Proceedings of Machine Learning Research vol 195:1–29, 2023 36th Annual Conference on Learning Theory

List Online Classification

Shay Moran SMORAN@TECHNION.AC.IL
Departments of Mathematics and Computer Science, Technion–IIT and Google Research

Ohad Sharon OHAD.SHARON@CAMPUS.TECHNION.AC.IL
Department of Mathematics, Technion–IIT

Iska Tsubari ISKA@CAMPUS.TECHNION.AC.IL
Department of Mathematics, Technion–IIT

Sivan Yosebashvili SIVAN.Y@CAMPUS.TECHNION.AC.IL
Mobileye

Editors: Gergely Neu and Lorenzo Rosasco

Abstract

We study multiclass online prediction where the learner can predict using a list of multiple labels (as opposed to just one label in the traditional setting). We characterize learnability in this model using the b-ary Littlestone dimension. This dimension is a variation of the classical Littlestone dimension with the difference that binary mistake trees are replaced with $(k+1)$-ary mistake trees, where k is the number of labels in the list. In the agnostic setting, we explore different scenarios depending on whether the comparator class consists of single-labeled or multi-labeled functions and its tradeoff with the size of the lists the algorithm uses. We find that it is possible to achieve negative regret in some cases and provide a complete characterization of when this is possible.

As part of our work, we adapt classical algorithms such as Littlestone's SOA and Rosenblatt's Perceptron to predict using lists of labels. We also establish combinatorial results for list-learnable classes, including a list online version of the Sauer-Shelah-Perles Lemma. We state our results within the framework of pattern classes — a generalization of hypothesis classes which can represent adaptive hypotheses (i.e. functions with memory), and model data-dependent assumptions such as linear classification with margin.

Keywords: Online Learning, List Learning, Multiclass Classification, Littlestone Dimension, Perceptron, Mistake Bound, Regret Bound

1. Introduction

In certain situations where supervised learning is used, it is acceptable if the prediction is presented as a short list of candidate outputs. For instance, recommendation systems like those used by Netflix, Amazon, and YouTube (see Figure 1) recommend a list of movies or products to their users for selection (as opposed to just a single movie/product). In fact, even in tasks where the objective is to predict a single label, it may be helpful to first reduce the output space (which can potentially be very large) by learning a short list of options. For instance, medical doctors could consult an algorithm that generates a short list of potential diagnoses based on a patient's medical data.

List prediction rules naturally arise in the setting of conformal learning. In this model, algorithms make their predictions while also offering some indication of the level of uncertainty or confidence in those predictions. For example in multiclass classification tasks, given an unlabeled test example x, the conformal learner might output a list of all possible classes along with scores which reflect the probability that x belongs to each class. This list can then be truncated to a shorter

Figure 1: YouTube's recommendation system generates a short list of videos to users based on their past watch history and likes/dislikes.

one which contains only the classes with the highest score. We refer the reader to the book by Vovk, Gammerman, and Shafer (2005) and the surveys by Shafer and Vovk (2008); Angelopoulos and Bates (2021) for more details.

Recently, list learning has been examined within the PAC (Probably Approximately Correct) framework. Brukhim, Carmon, Dinur, Moran, and Yehudayoff (2022) proved a characterization of multiclass PAC learning using the Daniely-Shwartz (DS) dimension. A central idea in this work is the use of list learners to decrease the (possibly infinite) label space; this enables them to learn the reduced problem using known methods which apply when the number of labels is bounded. In a subsequent study, Charikar and Pabbaraju (2022) investigated which classes can be learned in the list PAC model, and gave a characterization of list learnability using a natural extension of the DS dimension. In the process, they identified key characteristics of PAC learnable classes that extend naturally (but not trivially) to the list setting.

The purpose of this study is to examine list learnability in the setting of multiclass online classification. Specifically, we aim to understand:

Guiding Questions

- What tasks are online learnable by algorithms that make predictions using short lists of labels? Is there a natural dimension that measures list online learnability?
- How does the possibility of using lists impact the mistake- and regret-bounds? How do these bounds improve as the algorithm is allowed to use larger lists?

We study these questions in the context of multiclass online prediction using the 0/1 classification loss. Extensions to weighted and continuous losses are left for future research.

