X \times Y$. A *sample* $S \in (X \times Y)^\star$ is a (finite) sequence of examples. A *learning rule* is a mapping from $(X \times Y)^\star$ to Y^X; i.e., the input is a finite sample and the output is a hypothesis. Given a distribution D over $X \times Y$ and a hypothesis h, the *(population) loss* of h with respect to D is $\mathrm{L}_D(h) = \mathbb{E}_{(x,y)\sim D}\, 1[h(x) \neq y]$. Given a sample $S = \{(x_i, y_i)\}_{i=1}^m$, the (empirical) loss of h with respect to S is $\hat{\mathrm{L}}_S(h) = \frac{1}{m}\sum_{i=1}^m 1[h(x_i) \neq y_i]$, where $1[\cdot]$ is the indicator function.

MONOTONE LEARNERS

Definition 1 (Monotone learning rule) *A learning rule M is said to be* monotone *w.r.t a distribution D if,*

$$(\forall m) : \mathop{\mathbb{E}}_{S\sim D^m}\Big[\mathrm{L}_{\mathrm{D}}\big(M(S)\big)\Big] \geq \mathop{\mathbb{E}}_{S\sim D^{m+1}}\Big[\mathrm{L}_{\mathrm{D}}\big(M(S)\big)\Big].$$

That is, the expected population loss of M is monotone non-decreasing in the size of its training set.

The following theorem is the main result in this work; it asserts that every learning algorithm can be efficiently transformed to a monotone one with competitive generalization guarantees.

Theorem 2 *[Every learner can be monotonised] Consider the setting of multiclass classification to $k \in \mathbb{N}$ labels. Then, every learning algorithm A can be efficiently converted to a learning algorithm $M = M(A)$ such that M has only a black-box oracle access to A and*

1. *M is monotone with respect to* every *distribution D.*
2. *M's performance is* competitive *with that of A: for every source distribution D,*

$$\Big(\forall m\Big)\Big(\exists m' \text{ s.t. } \frac{m}{30} - 1 \leq m' \leq m\Big) : \mathop{\mathbb{E}}_{S\sim D^m}\Big[\mathrm{L}_{\mathrm{D}}\big(M(S)\big)\Big] \leq \mathop{\mathbb{E}}_{S\sim D^{m'}}\Big[\mathrm{L}_{\mathrm{D}}\big(A(S)\big)\Big] + O\Big(\sqrt{\frac{\log m}{m}}\Big).$$

Theorem 2 affirmatively answers questions posed by Devroye, Györfi, and Lugosi (1996), Viering, Mey, and Loog (2019), Viering and Loog (2021), and Mhammedi (2021). In addition, Theorem 2 readily implies monotone learners in a variety of contexts:

Corollary 3 (Bayes-Consistent Monotone Learners) *For every $k \in \mathbb{N}$ there exists a Bayes consistent monotone learner for multiclass classification into k labels.*

Indeed, this follows by applying the transformation on *any* Bayes-consistent learner (for example, k-nearest neighbor). This extends Pestov's result who focused on the case of binary classification and designed a clever histogram-based Bayes consistent algorithm. Moreover, while Pestov's algorithm and analysis are tailored to the binary case, our argument is more general, and at the same time conceptually (and arguably technically) simpler. The existence of Bayes-optimal consistent learner remained open for 25 years since it was asked by by Devroye, Györfi, and Lugosi (1996).

Theorem 2 is also applicable in other contexts. In fact, the distribution-free regret-bound on the learning rate of the monotone learner allows one to apply it in the PAC setting, where monotone learners were not known to exist Viering, Mey, and Loog (2019); Viering and Loog (2021):

Corollary 4 (Monotone PAC Learners) *Let $k \in \mathbb{N}$ and let $\mathcal{H} \subseteq [k]^X$ be a PAC learnable hypothesis class. Then, there exists a monotone agnostic PAC learner for $\mathcal{H}$.*

Indeed, this follows by applying the transformation on any PAC learning algorithm for $\mathcal{H}$ (say any empirical risk minimizer). Note that the learning rate of the resulting monotone PAC learner is suboptimal by an additive $\log m$ factor[1]. We leave the exploration for the optimal monotone PAC learning rate to future work.

1. The optimal PAC learning rate is proportional to $\sqrt{1/m}$.

1.1. Informal Explanation

A learning rule, even if it is Bayes consistent, does not have any reason, a priori, to be monotone. Indeed, the fact that the expected error converges to the Bayes error does not mean the convergence happens monotonically and it could very well be that the error strictly increases between m and $m+1$ infinitely often. Let us examine what are the difficulties one would encounter when trying to convert a Bayes consistent learner into one that is monotone.

Convergent Learners are Sparsely Monotone. Given any learning algorithm A that has the following convergence property:

$$\lim_{m\to\infty} \mathop{\mathbb{E}}_{s\sim D^m} [\mathrm{L}_{\mathrm{D}}(A(S))] = e_D,$$

with $\mathbb{E}_{s\sim D^m} [\mathrm{L}_{\mathrm{D}}(A(S))] \geq e_D$ for infinitely many m, which for example happens for any Bayes-consistent learner, we immediately get that there exists a subsequence $m_1, \ldots, m_k, \ldots$ of indices over which the error will be monotonically decreasing, i.e.,

$$\forall j \geq i, \quad \mathop{\mathbb{E}}_{s\sim D^{m_j}} [\mathrm{L}_{\mathrm{D}}(A(S))] \leq \mathop{\mathbb{E}}_{s\sim D^{m_i}} [\mathrm{L}_{\mathrm{D}}(A(S))] .$$

However, the above subsequence of indices is *distribution-dependent*, which means that we cannot guarantee that if we increase the sample size from some value m to some other value m', the error will be smaller for all distributions.

Sparse to Dense Monotonicity. We first observe that we could relax the monotonicity requirement to hold only for infinitely many steps (or arbitrary size). Indeed, while the monotonicity requirement of Definition 1 is written for m and $m+1$, we observe that if we had a learner that satisfies a *sparse* version of this inequality, such as

$$(\forall m)(\exists m' \geq m) : \mathop{\mathbb{E}}_{S\sim D^m}\Big[\mathrm{L}_{\mathrm{D}}\big(M(S)\big)\Big] \geq \mathop{\mathbb{E}}_{S\sim D^{m'}}\Big[\mathrm{L}_{\mathrm{D}}\big(M(S)\big)\Big], \tag{1}$$

it would be easy to convert it into a learner that satisfies Definition 1 without affecting the limit of its population loss as $m \to \infty$. Indeed, the above condition guarantees that there is an infinite sequence $m_1, \ldots, m_k, \ldots$ of indices over which the algorithm is guaranteed not to increase its expected loss. This sequence can be defined as $m_1 = 1$ and $m_{k+1} = m'(m_k)$. And from this, one could create a learner that, given m examples with $m \in [m_k, m_{k+1})$, simply ignores $m - m_k$ examples from the training set. The monotonicity condition would then be satisfied with equality between m_k and m_{k+1} since the output of our algorithm would be unchanged (in expectation).

Running over Prefixes. So in order to convert an arbitrary learning rule into one that is monotone, while still retaining its convergence properties, the main idea is to run the algorithm on prefixes of the training sample and measure the loss of the produced hypotheses in order to pick the best one. As the sample size increases, the pool of hypotheses to choose from will increase and the best one from a larger pool will thus have a smaller loss than from a smaller pool. This idea has been previously explored by Viering, Mey, and Loog (2020); Mhammedi (2021).

However, implementing this idea turns out to be a subtle task. Indeed, we can only *estimate* the loss of the hypotheses (by setting aside some examples and computing their empirical loss), so there is always some possibility that we choose a worse hypothesis (which empirically looks better).

To illustrate the issue, let's consider the simplest possible situation where the base algorithm has produced a hypothesis h_0 on a prefix of the sample, and another hypothesis h_1 on a longer prefix. If $\mathrm{L}_{\mathrm{D}}(h_1) \leq \mathrm{L}_{\mathrm{D}}(h_0)$, the base algorithm is already monotone, but in the case $\mathrm{L}_{\mathrm{D}}(h_1) > \mathrm{L}_{\mathrm{D}}(h_0)$, any wrapper algorithm would have to choose between outputting h_0 or h_1. Unfortunately, this choice will necessarily worsen the error (unless the wrapper always outputs h_0 deterministically, in which case it would not manage to track the performance of the base algorithm). Indeed, the expected loss of any wrapper would be a convex combination of $\mathrm{L}_{\mathrm{D}}(h_1)$ and $\mathrm{L}_{\mathrm{D}}(h_0)$ and would be strictly larger than $\mathrm{L}_{\mathrm{D}}(h_0)$.

So we cannot simply take the output of the base algorithm, and the idea is to *regularize* it, i.e., make it possibly a little worse but in such a way that this regularization can be reduced as the sample size increases and thus we can guarantee monotonicity.

How to Make the Learner Worse? The question thus becomes: given a hypothesis h, is there a way to produce a hypothesis h' that is guaranteed to be *worse* than h (i.e., $\mathrm{L}_{\mathrm{D}}(h') > \mathrm{L}_{\mathrm{D}}(h)$) and to possibly control how much worse it is? The first idea that comes to mind is to sometimes output a label which we know is incorrect. This would be possible if we add some extra label at our disposal, e.g., in binary classification we would allow the learner to output something different from 0 or 1, say $\perp$ and count $\perp$ as a mistake. But this is somewhat artificial and would require to change the nature of the algorithm's predictions. So the second idea that comes to mind is to just add noise to the output of the algorithm, i.e., to randomly pick a different label than the one predicted. Unfortunately, in the context of classification, adding noise does not guarantee that the loss is made worse! Indeed, in binary classification, if $\mathrm{L}_{\mathrm{D}}(h) > 1/2$, adding some uniform noise to the output would make the error closer to $1/2$, hence better and not worse.

How to Make the Learner Always Better Than Some Value? So we see that if we could, given a hypothesis h, which could have error larger than $1/2$, return one that is guaranteed to have error less than $1/2$, we could then make the latter worse by adding uniform noise. This brings us to our last key idea: we symmetrize the output by replacing the hypothesis produced by the base algorithm by the best (in terms of empirical error) between h and $1 - h$. We can then guarantee (we will prove it below) that the expected loss of this *symmetrized* output is less than $1/2$ and adding noise will thus strictly increase its loss, making room for reducing the loss when we choose between h_0 and h_1.

Symmetrization is more subtle in the context of multiclass classification with $k > 2$ labels; the idea there is to replace h with the best out of k hypotheses which are obtained by composing h with a cyclic permutation of the labels. For simplicity, we focus on the binary-case in this outline.

Formalization. Let us now try and write down some of the ideas above more formally. The overall approach is to run the base algorithm on a prefix of size n of the sample S to obtain some hypothesis h_0, apply some transformation (which we call *regularization*) to h_0 which consists of symmetrizing and adding noise, in order to obtain $R(h_0, S)$. Then perform the same operation on a longer prefix of size $N > n$ of S to obtain $R(h_1, S)$ and then decide whether to use h_0 or h_1 by estimating their respective errors (on an additional subset of N examples from S). If we denote by p_N the probability of choosing h_1 over h_0, we see that the expected error of the resulting procedure will have the form[2]

$$p_N \mathop{\mathbb{E}}_{S\sim D^N}\Big[\mathrm{L}_{\mathrm{D}}\big(R(h_1, S)\big)\Big] + (1 - p_N) \mathop{\mathbb{E}}_{S\sim D^N}\Big[\mathrm{L}_{\mathrm{D}}\big(R(h_0, S)\big)\Big]$$

2. We will later provide more details about how to split the training sample so as to guarantee independence and decouple the expecations appropriately.

and if we want to satisfy Inequality (1), this quantity would have to be smaller than the expected error of our procedure ran on n examples, i.e., we would want

$$p_N \mathop{\mathbb{E}}_{S\sim D^N}\left[\mathrm{L}_D\big(R(h_1,S)\big)\right] + (1-p_N)\mathop{\mathbb{E}}_{S\sim D^N}\left[\mathrm{L}_D\big(R(h_0,S)\big)\right] \le \mathop{\mathbb{E}}_{S\sim D^n}\left[\mathrm{L}_D\big(R(h_0,S)\big)\right]$$

As discussed above, without regularization, i.e., if R is the identity, there is no way to guarantee this inequality for every pair h_0, h_1, and the problematic case is when h_1 is worse than h_0, or when $R(h_1, S)$ is worse than $R(h_0, S)$. If we rewrite the above condition as follows:

$$p_N\left(\mathop{\mathbb{E}}_{S\sim D^N}\left[\mathrm{L}_D\big(R(h_1,S)\big)\right] - \mathop{\mathbb{E}}_{S\sim D^N}\left[\mathrm{L}_D\big(R(h_0,S)\big)\right]\right) \le \mathop{\mathbb{E}}_{S\sim D^n}\left[\mathrm{L}_D\big(R(h_0,S)\big)\right] - \mathop{\mathbb{E}}_{S\sim D^N}\left[\mathrm{L}_D\big(R(h_0,S)\big)\right]$$

we see that in order for it to be satisfied even when $R(h_1, S)$ is worse than $R(h_0, S)$, we need

1. p_N has to be small enough (to make the left hand side small enough).
2. The regularization over N examples has to be strictly better than the regularization over n examples (to make the right hand side positive and large enough).

1.2. Technical Contributions

The technical contribution in this work can roughly be partitioned to two parts:

(i) We develop a general axiomatic framework for constructing transformations which compile arbitrary learners to monotone learners with similar guarantees (Section 2). We attempt to state this framework in an abstract manner with the hope that it might be useful for other loss functions.

In a nutshell, this framework reduces the task to constructing for every hypothesis h a *small* and *symmetric* class B_h such that $h \in B_h$, and B_h can be learned by a monotone learner; for example, in the context of binary classification ($Y = \{0,1\}$) we use $B_h = \{h, 1-h\}$. More generally, in the context of multiclass classification ($Y = [k] = \{0,\ldots,k-1\}$), we use $B_h = \{s_i \circ h : i \in [k]\}$, where s_i is the cyclic permutations mapping a label y to $y+i \mod k$.

(ii) In Sections B and C we use our general framework to prove Theorem 2. In Section B we focus on the case of binary classification; this section serves as a warmup to the general multiclass setting which is considered in Section C.

The most technical proof in this work is that of Proposition 9, specifically Lemma 16 which asserts that the randomized ERM over B_h is monotone: Recall that for a hypothesis $h : X \to \{0,\ldots,k-1\}$, the class B_h consists of the k cyclic permutations of h: $B_h = \{s_i \circ h : i \in [k]\}$, where s_i is a cyclic permutation mapping $y \mapsto y+i \mod k$. The randomized ERM is the algorithm which given an input sample S, outputs an empirical risk minimizer from B_h which is drawn uniformly at random.

To prove Proposition 9 we exploit the following symmetry exhibited by B_h: for any example (x, y) there exists a unique $h' \in B_h$ such that $h'(x) = y$. This implies, via a symmetrization argument and via Chebychev's sum inequality[3] the desired monotonicity (Hardy, Littlewood, and Pólya, 1988).

3. Chebyshev's sum inequality asserts that if $a_1 \le \ldots \le a_n$ and $b_1 \ge \ldots \ge b_n$ then $\frac{1}{n}\sum a_i \cdot \frac{1}{n}\sum b_i \ge \frac{1}{n}\sum a_i b_i$.

We also note that the upper bound on the rate in Theorem 2 is *independent* of the number of labels k. To achieve this we once again appeal to the symmetric structure of B_h, and show that B_h satisfies uniform convergence with rate which is independent of k.

1.3. Related Work

The idea of monotone learning curves for universally consistent learners was first discussed by Devroye, Györfi, and Lugosi (1996).This problem attracted little attention until recently when Viering, Mey, and Loog (2019) considered monotone learning in a variety of contexts (e.g., when the goal is to learn a fix hypothesis class) and under more general loss functions. Viering, Mey, and Loog (2020), Viering and Loog (2021), and Mhammedi (2021) considered the problem of transforming a given learner to a monotone one using a *wrapper algorithm*. Viering, Mey, and Loog (2020) and Mhammedi (2021) derive weaker forms of monotonicity and leave open the question of whether such a transformation exists. In this work we resolve this problem in the context of multiclass classification.

The conjecture by Devroye et al. (1996) was finally answered in the positive by Pestov (2021). Pestov's result applies to binary classification, and here we prove an extension to general multiclass classification.

Other Notions of Mononoticity. Viering, Mey, and Loog (2020) proposes to relax the requirement of monotonicity in expectation into *high-probability* and *eventual monotonicity*. Viering, Mey, and Loog (2020) and Mhammedi (2021) also discuss the relationship with the multiple descent phenomenon established for many learners in recent years.

Other Notions of Consistency. It is important to note that the open problem proposed by Viering et al. (2019) is concerning consistency with respect to a fixed class of function. So this is less general than the universal consistency of (Devroye, Györfi, and Lugosi, 1996).