1.1. Our Contribution

Realizable Case (Theorems 2 and 4). Our first main result characterizes list learnability using a natural variation of the Littlestone dimension (Littlestone, 1988). We show that a class is k-list online learnable (i.e. can be learned by an algorithm that predicts using list of size k) if and only

if its $(k+1)$-ary Littlestone dimension is bounded. The latter is defined as the maximum possible depth of a complete $(k+1)$-ary mistake tree that is shattered by the class. We further show that the list Littlestone dimension is equal to the optimal mistake bound attainable by deterministic learners.

Agnostic Case (Theorems 5 and 6). We characterize agnostic list online learnability by showing it is also captured by a bounded list Littlestone dimension (and hence agnostic and realizable list learning are equivalent). Our proof hinges on an effective transformation that converts a list learner in the realizable case to an agnostic list learner.

We further investigate how does the expressiveness of the comparator class and the resources of the algorithm impact the optimal regret rates. For example, consider the task of agnostic learning a multi-labeled comparator class consisting of functions which maps an input instance x to a set of $k' = 3$ output labels. How does the optimal regret changes as we use list learners with different list size $k = 1, 2, 3, \ldots$? Clearly , the regret cannot increase as we increase k; does the regret decreases? By how much? Can it become negative? We show that once the algorithm can use larger lists than the minimum required for learning, then it can achieve negative regret (which can be called *pride*). In Theorem 6 we characterize the cases in which negative regret is possible as a function of the trade-off between the algorithm's resources and the class complexity.

Variations of Classical Results and Algorithms. We develop and analyze variations of classical algorithms such as Rosenblatt's Perceptron (Figure 3) and Littlestone's SOA (Figure 4). The new modified algorithms predict using lists and hence are more expressive. For example, our k-list Perceptron algorithm learns any linearly separable sequence with margin between the top class and the $(k+1)$'th top class (as opposed to margin between the top and the second top classes in the standard multiclass Perceptron).

Our transformation of realizable list learners to agnostic list learners extends the one by Ben-David, Pál, and Shalev-Shwartz (2009) in the binary case. Along the way we prove an online variation of the Sauer-Shelah-Perles Lemma for list learnable classes (Proposition 1).

Pattern Classes. We present our results using a general framework of pattern classes. In a nutshell, a pattern class $\mathcal{P}$ is a set of sequences of examples; these sequences are thought of as the set of allowable sequences that are expected as inputs. Thus, in the realizable case, the goal is to design a learning rule that makes a bounded number of mistake on every pattern in the class. Every hypothesis class $\mathcal{H}$ induces a pattern class consisting of all sequences that are realizable by $\mathcal{H}$:

$$\mathcal{P}(\mathcal{H}) = \{S : S \text{ is realizable by } \mathcal{H}\}$$

However, pattern classes have the advantage of naturally expressing data-dependent assumption such as linear classification with margin (see Section 3.1). In fact, pattern classes are more expressive than hypothesis classes: in Appendix B we give examples of an online learnable pattern class $\mathcal{P}$ such that for every hypothesis class $\mathcal{H}$, if $\mathcal{P} \subseteq \mathcal{P}(\mathcal{H})$ then $\mathcal{H}$ is not online learnable.

Pattern classes are even more expressive than partial concept classes (Long, 2001; Alon, Hanneke, Holzman, and Moran, 2021). In fact, partial concept classes essentially correspond to symmetric pattern classes: i.e. pattern classes with the property that a permutation of every pattern in the class is also in the class. Asymmetric pattern classes can be used to represent online functions (or functions with memory).[1]

1. Online functions are maps $h : \mathcal{X}^\star \to \mathcal{Y}$ that get as an input a finite sequence of instances $x_1, \ldots, x_T$ and produce a sequence of labels $y_1, \ldots, y_T$, where $y_t = h(x_t; x_{t-1}, \ldots, x_1)$ is interpreted as the label of x_t. In contrast with

1.2. Organization

We begin with giving the necessary background and formal definitions in Section 2. While this section contains standard and basic definitions in online learning, we still recommend that even the experienced readers will skim through it, especially through the less standard parts which define learnability of pattern classes and introduce the list learning setting.

In Section 3 we state and present our main results, and in Section A we provide the proofs along with some additional results with more elaborate bounds. Finally, Section 4 contains some suggestions for future research.