Comparing with Pestov's result. While Pestov (2021) just builds a specific algorithm and not a generic wrapper, and considers only the binary classification case, there are some similarities between his approach and ours that are worth illustrating. Indeed, his algorithm consists in the following three ingredients

1. Consider prefixes of the input sample of (exponentially) increasing size
2. Perform a majority vote over a partition of the input domain
3. Decide (empirically) whether or not to split each element of the partition into smaller pieces

The first ingredient is similar to our (and other's) approach of guaranteeing monotonicity on an infinite sequence of indices (what we call sparse monotonicity above), the second one bears some similarity with our symmetrization approach since the majority vote consists in comparing h and $1 - h$, and the last one is comparable with our update procedure which decides whether to continue using h_0 or to switch to h_1.

However there is one important difference which is key to obtaining a uniform bound on the excess loss of our monotone algorithm. Indeed, Pestov does not regularize by adding noise which requires him to refine the partition element under some very restrictive conditions (the conditional loss on the partition should not be close to $1/2$ nor to 0 or 1) which has the effect of requiring

to make a very large increases of the sample size between two stages, resulting in a slower, non-uniform, convergence rate (more specifically, he does not provide an explicit formula for computing N from n).

2. General Framework

Given an algorithm A which maps an input sample S to an output hypothesis $A(S)$, we construct a monotone algorithm M using two intermediate algorithms:

1. A *regularization* algorithm R is an algorithm that takes as input a sample S and a hypothesis h and returns a (possibly randomized) hypothesis $R(h, S)$. It might be useful/intuitive to think about $R(h, S)$ as a smooth/regularized version of h.

2. An *update* algorithm U is an algorithm that takes as an input a sample S and two hypotheses: (i) h_0 which is called the *current* hypothesis, and (ii) h_1 which is called the *candidate* hypothesis. The algorithm then outputs a hypothesis denoted by $U(h_0, h_1, S) \in \{h_0, h_1\}$. Intuitively, U chooses whether to replace the current hypothesis h_0 with the candidate hypothesis h_1, when the latter has smaller error.

The monotone algorithm will then be constructed in an iterative manner from A by applying A to prefixes of the training sample of increasing size and using the *update* algorithm at each step to decide whether to keep the current hypothesis or to update it to the new one (built on a longer prefix). At the end, we output a regularized version of the currently chosen hypothesis.

The *update* algorithm ensures that with high probability we update the hypothesis only when the new hypothesis has better (smaller) loss than the previous one. But since there is still a small chance of updating to a worse hypothesis, the regularization step will be used to correct for corresponding additional expected loss.

See Figure 1 for a more precise description of the algorithm M.

A Framework for Proving Monotonicity. We introduce several conditions on the update and regularization algorithms which guarantee the success of Algorithm M when applied to *any* learning algorithm A. We then analyze our algorithms by showing that they satisfy these conditions. Let us begin by introducing some notation: given a hypothesis h, denote by $\mathtt{LR}_n(h)$ the quantity

$$\mathtt{LR}_n(h) := \mathop{\mathbb{E}}_{S\sim D^n} \mathrm{L}_{\mathrm{D}}\, R(h, S) \tag{2}$$

where the expectation is with respect to the sample S of size n.

Definition 5 (Sufficient Conditions for Monotonicity) *Let R be a regularization algorithm, let U be an update algorithm. We say that* (R, U) are successful *if for every source distribution D the following conditions are satisfied:*

(C1) *After regularization, the expected loss of the update algorithm is* non-increasing*:*

$$(\forall n \in \mathbb{N})(\exists N \geq n)(\forall h_0, h_1) : \mathop{\mathbb{E}}_{S\sim D^N} \mathtt{LR}_N(U(h_0, h_1, S)) \leq \mathtt{LR}_n(h_0).$$

For $n \in \mathbb{N}$, let $N(n)$ denote the smallest number $N > n$ for which the above holds.

The Monotonizing Algorithm M

- Input: a learning algorithm A, a regularization algorithm R, an update algorithm U, an increasing function $b : \mathbb{N} \to \mathbb{N}$ such that $b(0) = 1$, and a sample $S \sim D^m$ of iid examples from D.
- Output: a hypothesis determined as follows
  1. If $m < 2 \cdot b(1) + b(0)$ then output $R(f, Z)$, where $f = A(\emptyset)$ and Z consists of the first example from S.
  2. Else, let $T \geq 2$ be maximal such that $b(T-1) + \sum_{t=0}^{T-1} b(t) \leq m$; discard from S the last $m - \left(b(T-1) + \sum_{t=0}^{T-1} b(t)\right)$ examples.
  3. Partition the remaining examples into $T+1$ blocks $\{B_t\}_{t=0}^{T}$ where $|B_t| = b(t)$ for $t \leq T-1$ and $|B_T| = b(T-1)$.
  4. Let S_t denote the union of the first t blocks: $S_t := \bigcup_{i=0}^{t} B_i$.
  5. Set $f_0 := A(\emptyset)$.
  6. For each $t = 1, \ldots, T-1$ perform the following operations:
     (a) Compute the new candidate hypothesis using A: $h_t := A(S_{t-1})$.
     (b) Choose the new hypothesis between f_{t-1}, h_t: $f_t := U(f_{t-1}, h_t, B_t)$.
  7. On the last block, output $R(f_{T-1}, B_T)$.

Figure 1: Pseudo-code for the transformation M which converts any learning rule A to a monotone one with similar guarantees. The algorithm proceeds by running A on increasing prefixes of the input sample and apply carefully tailored model selection from the outputs of A. Item 1 handles a trivial base-case and can be ignored at first read. Note that the last two blocks have identical size $|B_{T-1}| = |B_T| = b(T-1)$. This is because the last block serves for regularization, while the first $T-1$ blocks are used for training and model selection. Also notice that the regularizer is applied only on the last iteration, and not on each iteration. This is because the noise is not needed when one compares different hypotheses. Its role is to slightly deteriorate the loss of the output in order to compensate for future mistakes.

(C2) *The* update *algorithm competes with the new hypothesis at a small cost:*

$$(\forall n \in \mathbb{N})(\forall h_0, h_1) : \mathop{\mathbb{E}}_{S\sim D^n} \mathtt{LR}_n(U(h_0, h_1, S)) \leq \mathrm{L}_{\mathrm{D}}(h_1) + c(n),$$

for some function c such that $\lim_{n\to\infty} c(n) = 0$.

Proposition 1 (Monotonicity and Competitiveness of M) *Let R, U be a regularization and update algorithm such that (R, U) are successful. Define $b : \mathbb{N} \to \mathbb{N}$ according to the recurrence $b(0) = 1$ and $b(t+1) = N(b(t))$, where $N(\cdot)$ is the function defined in Condition (C1) of Definition 5. Consider algorithm M with (R, U) and b as inputs. Then, for every algorithm A, applying M on A yields a monotone algorithm whose performance is competitive with that of A: for every source distribution D and every sample size $m \geq 2b(1) + b(0)$,*

$$\mathop{\mathbb{E}}_{S\sim D^m} \mathrm{L}_{\mathrm{D}}(M(S)) \leq \mathop{\mathbb{E}}_{S\sim D^m} \mathrm{L}_{\mathrm{D}}(A(S_{T-2})) + c(b(T-1)),$$

where T and S_{T-2} are as in the pseudo-code of M in Figure 1.

Proof The case of $m < 2b(1) + b(0)$ is trivial, so we assume that $m \geq 2b(1) + b(0)$. We start by proving monotonicity. We only need to consider the case where a new block is added (otherwise, the additional examples are simply discarded and thus the expected error is unchanged). In this case, it is sufficient to consider the effect of adding a new block B_T and prove that

$$\mathbb{E}\,\mathtt{LR}_N(f_{T-1}) \leq \mathbb{E}\,\mathtt{LR}_n(f_{T-2}), \tag{3}$$

for $n = b(T-2)$ and $N = N(n) = b(T-1)$.

Conditioned on S_{T-2}, the hypotheses f_{T-2} and h_{T-1} are fixed and

$$f_{T-1} = U(f_{T-2}, h_{T-1}, B_{T-1})$$

is a function of B_{T-1} (which is not under conditioning). Therefore,

$$\begin{aligned}\mathbb{E}\left[\mathtt{LR}_N(f_{T-1})\middle|S_{T-2}\right] &= \mathop{\mathbb{E}}_{B_{T-1}\sim D^N}\left[\mathtt{LR}_N(U(f_{T-2}, h_{T-1}, B_{T-1}))\middle|S_{T-2}\right] \\ &\leq \mathbb{E}\left[\mathtt{LR}_n(f_{T-2})\middle|S_{T-2}\right],\end{aligned}$$

where the last inequality follows by of Condition (C1)[4]. Equation (3) now follows by taking expectation over S_{T-2}.