2. Definitions and Background

Let $\mathcal{X}$ be a set called the domain and let $\mathcal{Y}$ be a set called the label space. We assume that $\mathcal{Y}$ is finite (but possibly very large). A pair $z = (x, y) \in \mathcal{X} \times \mathcal{Y}$ is called an example, and $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ denotes the space of examples. An hypothesis (or a concept) is a mapping $h : \mathcal{X} \to \mathcal{Y}$, an hypothesis class (or a concept class) $\mathcal{H}$ is a set of hypotheses. More generally, for $|\mathcal{Y}| \geq k \geq 1$, a k multi-labeled hypothesis is a map $h : \mathcal{X} \to \binom{\mathcal{Y}}{k}$ taking an input $x \in \mathcal{X}$ to a set of k labels from $\mathcal{Y}$. A k multi-labeled hypothesis class is a class of k multi-labeled hypotheses.

Pattern Classes. We use pattern classes – a generalization of hypothesis classes which enables modelling data-dependent assumptions such as linear classification with margin in a unified and natural way.

A pattern $S = \{z_t\}_{t=1}^{T}$ is a finite sequence of examples. For $0 < i \leq T$ we let $S_{<i}$ denote the prefix of S which consists of the first $i-1$ examples in S. For a pair of patterns S_1, S_2 we denote by $S_1 \circ S_2$ the pattern obtained by concatenating S_2 after S_1. We denote by $S_1 \subseteq S_2$ the relation "S_1 is a subsequence of S_2". For a set A, we denote by $A^\star = \cup_{T=0}^{\infty} A^T$ the set of all finite sequences[2] of elements in A. Thus, $\mathcal{Z}^\star$ is the set of all patterns.

A pattern class $\mathcal{P} \subseteq \mathcal{Z}^\star$ is a set of patterns which is *downward closed*: for every $S \in \mathcal{P}$, if $S' \subseteq S$ is a subsequence of S then $S' \in \mathcal{P}$. For a pattern class $\mathcal{P}$, $y \in \mathcal{Y}$ and $x \in \mathcal{X}$, we denote by $\mathcal{P}_{x \to y}$ the subset of $\mathcal{P}$ consists of all patterns such that their concatenation with the example (x, y) is also in $\mathcal{P}$:

$$\mathcal{P}_{x \to y} = \{S : (x, y) \circ S \in \mathcal{P}\}.$$

We say that a sequence/pattern S is realizable by $\mathcal{P}$ if $S \in \mathcal{P}$. we say that a sequence/pattern $S \in \mathcal{Z}^\star$ is consistent with (or realizable by) a multi-labeled hypothesis $h : \mathcal{X} \to \mathcal{Y}$ if for every $(x_t, y_t) \in S$ we have $y_t \in h(x_t)$ (or $y_t = h(x_t)$ when h is a single-labeled hypothesis).

Notice that every (multi-labeled) hypothesis class $\mathcal{H}$ induces a pattern class:

$$\mathcal{P}(\mathcal{H}) = \{S \in \mathcal{Z}^\star : S \text{ is consistent with some } h \in \mathcal{H}\}.$$

So, a sequence S is realizable by $\mathcal{H}$ if and only if $S \in \mathcal{P}(\mathcal{H})$. The pattern class $\mathcal{P}(\mathcal{H})$ is equivalent to $\mathcal{H}$ from a learning theoretic perspective in the sense that a learning rule learns $\mathcal{H}$ if and only

oblivious (memoryless) functions, the rule that assigns y_t to x_t can depend on the history (i.e. on $x_1, \ldots x_{t-1}$). As an example, consider the online function which gets as an input a sequence of points $x_1, \ldots, x_T \in \mathbb{R}^d$ and produces labels $y_1, \ldots y_T \in \{0, 1\}$ such that $y_1 = 0, y_2 = 1$ and for $t > 2$, y_t is set to be the label of the closest point to x_t among $x_1, \ldots x_{t-1}$ (breaking ties arbitrarily). One can model online function classes using pattern classes by taking the set of all patterns $(x_1, y_1), \ldots, (x_t, y_t)$ that are obtained by feeding $x_1, \ldots, x_t$ to one of the online functions in the class.