For the second part, observe that Condition (C2) implies that

$$\begin{aligned}\mathop{\mathbb{E}}_{S\sim D^m} \mathrm{L}_{\mathrm{D}}(M(S)) &= \mathbb{E}\,\mathrm{L}_{\mathrm{D}}\, R(f_{T-1}, B_T) \\ &= \mathbb{E}\,\mathtt{LR}_N\left(U(f_{T-2}, h_{T-1}, B_{T-1})\right) \\ &\leq \mathbb{E}\,\mathrm{L}_{\mathrm{D}}(h_{T-1}) + c(N) && \text{(Condition (C2))} \\ &= \mathbb{E}\,\mathrm{L}_{\mathrm{D}}(A(S_{T-2})) + c(b(T-1)).\end{aligned}$$

■

4. Notice that we apply (C1) here while conditioning on S_{T-2}. This is valid because B_{T-1} is independent of S_{T-2} and therefore Condition (C1) applies for every fixing of S_{T-2}.

2.1. The Base-Class Approach

We design two algorithms using our framework above, one in binary classification which serves as a "warmup" and a more general one in multiclass classification. In this subsection we describe a common abstraction of these two algorithms. We hope this abstraction will be useful in other contexts as well. (E.g., other loss functions.)

The common abstraction boils down to assuming that every hypothesis h has an associated *simple* hypothesis class B_h such that $h \in B_h$. For example in the context of binary classification ($Y = \{0, 1\}$) our B_h will consist of two hypotheses: h and its negation $1 - h$; i.e., $B_h = \{h, 1 - h\}$. More generally in multiclass classification with k labels our B_h will consist of k hypotheses.

We require that B_h is "well-behaved" in a precise sense which we next describe. Consider the following randomized empirical risk minimizer over B_h.

Algorithm G: Randomized Empirical Risk Minimization

- Input: a hypothesis $h \in Y^X$ and a sample $S \in (X \times Y)^n$
- Set $B_h^\star = \{f \in B_h : \hat{\mathrm{L}}_{\mathrm{S}}(f) = \min_{g \in B_h} \hat{\mathrm{L}}_{\mathrm{S}}(g)\}$
- Output: a uniformly random hypothesis from $B_h^\star$, which is denoted by $G(h, S)$.

In words, $G(h, S)$ is a random empirical risk minimizer in B_h.

Note that on the empty sample, $G(h, \emptyset)$ is a random hypothesis drawn uniformly from B_h.

Let $\mathtt{LG}_n(h)$ denote the expected loss of $G(h, S)$ where S is of size n:

$$\mathtt{LG}_n(h) := \mathop{\mathbb{E}}_{S \sim D^n} \mathrm{L}_{\mathrm{D}}(G(h, S)) \,.$$

Definition 6 (Successful Base-Class) *Let $h \mapsto B_h$ be a mapping which associates with every hypothesis h a finite hypothesis class B_h such that $h \in B_h$. This mapping is called* successful *if the following properties are satisfied:*

1. *The loss of the randomized ERM over B_h is monotone*

$$(\forall D)(\forall h)(\forall n) : \mathtt{LG}_{n+1}(h) \leq \mathtt{LG}_n(h) \,.$$

2. *There exists a function $e(n)$ with $\lim_{n\to\infty} e(n) = 0$ such that every B_h satisfies uniform convergence with rate $e(n)$:*

$$(\forall D)(\forall h)(\forall n) : \mathop{\mathbb{E}}_{S \sim D^n}\left[\max_{f \in B_h} \left|\hat{\mathrm{L}}_{\mathrm{S}}(f) - \mathrm{L}_{\mathrm{D}}(f)\right|\right] \leq e(n).$$

3. *The loss of the random ERM on the empty sample is independent of h and of the source distribution D:*

$$(\forall D)(\exists \alpha)(\forall h) : \mathtt{LG}_0(h) = \alpha \,.$$

In other words, the expected loss of a uniform random hypothesis from B_h is equal to a universal constant α which does not depend on h. (α can depend on the source distribution D.)

Remark 7 *The first condition above could be relaxed into* $\mathtt{LG}_N(h) \leq \mathtt{LG}_n(h)$ *for some* $N \geq n$ *large enough. Indeed this would be sufficient to derive condition (C1). But we will show that the randomized ERM that we consider actually satisfies the stronger property of monotonicity for* $N = n + 1$.

Remark 8 *The second item above implies (via a triangle inequality) that the randomized ERM* $G(h, \cdot)$ *is competitive with* h*:*

$$(\forall D)(\forall h)(\forall n) : 0 \leq \mathtt{LG}_n(h) - \min_{f \in B_h} \mathrm{L}_{\mathrm{D}}(f) \leq 2e(n). \tag{4}$$

THE REGULARIZATION AND UPDATE ALGORITHMS

We next describe how a successful map $h \mapsto B_h$ yields a successful pair of regularization and update algorithms.

The Regularization Algorithm R

- Input: a hypothesis $h \in Y^X$, a sample $S \in (X \times Y)^n$.
- Output: with probability $1-\eta_n$ output $G(h, S)$ and with probability η_n output $G(h, \emptyset)$. Here $\eta_n \in (0, 1)$ is a decreasing function of n satisfying $\lim_{n\to\infty} \eta_n = 0$.

The Update Algorithm U

- Input: two hypotheses $h_0, h_1 \in Y^X$, a sample $S \in (X \times Y)^n$.
- Compute $\min_{f \in B_{h_0}} \hat{\mathrm{L}}_S(f)$ and $\min_{f \in B_{h_1}} \hat{\mathrm{L}}_S(f)+\epsilon_n$, and output h_0 if the first quantity is smaller and h_1 otherwise. Here $\epsilon_n \in (0, 1)$ is a decreasing function of n satisfying $\lim_{n\to\infty} \epsilon_n = 0$.

Proposition 2 (Base-class) *Assume* $h \mapsto B_h$ *is successful with uniform convergence rate* $e(n)$*. Then, the update algorithm* U *and the regularization algorithm* R *with parameters* $\eta_n = \frac{1}{2\sqrt{n}}$*, and* $\epsilon_n = \sqrt{\ln(64n)/n} + 2e(n)$ *satisfy Condition (C1) with* $N(n) = 4n$ *and condition (C2) with* $c(n) = 2\eta_n + 3\epsilon_n$.

The proof of Proposition 2 is somewhat lengthy (3 pages), but it involves only elementary arguments. Due to space limitations, it is deferred to Section A in the Appendix.

3. Multiclass Classificaiton

We consider the multiclass case with $k < \infty$ labels. A hypothesis is a map from X to $[k]$. To define B_h we introduce cyclic permutations $s_0, \ldots, s_{k-1}$ of $[k]$ ($s_i(j) = (j + i) \mod k$) and let $B_h = \{s_i \circ h : i \leq k\}$. Note that for binary classification ($k = 2$) we have $B_h = \{h, 1 - h\}$.

Proposition 9 (Successful Base-Class: Multiclass Classification) *Let the label-space* Y *be* $Y = [k]$*. For each* $h : X \to Y$ *let* $B_h = \{s_i \circ h : i \leq k\}$*. Then, the mapping* $h \to B_h$ *is successful with uniform convergence rate* $e(n) = 36/\sqrt{n}$ *and with* $\alpha = \frac{k-1}{k}$.

The proof of Proposition 9 is arguably the most involved proof in this manuscript; in particularly establishing that the randomized ERM is monotone. Due to space limitation, the proof is deferred to Section C in the Appendix. The appendix (Section D) also contains the final (and simple) derivation of our main result, Theorem 2.

4. Open Questions and Future Research

We conclude this manuscript with some suggestions of open problems for future research.

Other Loss Functions. While the abstract framework developed in Section 2 extends to other (bounded) loss functions, the construction of the base classes B_h is tailored to the zero/one loss. It will be interesting to explore to which loss functions can one extend Theorem 2.

Can Monotone Learning Rules Achieve Optimal Rates? It will be interesting to explore whether the bound on the rate in Theorem 2 can be strengthened to retain optimal learning rates.

For example, for PAC learnable classes $\mathcal{H} \subseteq \{0,1\}^X$, the optimal learning rate in the agnostic setting scales like $\sqrt{d/m}$, and in the realizable setting like d/m, where d is the VC dimension and m is the input-sample size. Can these optimal rates be achieved by a monotone learning rule? Note that Theorem 2 is off by a $\log m$ factor.

Another interesting setting to explore this question is the model of *universal* learning, in which one focuses on distribution-dependent rates (Bousquet, Hanneke, Moran, van Handel, and Yehudayoff, 2021). In contrast with the distribution-free nature of PAC learning, some classes can be learned exponentially fast, at a rate which scales like $\exp(-n)$ (Schuurmans, 1997; Bousquet, Hanneke, Moran, van Handel, and Yehudayoff, 2021). Can such classes be learned monotonically in this fast rate?

Monotone Empirical Risk Minimization. Which classes admit a monotone empirical risk minimizer (ERM)? One of the key technical steps in our proof was to show that the class B_h admits a monotone ERM. Our proof exploited the symmetry of B_h (specifically, that each example is classified correctly by exactly one hypothesis in B_h). It will be interesting to determine which other classes admit monotone ERMs. In fact, as far as we know it is even open whether *every* (learnable) class admits a monotone ERM.

Acknowledgments

We thank Gábor Lugosi, Vladimir Pestov, and the anonymous reviewers for helpful feedlback on an earlier draft of this work.

YM received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 882396), the Israel Science Foundation (grant number 993/17), Tel Aviv University Center for AI and Data Science (TAD), and the Yandex Initiative for Machine Learning at Tel Aviv University.

SM is a Robert J. Shillman Fellow, his research is supported in part by the Israel Science Foundation (grant No. 1225/20), by a grant from the United States - Israel Binational Science Foundation (BSF), by an Azrieli Faculty Fellowship, by Israel PBC-VATAT, and by the Technion Center for Machine Learning and Intelligent Systems (MLIS).

HK is supported by the Israel Science Foundation grant no. 1595-19, and the Blavatnik Family Foundation.

US is supported by the Israel Science Foundation (grant 1871/19) and by Len Blavatnik and the Blavatnik Family foundation.

AD received funding from the European Research Council (ERC) under the European Union's Horizon 2022 research and innovation program (grant agreement No. 101041711), and the Israel Science Foundation (grant number 2258/19), and

References

Olivier Bousquet, Steve Hanneke, Shay Moran, Ramon van Handel, and Amir Yehudayoff. A theory of universal learning. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 532–541. ACM, 2021. doi: 10.1145/3406325.3451087. URL https://doi.org/10.1145/3406325.3451087.

Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media, 1996.

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley, 2 edition, 2001. ISBN 978-0-471-05669-0.

Baohua Gu, Feifang Hu, and Huan Liu. Modelling classification performance for large data sets. WAIM '01, page 317–328, Berlin, Heidelberg, 2001. Springer-Verlag. ISBN 3540422986.

G. H. Hardy, John E. Littlewood, and George Pólya. *Inequalities*. Cambridge University Press, Cambridge, 1988. ISBN 0521358809.

Gábor Lugosi. Pattern classification and learning theory. In *Principles of nonparametric learning*, pages 1–56. Springer, 2002.

Colin McDiarmid. On the method of bounded differences. In *Surveys in combinatorics, 1989 (Norwich, 1989)*, volume 141 of *London Math. Soc. Lecture Note Ser.*, pages 148–188. Cambridge Univ. Press, Cambridge, 1989.

Zakaria Mhammedi. Risk monotonicity in statistical learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Vladimir Pestov. A universally consistent learning rule with a universally monotone error. *arXiv preprint arXiv:2108.09733*, 2021.

D. Schuurmans. Characterizing rational versus exponential learning curves. *Journal of Computer and System Sciences*, 55(1):140–160, 1997.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014. ISBN 978-1-10-705713-5. URL http://www.cambridge.org/de/academic/subjects/computer-science/pattern-recognition-and-machine-learning/understanding-machine-learning-theory-algorithms.

David Tax and Robert Duin. Learning curves for the analysis of multiple instance classifiers. volume 5342, pages 724–733, 12 2008. ISBN 978-3-540-89688-3. doi: 10.1007/978-3-540-89689-0_76.

Tom Viering, Alexander Mey, and Marco Loog. Open problem: Monotonicity of learning. In *Conference on Learning Theory*, pages 3198–3201. PMLR, 2019.

Tom J. Viering and Marco Loog. The shape of learning curves: a review. *CoRR*, abs/2103.10948, 2021. URL https://arxiv.org/abs/2103.10948.

Tom Julian Viering, Alexander Mey, and Marco Loog. Making learners (more) monotone. In *International Symposium on Intelligent Data Analysis*, pages 535–547. Springer, 2020.

Gary M. Weiss and Alexander Battistin. Generating well-behaved learning curves : An empirical study. 2014.

Appendix A. Proof of Proposition 2

This section is dedicated to proving Proposition 2. We begin with collecting some simple properties of the regularization and update algorithms, and then continue to establish Conditions (C1) and (C2).

A.1. Basic Properties of R and U

Fix a source distribution D. Recall the definition of $\mathtt{LR}_n$ (Equation 2):

$$\mathtt{LR}_n(h) = \mathop{\mathbb{E}}_{S\sim D^n} \mathrm{L}_{\mathrm{D}}\, R(h,S)$$

We next present some basic properties of $\mathtt{LR}_n$ which will be useful in proving Proposition 2.

A simple calculation yields the following relationship between $\mathtt{LG}_n$ and $\mathtt{LR}_n$: for every hypothesis h and every n:

$$\begin{aligned}\mathtt{LR}_n(h) &= (1-\eta_n)\cdot \mathtt{LG}_n(h) + \eta_n \cdot \mathtt{LG}_0(h)\\ &= (1-\eta_n)\cdot \mathtt{LG}_n(h) + \eta_n \cdot \alpha \qquad (5)\\ &= \mathtt{LG}_n(h) + \eta_n\,(\alpha - \mathtt{LG}_n(h))\,. \qquad (6)\end{aligned}$$

The following claim asserts that $\mathtt{LR}_n$ is monotone:

Claim 1 *For all n, m, if $m \geq n$ and $\eta_m \leq \eta_n \leq 1$ then*

$$\mathtt{LR}_n(h) - \mathtt{LR}_m(h) \geq (\eta_n - \eta_m)\,(\alpha - \mathtt{LG}_m(h))\ ,$$

and in particular,

$$\forall h,\ \mathtt{LR}_m(h) \leq \mathtt{LR}_n(h)\,.$$

Proof By Equation (6):

$$\begin{aligned}\mathtt{LR}_n(h) - \mathtt{LR}_m(h) &= (1-\eta_n)\,(\mathtt{LG}_n(h) - \mathtt{LG}_m(h)) + (\eta_n - \eta_m)\,(\alpha - \mathtt{LG}_m(h))\\ &\geq (\eta_n - \eta_m)\,(\alpha - \mathtt{LG}_m(h)) && (\mathtt{LG}_m(h) \leq \mathtt{LG}_n(h) \text{ by Definition 6})\\ &\geq 0 && (\mathtt{LG}_m(h) \leq \mathtt{LG}_0(h) = \alpha \text{ by Definition 6})\end{aligned}$$

which proves the claim. ■

Lastly, observe that R does not deteriorate the performance of h:

Claim 2 *For all n,*

$$\mathtt{LR}_n(h) \leq \mathrm{L}_{\mathrm{D}}(h) + \eta_n + 2e(n)\,.$$

Proof

$$\begin{aligned}\mathtt{LR}_n(h) &= \mathtt{LG}_n(h) + \eta_n\,(\alpha - \mathtt{LG}_n(h)) && (\text{Equation (6)})\\ &\leq \mathtt{LG}_n(h) + \eta_n && (\mathtt{LG}_n(h) \leq \mathtt{LG}_0(h) = \alpha \leq 1)\\ &\leq \mathrm{L}_{\mathrm{D}}(h) + \eta_n + 2e(n). && (\text{Equation (4)})\end{aligned}$$

■

A.2. Update Probability

We first provide upper and lower bounds on the probability of making an update.

Lemma 10 *Let $u_n = \epsilon_n - 2e(n) = \sqrt{\ln(64n)/n}$. (Recall the definition of ϵ_n and η_n in Proposition 2.) If $\mathtt{LR}_n(h_1) > \mathtt{LR}_n(h_0)$ then*

$$\Pr_{S\sim D^n}[U(h_0,h_1,S)=h_1] \le 4\exp\left(-nu_n^2/2\right) = \frac{1}{2\sqrt{n}} = \eta_n\,,$$

and if $\mathtt{LR}_n(h_1) < \mathtt{LR}_n(h_0) - 2\epsilon_n$ then

$$\Pr_{S\sim D^n}[U(h_0,h_1,S)=h_1] \ge 1 - 4\exp\left(-nu_n^2/2\right).$$

Proof To simplify the calculations below, for a hypothesis h let

$$\hat{\mathrm{L}}^*_{\mathrm{S}}(h) := \min_{f\in B_h} \hat{\mathrm{L}}_{\mathrm{S}}(f) \tag{7}$$

$$\mathrm{L}^*_{\mathrm{D}}(h) := \min_{f\in B_h} \mathrm{L}_{\mathrm{D}}(f) \tag{8}$$

We first connect the performance of the regularized versions of the hypotheses h_0 and h_1 to their error probability $\mathrm{L}^*_{\mathrm{D}}(h_0)$ and $\mathrm{L}^*_{\mathrm{D}}(h_1)$. From Equation (5) we obtain

$$\mathtt{LR}_n(h_1) - \mathtt{LR}_n(h_0) = (1-\eta_n)(\mathtt{LG}_n(h_1) - \mathtt{LG}_n(h_0))\,. \tag{9}$$