2. Here, A^0 is the set $A^0 = \{\emptyset\}$ which contains only the empty sequence $\emptyset$.

if it learns $\mathcal{P}(\mathcal{H})$. Hence, pattern classes are at least as expressive as hypothesis classes. The upside is that pattern classes allow to naturally express data-dependent assumptions such as linear separability with margin (as we study in more detail below). This is done by simply considering the pattern class which consists of all sequences that satisfy the assumed data-dependent condition. Pattern classes extend the notion of partial concept classes (Auer and Long, 1999; Alon et al., 2021). In Appendix B we give examples of a learnable task that can be represented by a pattern class, but not by an hypothesis class. That is, a learnable pattern class $\mathcal{P}$ such that every hypothesis class $\mathcal{H}$ for which $\mathcal{P} \subseteq \mathcal{P}(\mathcal{H})$ is not learnable.

Deterministic Learning Rules. A (deterministic) online learner is a mapping $\mathcal{A} : \mathcal{Z}^\star \times \mathcal{X} \to \mathcal{Y}$. That is, it is a mapping that maps a finite sequence $S \in \mathcal{Z}^\star$ (the past examples), and an unlabeled example x (the current test point) to a label y, which is denoted by $y = \mathcal{A}(x; S)$.

Let $k \geq 1$; a deterministic k-list learner is a mapping $\mathcal{A} : \mathcal{Z}^\star \times \mathcal{X} \to \binom{\mathcal{Y}}{k}$. That is, rather than outputting a single label, a k-list learner outputs a set of k labels. For a sequence $S = \{(x_t, y_t)\}_{t=1}^T \in \mathcal{Z}^\star$ we denote by $\mathtt{M}(\mathcal{A}; S)$ the number of mistakes $\mathcal{A}$ makes on S:

$$\mathtt{M}\big(\mathcal{A}; S\big) = \sum_{t=1}^{n} 1\big[y_t \notin \mathcal{A}(x_t; S_{<t})\big].$$

Randomized Learning Rules. In the randomized setting we follow the standard convention in the online learning literature and model randomized learners as deterministic mappings $\mathcal{A} : \mathcal{Z}^\star \times \mathcal{X} \to \Delta(\mathcal{Y})$, where $\Delta(\mathcal{Y})$ denote the space of all distributions over $\mathcal{Y}$. Thus, $\hat{p} = \mathcal{A}(x; S)$ is a $|\mathcal{Y}|$-dimensional probability vector representing the probability of the output label $\mathcal{A}$ assigns to x given S (see e.g. Cesa-Bianchi and Lugosi (2006); Shalev-Shwartz (2012); Hazan (2019)). For a sequence $S = \{(x_t, y_t)\}_{t=1}^T \in \mathcal{Z}^\star$ we denote by $\mathtt{M}(\mathcal{A}; S)$ the expected number of mistakes $\mathcal{A}$ makes on S:

$$\mathtt{M}\big(\mathcal{A}; S\big) = \mathop{\mathbb{E}}_{\hat{y}_t \sim \hat{p}_t}\Big[\sum_{t=1}^{T} 1[\hat{y}_t \neq y_t]\Big] = \sum_{t=1}^{T} \Pr_{\hat{y}_t \sim \hat{p}_t}[\hat{y}_t \neq y_t]. \qquad \text{(where } \hat{p}_t = \mathcal{A}(x_t; S_{<t})\text{)}$$

Similarly, for $k \geq 1$ a randomized k-list learner is a mapping $\mathcal{A} : \mathcal{Z}^\star \times \mathcal{X} \to \Delta\big(\binom{\mathcal{Y}}{k}\big)$, where $\Delta\big(\binom{\mathcal{Y}}{k}\big)$ is the space over all distributions over subsets of size k of $\mathcal{Y}$. Thus, the expected number mistakes $\mathcal{A}$ makes on a sequence S is given by

$$\mathtt{M}\big(\mathcal{A}; S\big) = \sum_{t=1}^{T} \Pr_{\hat{L}_t \sim \hat{q}_t}[y_t \notin \hat{L}_t]. \qquad \text{(where } \hat{q}_t = \mathcal{A}(x_t; S_{<t})\text{)}$$

Realizable Case Learnability (Pattern Classes). Let $\mathcal{P}$ be a pattern class and let $k \geq 1$. We say that $\mathcal{P}$ is k-list online learnable if there exists a learning rule $\mathcal{A}$ and $M \in \mathbb{N}$ such that $\mathtt{M}(\mathcal{A}; S) \leq M$ for every input sequence S which is realizable by $\mathcal{P}$. An hypothesis class $\mathcal{H}$ is k-list online learnable if $\mathcal{P}(\mathcal{H})$ is k-list online learnable. Let $\mathtt{M}_k(\mathcal{P})$ denote the optimal mistake bound achievable in learning $\mathcal{P}$ by deterministic learners, and let $\mathtt{R}_k(\mathcal{P})$ denote the optimal expected mistake bound in learning $\mathcal{P}$ by randomized learners.