Recall from the pseudo-code of the update algorithm U that the probability of update, which we denote p_n is given by

$$p_n := \Pr_{S\sim D^n}[U(h_0,h_1,S)=h_1] = \Pr_{S\sim D^n}\left[\hat{\mathrm{L}}^*_{\mathrm{S}}(h_0) - \hat{\mathrm{L}}^*_{\mathrm{S}}(h_1) > \epsilon_n\right].$$

Hence we have the following implications:

$$\begin{aligned}
\mathtt{LR}_n(h_1) > \mathtt{LR}_n(h_0) &\Rightarrow \mathtt{LG}_n(h_1) > \mathtt{LG}_n(h_0) && \text{(by Equation (9))}\\
&\Rightarrow \mathrm{L}^*_{\mathrm{D}}(h_1) + 2e(n) > \mathrm{L}^*_{\mathrm{D}}(h_0) && \text{(by Equation (4))}\\
&\Rightarrow p_n \le \Pr\left[|\hat{\mathrm{L}}^*_{\mathrm{S}}(h_0) - \mathrm{L}^*_{\mathrm{D}}(h_0)| + |\hat{\mathrm{L}}^*_{\mathrm{S}}(h_1) - \mathrm{L}^*_{\mathrm{D}}(h_1)| > \epsilon_n - 2e(n)\right]
\end{aligned}$$

and similarly

$$\begin{aligned}
\mathtt{LR}_n(h_0) > \mathtt{LR}_n(h_1) + 2\epsilon_n &\Rightarrow \mathtt{LG}_n(h_0) > \mathtt{LG}_n(h_1) + 2\epsilon_n && \text{(by Equation (9) and } \eta_n < 1)\\
&\Rightarrow \mathrm{L}^*_{\mathrm{D}}(h_0) + 2e(n) > \mathrm{L}^*_{\mathrm{D}}(h_1) + 2\epsilon_n && \text{(by Equation (4))}\\
&\Rightarrow 1 - p_n \le \Pr\left[|\hat{\mathrm{L}}^*_{\mathrm{S}}(h_0) - \mathrm{L}^*_{\mathrm{D}}(h_0)| + |\hat{\mathrm{L}}^*_{\mathrm{S}}(h_1) - \mathrm{L}^*_{\mathrm{D}}(h_1)| > \epsilon_n - 2e(n)\right]
\end{aligned}$$

Since $\epsilon_n - 2e(n) = u_n$ we have in both cases to upper bound

$$\Pr\left[|\hat{\mathrm{L}}^*_{\mathrm{S}}(h_0) - \mathrm{L}^*_{\mathrm{D}}(h_0)| + |\hat{\mathrm{L}}^*_{\mathrm{S}}(h_1) - \mathrm{L}^*_{\mathrm{D}}(h_1)| > u_n\right],$$

which is less than

$$\Pr\left[|\hat{\mathrm{L}}^*_{\mathrm{S}}(h_0) - \mathrm{L}^*_{\mathrm{D}}(h_0)| > u_n/2\right] + \Pr\left[|\hat{\mathrm{L}}^*_{\mathrm{S}}(h_1) - \mathrm{L}^*_{\mathrm{D}}(h_1)| > u_n/2\right]$$

We can apply Mcdiarmid's inequality (McDiarmid, 1989) to both terms. Recall that (a special case of) Mcdiarmid's inequality asserts that $\Pr[|Z - \mathbb{E}\,Z| \geq \epsilon] \leq 2\exp(-2n\epsilon^2)$ for every random function $Z = Z(V_1, \ldots, V_n)$, where the V_i's are independent, and such that Z is stable in the sense that $|Z(\vec{v}') - Z(\vec{v}'')| \leq 1/n$ for every pair of vectors $\vec{v}', \vec{v}''$ whose hamming distance is 1. Here it is applied on $Z(S) = \hat{\mathrm{L}}^*_{\mathrm{S}}(h) = \min_{f \in B_h} \hat{\mathrm{L}}_{\mathrm{S}}(f)$. We thus get the following upper bound:

$$2 \cdot 2\exp\left(-2n(u_n/2)^2\right) = 4\exp\left(-nu_n^2/2\right).$$

■

A.3. Condition (C1)

Lemma 11 *Under the assumptions of Proposition 2, Condition (C1) is satisfied with $N(n) = 4n$.*

Proof Let $n \in \mathbb{N}$ and $N(n) = 4n$. Set $p_N := \Pr_{S\sim D^N}[U(h_0, h_1, S) = h_1]$, the left-hand side of (C1) can be written as

$$\mathop{\mathbb{E}}_{S\sim D^N} \mathtt{LR}_N(U(h_0, h_1, S)) = p_N \mathtt{LR}_N(h_1) + (1 - p_N)\mathtt{LR}_N(h_0) \tag{10}$$

Assume first that $\mathtt{LR}_N(h_1) \leq \mathtt{LR}_n(h_0)$. In this case, since $N > n$, we have $\mathtt{LR}_N(h_0) \leq \mathtt{LR}_n(h_0)$. Thus, Equation (10) shows that $\mathbb{E}_{S\sim D^N} \mathtt{LR}_N(U(h_0, h_1, S))$ is a convex combination of two terms both $\leq \mathtt{LR}_n(h_0)$ hence $\mathbb{E}_{S\sim D^N} \mathtt{LR}_N(U(h_0, h_1, S)) \leq \mathtt{LR}_n(h_0)$.

Thus, assume that $\mathtt{LR}_N(h_1) > \mathtt{LR}_n(h_0)$. By Equation (10),

$$\mathop{\mathbb{E}}_{S\sim D^N}\left[\mathtt{LR}_N(U(h_0, h_1, S)) - \mathtt{LR}_n(h_0)\right] = p_N\left(\mathtt{LR}_N(h_1) - \mathtt{LR}_N(h_0)\right) + \mathtt{LR}_N(h_0) - \mathtt{LR}_n(h_0).$$

Therefore, it suffices to show that in this case

$$p_N\left(\mathtt{LR}_N(h_1) - \mathtt{LR}_N(h_0)\right) \leq \mathtt{LR}_n(h_0) - \mathtt{LR}_N(h_0).$$

Denoting $\eta' = \eta_n - \eta_N \geq 0$, we have, by Claim 1:

$$\mathtt{LR}_N(h_0) - \mathtt{LR}_n(h_0) \leq -\eta'\left(\alpha - \mathtt{LG}_N(h_0)\right) \tag{11}$$

Further, because $\mathtt{LG}_N(\cdot) \leq \alpha$,

$$\mathtt{LR}_N(h_1) - \mathtt{LR}_N(h_0) = (1 - \eta_N)\left(\mathtt{LG}_N(h_1) - \mathtt{LG}_N(h_0)\right) \leq \alpha - \mathtt{LG}_N(h_0).$$

The last two inequalities yield:

$$\begin{aligned} p_N\left(\mathtt{LR}_N(h_1) - \mathtt{LR}_N(h_0)\right) &\leq p_N\left(\alpha - \mathtt{LG}_N(h_0)\right) \\ \eta'\left(\alpha - \mathtt{LG}_N(h_0)\right)) &\leq \mathtt{LR}_n(h_0) - \mathtt{LR}_N(h_0). \end{aligned}$$

Hence, condition (C1) is satisfied provided that $p_N \leq \eta'$. By Lemma 10, we have

$$p_N \leq 2 \cdot 2\exp\left(-2N(u_N/2)^2\right) = 4\exp\left(-Nu_N^2/2\right) = 4\exp\left(-\frac{1}{2}\ln(64N)\right) = \frac{1}{2\sqrt{N}}\,.$$

Now we can verify that for $N = 4n$,

$$\eta' = \frac{1}{2\sqrt{n}} - \frac{1}{2\sqrt{N}} = \frac{1}{2\sqrt{N}},$$

and hence

$$p_N \leq \eta'$$

which concludes the proof. ■

A.4. Condition (C2)

Lemma 12 *Under the assumptions of Propositon 2, Condition (C2) is satisfied with*

$$c(n) = 2\eta_n + 3\epsilon_n\,.$$

Proof Let $q_n := \Pr_{S\sim D^n}[U(h_0, h_1, S) = h_0]$ be the probability of not switching to h_1. Thus,

$$\mathop{\mathbb{E}}_{S\sim D^n} \mathtt{LR}_n(U(h_0, h_1, S)) = q_n \mathtt{LR}_n(h_0) + (1 - q_n)\mathtt{LR}_n(h_1)$$