Agnostic Case Learnability (Hypothesis Classes). Pattern classes are suitable to define a learning task in terms of its possible input sequences (patterns). While it is possible to also define agnostic online learning with respect to pattern classes, we find it more natural to stick to the traditional

approach and define it with respect to hypothesis classes. For example, hypothesis classes allows to explicitly study the tradeoff between the list-size used by the learning rule, and the list size used by the functions in the class.

Let h be a k multi-labeled hypothesis. For a sequence $S = \{(x_t, y_t)\}_{t=1}^{T}$ of examples, we let $\mathtt{M}(h; S) = \sum_{t=1}^{T} 1[y_t \notin h(x_t)]$ denote the number of mistakes h makes on S. Let $\mathcal{H}$ be a multi-labeled hypothesis class and let $\mathcal{A}$ be a k' list learning rule (notice that k' might be different than k). For a sequence $S = \{(x_t, y_t)\}_{t=1}^{T}$ of examples, denote the regret of $\mathcal{A}$ with respect to $\mathcal{H}$ on S by

$$\mathtt{R}_{\mathcal{H}}(\mathcal{A}; S) = \mathtt{M}(\mathcal{A}; S) - \min_{h \in \mathcal{H}} \mathtt{M}(h; S)$$

For an integer T, we let $\mathtt{R}_{\mathcal{H}}(\mathcal{A}; T) = \max\{\mathtt{R}_{\mathcal{H}}(\mathcal{A}; S) : |S| = T\}$. We say that $\mathcal{H}$ is k'-list agnostic learnable if there exists a learning rule $\mathcal{A}$ and a sublinear function $R(T) = o(T)$ such that $\mathtt{R}_{\mathcal{H}}(\mathcal{A}; T) \leq R(T)$ for every T.

Mistake Trees and Littlestone Dimension. A b-ary mistake tree is a decision tree in which each internal node has out-degree b. Each internal node v in the tree is associated with an unlabeled example $x(v) \in \mathcal{X}$ and each edge e is associated with a label $y(e) \in \mathcal{Y}$ such that for every internal node v all its b out-going edges are associated with different labels from $\mathcal{Y}$. A branch is a root-to-leaf path in the tree. Notice that every branch

$$v_0 \to_{e_0} v_1 \to_{e_1} \ldots \to_{e_r} v_{r+1}$$

naturally induces a sequence of examples $\{(x_i, y_i)\}_{i=0}^{r}$ where $x_i = x(v_i)$ and $y_i = y(v_i \to v_{i+1})$ (Figure 2).

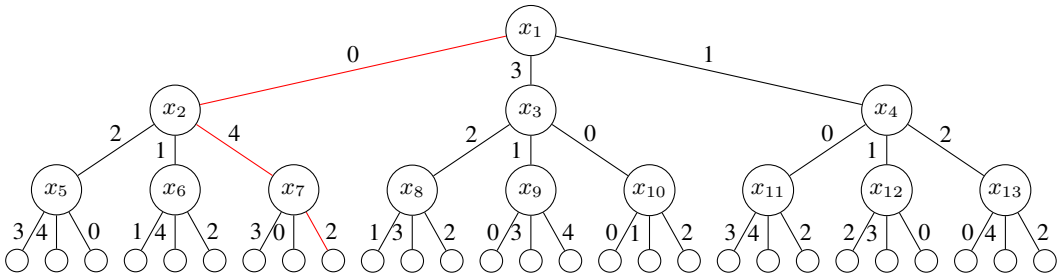

Figure 2: An example of a 3-ary mistake tree, over the label set $\{0, 1, 2, 3, 4\}$. The tree has 13 internal nodes, each associated with an unlabeled example $x_i \in \mathcal{X}$. The red marked path defines the sequence $(x_1, 0), (x_2, 4), (x_7, 2)$.

We say that a mistake tree $\mathcal{T}$ is shattered by a pattern class $\mathcal{P}$ if every branch in $\mathcal{T}$ induces a sequence of examples in $\mathcal{P}$. A mistake tree $\mathcal{T}$ is shattered by a (possibly multi-labeled) hypothesis class $\mathcal{H}$ if it is shattered by $\mathcal{P}(\mathcal{H})$. The b-ary Littlestone dimension of a pattern class $\mathcal{P}$, denoted $\mathtt{L}_{b-1}(\mathcal{P})$, is the largest possible depth of a complete b-ary mistake tree that is shattered by $\mathcal{P}$. (Thus, $\mathtt{L}_1(\cdot)$ is the classical Littlestone dimension.) If $\mathcal{P}$ shatters such trees of arbitrarily large depths then we define $\mathtt{L}_{b-1}(\mathcal{P}) = \infty$. The b-ary Littlestone dimension of a (multi-labeled) hypothesis class $\mathcal{H}$ is $\mathtt{L}_{b-1}(\mathcal{H}) := \mathtt{L}_{b-1}(\mathcal{P}(\mathcal{H}))$.

3. Results

3.1. Warmup: List Perceptron

As a warmup, we begin with a natural adaptation of the classical Perceptron algorithm by Rosenblatt (1958) to the list online learning setting. Our list Perceptron algorithm may be seen as a natural adaptation of the multiclass Perceptron algorithm by Kesler (Duda and Hart, 1973), and the more recent variants by Crammer and Singer (2003) and by Beygelzimer, Pál, Szörényi, Thiruvenkatachari, Wei, and Zhang (2019).

Mutliclass Linear Separability. Let $\mathcal{X} = \mathbb{R}^d$ and let $\mathcal{Y}$ denote the label space. A sequence of examples $S = \{(x_t, y_t)\}_{t=1}^T$ is called linearly separable if for every label $y \in \mathcal{Y}$ there exists a $w_y^\star \in \mathbb{R}^d$ such that for every $t = 1, \ldots, T$:

$$(\forall y \neq y_t) : w_{y_t}^\star \cdot x_t > w_y^\star \cdot x_t, \tag{1}$$

where "$\cdot$" denotes the standard inner product in $\mathbb{R}^d$. In words, the label y of each point x corresponds to the direction $w_y^\star$ on which x has the largest projection. We follow standard convention and assume that the $w_y^\star$'s are normalized such that

$$\sum_{y \in \mathcal{Y}} \|w_y^\star\|^2 = 1, \tag{2}$$

where $\|\cdot\|$ denotes the ℓ_2 euclidean norm. If a sequence of vectors $w_y^\star$ for $y \in \mathcal{Y}$ satisfies Equations 1 and 2 above then we say that it *linearly separates* S.

Multiclass Margin. For $k > 0$, the k'th margin of a separable sequence $S = \{(x_t, y_t)\}_{t=1}^T$, denoted by $\gamma_k(S)$ is the maximal real number $\gamma_k > 0$ for which there exist vectors $w_y^\star$ for $y \in \mathcal{Y}$ which linearly separate S such that for every $t \leq T$,

$$w_{y_t}^\star \cdot x_t - w_y^\star \cdot x_t \geq \gamma_k, \text{ for all but at most } k \text{ labels } y \in \mathcal{Y}.$$

So, if we sort the labels y's according to the inner product $w_y^\star \cdot x_t$ then y_t is the largest and the $(k+1)$'th largest label y in the sorted list satisfies $(w_{y_t}^\star - w_y^\star) \cdot x_t \geq \gamma_k$.

Notice that $\gamma_1 \leq \gamma_2 \leq \ldots$ and that γ_1 is the standard margin in multiclass linear classification.

Theorem 1 (List Perceptron Mistake Bound) *Let $S = \{(x_t, y_t)\}_{t=1}^T$ be a linearly separable input sequence and let $R = \max_{t \leq T} \|x_t\|$. Then, the k-list Perceptron algorithm (Figure 3) makes at most $k(k+1)\frac{R^2}{\gamma_k^2}$ mistakes on S, where $\gamma_k = \gamma_k(S)$ is the k'th margin of S.*

We prove Theorem 1 in Section A.1. The proof follows an adaptation to the list setting of the classical analysis of the Perceptron mistake bound (Rosenblatt, 1958; Crammer and Singer, 2003; Beygelzimer et al., 2019).