Therefore, if $\mathtt{LR}_n(h_0) \leq \mathtt{LR}_n(h_1) + 2\epsilon_n$ then :

$$\begin{aligned}
\mathop{\mathbb{E}}_{S\sim D^n} \mathtt{LR}_n(U(h_0, h_1, S)) &\leq \mathtt{LR}_n(h_1) + 2\epsilon_n \\
&\leq \mathrm{L}_{\mathrm{D}}(h_1) + \eta_n + 2e(n) + 2\epsilon_n && \text{(by Claim 2)} \\
&\leq \mathrm{L}_{\mathrm{D}}(h_1) + 2\eta_n + 3\epsilon_n\,. && (2e(n) \leq \epsilon_n)
\end{aligned}$$

Thus, the conclusion holds in this case. Therefore, assume that $\mathtt{LR}_n(h_0) > \mathtt{LR}_n(h_1) + 2\epsilon_n$. In this case,

$$\begin{aligned}
\mathop{\mathbb{E}}_{S\sim D^n} \mathtt{LR}_n(U(h_0, h_1, S)) &= q_n \mathtt{LR}_n(h_0) + (1 - q_n)\mathtt{LR}_n(h_1) \\
&\leq q_n + \mathtt{LR}_n(h_1). && (q_n, \mathtt{LR}_n(h_0) \in [0, 1]) \\
&\leq \mathtt{LR}_n(h_1) + \eta_n && \text{(by Lemma 10)} \\
&\leq \mathrm{L}_{\mathrm{D}}(h_1) + 2\eta_n + 2e(n) && \text{(by Claim 2)} \\
&\leq \mathrm{L}_{\mathrm{D}}(h_1) + 2\eta_n + \epsilon_n\,. && (2e(n) \leq \epsilon_n)
\end{aligned}$$

■

Appendix B. Binary Classification

In this section we prove that in the setting of binary classification, one can associate with every hypothesis h a base class B_h such that Definition 6 is satisfied.

Proposition 13 (Successful Base-Class: Binary Classification) *Let the label-space Y be $Y = \{0, 1\}$. For each $h : X \to Y$ let $B_h = \{h, 1 - h\}$. Then, the mapping $h \to B_h$ is successful with uniform convergence rate $e(n) = 1/\sqrt{n}$ and with $\alpha = 1/2$.*

Proof That $e(n) = 1/\sqrt{n}$ follows by elementary probabilistic argument (essentially bounding the variance of a Binomial random variable):

$$\begin{aligned}
e(n) &= \mathop{\mathbb{E}}_{S\sim D^n} \max\left(\left|\hat{\mathrm{L}}_S(h) - \mathrm{L}_D(h)\right|, \left|\hat{\mathrm{L}}_S(1-h) - \mathrm{L}_D(1-h)\right|\right) \\
&= \mathop{\mathbb{E}}_{S\sim D^n} \left|\hat{\mathrm{L}}_S(h) - \mathrm{L}_D(h)\right| \\
&\le \sqrt{\mathop{\mathbb{E}}_{S\sim D^n} \left(\hat{\mathrm{L}}_S(h) - \mathrm{L}_D(h)\right)^2} && \text{(By Jensen's inequality)} \\
&= \sqrt{\mathtt{Var}(\hat{\mathrm{L}}_S(h))} \\
&\le \frac{1}{\sqrt{n}}.
\end{aligned}$$

Also, showing that $\alpha = 1/2$ is simple: indeed for every distribution D:

$$\mathtt{LG}_0(h) = \frac{1}{2}\mathrm{L}_D(h) + \frac{1}{2}\mathrm{L}_D(1-h) = \frac{1}{2}\mathrm{L}_D(h) + \frac{1}{2}\big(1 - \mathrm{L}_D(h)\big) = \frac{1}{2}.$$

Thus, it remains to prove the first Item in Definition 6:

Lemma 14 *For all n, m such that $m \ge n$ we have*

$$\forall h,\ \mathtt{LG}_m(h) \le \mathtt{LG}_n(h).$$

We note that Pestov (2021) proved this statement when m, n are odd. (See Lemma 3.1 in (Pestov, 2021)). We defer the proof to the next section where we derive a more general result that applies to multiclass with an arbitrary number of labels k. (See Lemma 16.) ■

Appendix C. Multiclass Classification

We consider the multiclass case with $k < \infty$ labels. A hypothesis is a map from X to $[k]$. To define B_h we introduce cyclic permutations $s_0, \ldots, s_{k-1}$ of $[k]$ ($s_i(j) = (j + i) \mod k$) and let $B_h = \{s_i \circ h : i \le k\}$.

Proposition 15 (Successful Base-Class: Multiclass Classification) *Let the label-space $Y = [k]$. For each $h : X \to Y$ let $B_h = \{s_i \circ h : i \le k\}$. Then, the mapping $h \to B_h$ is successful with uniform convergence rate $e(n) = 36/\sqrt{n}$ and with $\alpha = \frac{k-1}{k}$.*

Proof Let $h \in H$. We begin with the simplest part: namely that $\alpha = \frac{k-1}{k}$. Notice that the events

$$E_i = \{(x, y) : s_i \circ h(x) = y\}$$

are pairwise disjoint; in fact they form a partition of $X \times Y$ because for each (x, y) there is a unique i such that $s_i \circ h(x) = y$). Therefore, for every distribution D over $X \times Y$:

$$\begin{aligned}
\mathtt{LG}_0(h) &= \frac{1}{k}\sum_{i=1}^{k} \mathrm{L}_{\mathrm{D}}(s_i \circ h) \\
&= \frac{1}{k}\sum_{i=1}^{k} \big(1 - D(E_i)\big) \\
&= \frac{1}{k}\big(k - \sum_{i=1}^{k} D(E_i)\big) \\
&= \frac{k-1}{k}.
\end{aligned}$$

To see that $e(n) = 36/\sqrt{n}$ notice that the family of events $\mathcal{E} = \{E_i : i \le k\}$ consists of pairwise disjoint events and therefore its VC dimension is 1. By VC theory (see[5] e.g., (Lugosi, 2002))

$$\begin{aligned}
\frac{36}{\sqrt{n}} &\ge \mathop{\mathbb{E}}_{S\sim D^n} \max_i \big|D_S(E_i) - D(E_i)\big| \\
&= \mathop{\mathbb{E}}_{S\sim D^n} \max_i \big|\big(1 - D_S(E_i)\big) - \big(1 - D(E_i)\big)\big| \\
&= \mathop{\mathbb{E}}_{S\sim D^n} \max_i \big|\hat{\mathrm{L}}_{\mathrm{S}}(s_i \circ h) - \mathrm{L}_{\mathrm{D}}(s_i \circ h)\big|.
\end{aligned}$$

Above, D_S denotes the empirical distribution induced by the sample $S = \{(x_i, y_i)\}_{i=1}^m$ (i.e., $D_S(E) = \frac{1}{m}\sum_{i=1}^m 1[(x_i, y_i) \in E]$.). Thus, it remains to prove the first Item in Definition 6:

Lemma 16 *For all n, m such that $m \ge n$ we have*

$$\forall h,\ \mathtt{LG}_m(h) \le \mathtt{LG}_n(h).$$

Proof By induction, it suffices to consider the case of $m = n + 1$.

We first reformulate this problem in simpler terms. Recall that any example $z = (x, y)$ is classified correctly by exactly one of the h_j's in B_h. Thus, partition the domain into sets $E_1, \ldots, E_k$ (with $k = |B_h|$) such that $z \in E_j \Leftrightarrow h_j(x) = y$. Thus, we have $\Pr(E_j) = 1 - \mathrm{L}_{\mathrm{D}}(h_j)$. To simplify the expressions below we denote

$$p_i = 1 - \mathrm{L}_{\mathrm{D}}(h_i),\ \ q_i = \mathrm{L}_{\mathrm{D}}(h_i)\,.$$

Let $S = \{z_i\}_{i=1}^n \sim D^n$ be an i.i.d sample and let $i \le n$. Define X_i to be the random variable which is equal to the unique index j such that $z_i \in E_j$. (I.e., h_j correctly classifies z_i.) Notice that

5. Theorem 1.16 in (Lugosi, 2002) gives $\mathbb{E}\sup_{E\in\mathcal{E}} |D_S(E) - D(E)| \le \frac{24}{\sqrt{n}}\int_0^1 \sqrt{\ln(2N(\epsilon))}d\epsilon$, where $N(\epsilon)$ denotes the covering number of the family $\mathcal{E}$. Here, since the E_i's are disjoint, $N(\epsilon) \le 1+1/\epsilon$, and we have $\int_0^1 \sqrt{\ln(2+2/\epsilon)}d\epsilon \approx 1.22 \le \frac{3}{2}$ so we get $\mathbb{E}\max_i |D_S(E_i) - D(E_i)| \le \frac{36}{\sqrt{n}}$

the variables $(X_1, \ldots, X_n)$ are i.i.d. with values in $[k]$ and distribution given by $\Pr(X_i = j) = p_j$ for all $i \le n, j \le k$. We can then express $\mathtt{LG}_n(h)$ as the expectation over the sample $L_n = (X_1, \ldots, X_n)$ of the quantity

$$f(L_n) := \frac{1}{|I|} \sum_{j \in I} q_j,$$

where $I = I(L_n)$ is the set of indices j such that $\sum_{i=1}^{n} 1[X_i = j]$ is largest. (Equivalently, such that h_j is an empirical risk minimizer in B_h with respect to the input sample S.)