Note that we can also state Theorem 1 in the language of pattern classes. For $R, \gamma > 0$ let $\mathcal{P} = \mathcal{P}(R, \gamma_k)$ denote the set of all patterns $S = \{(x_t, y_t)\}_{t=1}^T$ whose k'th margin is at least γ_k and such that $\|x_t\| \leq R$ for all $t \leq T$. Thus, Theorem 1 implies that $\mathcal{P}$ is online learnable with mistake bound $\leq k(k+1)\frac{R^2}{\gamma_k^2}$.

List Perceptron Algorithm

Parameters: $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y}$ = label space, and k = list size.
Input: A linearly separable sequence S of length T.
Initialize: For all $y \in \mathcal{Y}$ set $w_y = \mathbf{0}$.
For $t = 1, \ldots, T$

1. Receive unlabeled example $x_t \in \mathcal{X}$.
2. Sort the labels $y \in \mathcal{Y}$ in a non-increasing order according to the inner product $w_y \cdot x_t$.
3. Predict the list P_t which consists of the top k labels in the above order.
4. Receive correct label $y_t \in \mathcal{Y}$.
5. If $y_t \notin P_t$ then update the w_y's as follows:
   - $w_{y_t} \leftarrow w_{y_t} + kx_t$,
   - $w_y \leftarrow w_y - x_t$ for all $y \in P_t$.

Figure 3: List Perceptron Algorithm

3.2. Realizable Case

We now turn to characterize list online learnability in the realizable setting. We will state our results in the general framework of pattern classes which generalizes the traditional framework of hypothesis classes, as detailed above.

3.2.1. QUALITATIVE CHARACTERIZATION

Theorem 2 (Pattern Classes) *Let $\mathcal{P}$ be a pattern class, and let $k \in \mathbb{N}$ denote the list-size parameter. Then, the following statements are equivalent:*

1. *$\mathcal{P}$ is k-list online learnable in the realizable setting.*
2. *The $(k+1)$-ary Littlestone dimension of $\mathcal{P}$ is finite: $\mathtt{L}_k(\mathcal{P}) < \infty$.*

As a corollary, we deduce the same characterization for (multi-labeled) hypothesis classes.

Corollary 3 (Hypothesis Classes) *Let $\mathcal{H}$ be a k' multi-labeled hypothesis class, and let $k \in \mathbb{N}$ denote the list-size parameter. Then, the following statements are equivalent:*

1. *$\mathcal{H}$ is k-list online learnable in the realizable setting.*
2. *The $(k+1)$-ary Littlestone dimension of $\mathcal{H}$ is finite: $\mathtt{L}_k(\mathcal{H}) < \infty$.*

Proof Notice that $\mathcal{H}$ is k-list online learnable if and only if $\mathcal{P}(\mathcal{H})$ is, and that $\mathtt{L}_k(\mathcal{H}) = \mathtt{L}_k(\mathcal{P}(\mathcal{H}))$. Thus, the corollary follows from Theorem 2. ■

List SOA

Parameter: $k =$ list size.
Input: A pattern class $\mathcal{P}$ with domain $\mathcal{X}$ and label space $\mathcal{Y}$.
Initialize: $V = \mathcal{P}$.
For $t = 1, 2, \ldots$

1. Receive unlabeled example $x_t \in \mathcal{X}$.
2. Sort the labels $y \in \mathcal{Y}$ in a non-increasing order according to the values $\mathtt{L}_k(V_{x_t \to y})$.
3. Predict the list P_t which consists of the top k labels in the above order.
4. Receive correct label $y_t \in \mathcal{Y}$.
5. If $y_t \notin P_t$ then update $V \leftarrow V_{x_t \to y_t}$ [a].

a. One may wonder, why V is not updated at each time step. The reason is because in the proof of the list version of the SSP lemma (See Section A.3), we use the List SOA to construct an online cover, and this construction exploits the property that V is changed only upon making a mistake.

Figure 4: List Standard Optimal Algorithm (SOA)

3.2.2. QUANTITATIVE MISTAKE BOUNDS

Theorem 4 (Optimal Mistake Bound) *Let $\mathcal{P}$ be a pattern class and recall that $\mathtt{M}_k(\mathcal{P})$ is the optimal (deterministic) mistake bound in k-list learning $\mathcal{P}$ in the realizable case. Then,*

$$\mathtt{M}_k(\mathcal{P}) = \mathtt{L}_k(\mathcal{P}).$$

Further, the optimal expected mistake bound for randomized algorithms, $\mathtt{R}_k(\mathcal{P})$, satisfies

$$\mathtt{M}_k(\mathcal{P}) \geq \mathtt{R}_k(\mathcal{P}) \geq \frac{1}{k+1}\mathtt{M}_k(\mathcal{P}),$$

and both inequalities can be tight.