Let $L = L_{n+1}$ be a sample of $n+1$ such variables $(X_1, \ldots, X_{n+1})$ (corresponding to an input sample $S \sim D^{n+1}$), and let L^{-i} denote the sample L without its i-th element, by symmetry and by linearity of expectation, we have

$$\mathtt{LG}_{n+1}(h) - \mathtt{LG}_n(h) = \frac{1}{n+1} \sum_{i=1}^{n+1} \mathop{\mathbb{E}}_{L} \left[f(L) - f(L^{-i}) \right]$$

We first study the above quantity when conditioned on $|I| > 1$ (i.e., there are at least 2 empirical risk minimizers in B_h). Notice that when $|I| > 1$ every $i \le n+1$ satisfies $I_i \subseteq I$, where $I_i = I(L^{-i})$. Also notice that if $|I| > 1$ then $I_i = I$ if and only if $X_i \notin I$. Thus,

$$\begin{aligned}
&\sum_{i=1}^{n+1} \mathop{\mathbb{E}}_{L} \left[f(L) - f(L^{-i}) \middle| |I| > 1 \right] \\
&= \sum_{i=1}^{n+1} \mathop{\mathbb{E}}_{L} \left[1[X_i \in I] \cdot \left(\frac{1}{|I|} \sum_{j \in I} q_j - \frac{1}{|I|-1} \sum_{j \in I, j \neq X_i} q_j \right) \middle| |I| > 1 \right] \qquad \text{(if } X_i \notin I, f(L) = f(L^{-i})\text{)} \\
&= \mathop{\mathbb{E}}_{L} \left[\sum_{i=1}^{n+1} 1[X_i \in I] \cdot \left(\frac{1}{|I|} \sum_{j \in I} q_j - \frac{1}{|I|-1} \sum_{j \in I, j \neq X_i} q_j \right) \middle| |I| > 1 \right] \\
&= \mathop{\mathbb{E}}_{L} \left[\sum_{k \in I} \left(\sum_{i=1}^{n+1} 1[X_i = k] \right) \cdot \left(\frac{1}{|I|} \sum_{j \in I} q_j - \frac{1}{|I|-1} \sum_{j \in I, j \neq k} q_j \right) \middle| |I| > 1 \right]
\end{aligned}$$

Since all elements of I have the same number of successes, the sum $\sum_{i=1}^{n+1} 1[X_i = k]$ is the same for all $k \in I$ and since

$$\frac{1}{|I|} \sum_{k \in I} \sum_{j \in I} q_j = \sum_{j \in I} q_j = \frac{1}{|I|-1} \sum_{k \in I} \sum_{j \in I, j \neq k} q_j$$

this shows that

$$\sum_{i=1}^{n+1} \mathop{\mathbb{E}}_{L} \left[f(L) - f(L^{-i}) \middle| |I| > 1 \right] = 0\,. \tag{12}$$

Now let us consider the case $|I| = 1$. Let $J \supseteq I$ denote the set of almost minimizers (i.e., which are either optimal or one away from being optimal) We further condition on J being some arbitrary fixed set J_0:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{L} \left[f(L) - f(L^{-(n+1)}) \middle| |I| = 1, J = J_0 \right] \\
&= \Pr\left[I = \{X_{n+1}\} \middle| |I| = 1, J = J_0 \right] \cdot \mathop{\mathbb{E}}_{L} \left[f(L) - f(L^{-(n+1)}) \middle| I = \{X_{n+1}\}, J = J_0 \right]. \\
&\qquad\qquad \text{(if } X_{n+1} \notin I \text{ then } f(L) - f(L^{-(n+1)}) = 0\text{)}
\end{aligned}$$

Therefore, since $\Pr[I = \{X_{n+1}\} || I| = 1, J = J_0\}] > 0$, it is enough to consider

$$\begin{aligned}\mathop{\mathbb{E}}_{L}\left[f(L) - f(L^{-(n+1)})\middle| I = \{X_{n+1}\}, J = J_0\right] &= \sum_{i \in J_0} \Pr(X_{n+1} = i) \mathop{\mathbb{E}}_{L}\left[f(L) - f(L^{-(n+1)})\middle| I = \{i\}, J = J_0\right] \\ &= \sum_{i \in J_0} \Pr(X_{n+1} = i) \mathop{\mathbb{E}}_{L}\left[q_i - \frac{1}{|J|}\sum_{j \in J} q_j \middle| I = \{i\}, J = J_0\right] \\ &= \mathop{\mathbb{E}}_{L}\left[\sum_{i \in J_0} p_i q_i - \frac{1}{|J_0|}\sum_{i,j \in J_0} p_i q_j \middle| I = \{i\}, J = J_0\right].\end{aligned}$$

To see that the above is non-positive we use Chebyshev's sum inequality, which asserts that if $a_1 \leq \ldots \leq a_n$ and $b_1 \geq \ldots \geq b_n$ then $\frac{1}{n}\sum a_i \cdot \frac{1}{n}\sum b_i \geq \frac{1}{n}\sum a_i b_i$ (Hardy, Littlewood, and Pólya, 1988). Thus, since $p_i = 1 - q_i$, this inequality implies that the left term in the expectation is upper bounded by the right term, and so we get that the difference is non-positive. Hence, for every choice of J_0:

$$\mathop{\mathbb{E}}_{L}\left[f(L) - f(L^{-(n+1)})\middle| |I| = 1, J = J_0\right] \leq 0\,,$$

which together with Equation (12) allows to conclude the proof of the lemma. ■

This concludes the proof of Proposition 9. ■

Appendix D. Wrapping Up

Proof [Proof of Theorem 2.] Let A be any learning algorithm Proposition 9, Proposition 2, and Proposition 1 imply the existence of a monotone algorithm M such that for all $m \geq 2 \cdot b(1) + b(0)$,

$$\mathop{\mathbb{E}}_{S \sim D^m} \mathrm{L_D}(M(S)) \leq \mathop{\mathbb{E}}_{S \sim D^m} \mathrm{L_D}(A(S_{T-2})) + c(b(T-1)), \tag{13}$$

where

1. $b(x) = 4^x$,
2. $T = T(m)$ is the maximal integer such that $b(T-1) + \sum_{t=0}^{T-1} b(t) \leq m$, and
3. S_{T-2} is an i.i.d sample from the source distribution D of size $\sum_{t=0}^{T-2} b(t)$.
4. $c(x) = 2 \cdot \frac{1}{2\sqrt{x}} + 3\left(\sqrt{\ln(64x)/x} + 2 \cdot \frac{36}{\sqrt{x}}\right) = O\left(\sqrt{\frac{\log x}{x}}\right)$,

Since M is monotone, it remains to prove that M's performance is competitive with that of A. The case of input sample-size $m < 2b(1) + b(0) = 9$ is trivial. So, assume $m \geq 9$ and hence Equation 13 holds. By Items 1–4 above it suffices to show that $|S_{T-2}| \geq (m/30) - 1$ and that $b(T-1) \geq m/10$. We proceed by showing that $T = T(m) = \Theta(\log m)$. Recall that T is the maximal positive integer such that

$$m \geq 4^{T-1} + \sum_{t=0}^{T-1} 4^t = 2 \cdot 4^{T-1} + \frac{4^{T-1} - 1}{4 - 1} = \frac{7}{3} \cdot 4^{T-1} - \frac{1}{3}.$$

Thus,

$$T = 1 + \left\lfloor \log_4\left(\frac{3m+1}{7}\right)\right\rfloor,$$

and $b(T-1)$, $|S_{T-2}|$ satisfy:

$$b(T-1) = 4^{T-1} = 4^{\lfloor \log_4(\frac{3m+1}{7})\rfloor} \geq \frac{1}{4}\cdot\frac{3m+1}{7} \geq \frac{m}{10}.$$

$$\begin{aligned} |S_{T-2}| &= \sum_{t=0}^{T-2} 4^t \\ &= \frac{4^{T-1}-1}{3} \\ &\geq \frac{\frac{1}{4}\cdot\frac{3m+1}{7}-1}{3} \\ &= \frac{m}{28} - \frac{1}{14} \geq \frac{m}{30} - 1. \end{aligned}$$

■