As before, this theorem applies verbatim to (multi-labeled) hypothesis classes.

The lower bound in Theorem 4 follows directly from the definition of mistake trees and $\mathtt{L}_k(\mathcal{P})$. The upper bound relies on an adaptation of Littlestone's Standard Optimal Algorithm (SOA) to the list learning setting and to pattern class (see Figure 4).

3.3. Agnostic Case

We next turn to the agnostic case where we lift the assumption that the input sequence is realizable by the class. Consequently, the goal of the learner is also adapted to achieve a mistake bound which is competitive with the best function in the class. (See Section 2 for the formal definition.) Thus, in the agnostic setting the class is viewed as a *comparator* class with respect to which the algorithm should achieve vanishing regret.

In this section we address the following questions: can every (multi-labeled) hypothesis class $\mathcal{H}$ that is learnable in the realizable setting be learned in the agnostic setting? How do the regret bounds behave as a function of the list size used by the learner? E.g. say $\mathcal{H}$ is $k = 2$-list learnable and that the learner $\mathcal{A}$ uses lists of size 3. Is it possible to achieve negative regret in this case?

Notice that we model the agnostic setting using hypothesis classes and not pattern classes. We made this choice because we find it more natural to address the above questions, specifically regarding the way the regret depends on the list size of the algorithm and the number of labels each hypothesis $h \in \mathcal{H}$ assigns to a given point.

3.3.1. CHARACTERIZATION

We show a similar qualitative result for the agnostic case. That is, a class is agnostic k-list online learnable if and only if its $(k+1)$-ary Littlestone dimension is finite.

Theorem 5 *Let $\mathcal{H}$ be a (possibly multi-labeled) hypothesis class, and let $k \in \mathbb{N}$. Then, the following statements are equivalent:*

1. *$\mathcal{H}$ is k-list online learnable in the agnostic setting.*
2. *The $(k+1)$-ary Littlestone dimension of $\mathcal{H}$ in finite, $\mathtt{L}_k(\mathcal{H}) < \infty$.*

Quantitatively we get the following upper bound on the regret

$$O\left(\sqrt{dT + dT\ln\left(\frac{T}{d}\left\lceil\frac{L-k}{k}\right\rceil k\right)}\right), \tag{3}$$

where $d = \mathtt{L}_k(\mathcal{H})$, and $L = |\mathcal{Y}|$.

The proof of Theorem 5 follows an adaptation to the list setting of the agnostic-to-realizable reduction by Ben-David, Pál, and Shalev-Shwartz (2009); Daniely, Sabato, Ben-David, and Shalev-Shwartz (2015), which hinges on the List Online Sauer-Shelah-Perles Lemma. We note that like in classical online learning, agnostic list online learners must be randomized.

3.3.2. PRIDE AND REGRET

Let $\mathcal{H}$ be an hypothesis class such that $\mathtt{L}_k(\mathcal{H}) < \infty$. Theorem 5 implies that there is a k-list learner that agnostically learns $\mathcal{H}$ with vanishing regret. How does the regret changes if we use more resourceful list learners, with list size larger than k? Can it become negative? The following theorem answers this question. In a nutshell, it shows that once the algorithm has more resources than necessary for learning $\mathcal{H}$ in the realizable setting, then it can achieve negative regret.

Theorem 6 *Let $\mathcal{H}$ be a multi-labeled hypothesis class, and assume that $|\mathcal{H}| > 1$. Let $1 \leq \mathtt{k}_{\mathcal{H}} \leq |\mathcal{Y}|$ be the minimal number such that $\mathtt{L}_{\mathtt{k}_{\mathcal{H}}}(\mathcal{H}) < \infty$. For a list-learning rule $\mathcal{A}$ let $\mathtt{k}_{\mathcal{A}} \leq |\mathcal{Y}|$ denote the size of the lists it uses. Then, if $\mathcal{A}$ agnostically learns $\mathcal{H}$ then necessarily $\mathtt{k}_{\mathcal{A}} \geq \mathtt{k}_{\mathcal{H}}$ and the following holds in these cases:*

1. **Negative Regret.** *Let $|\mathcal{Y}| \geq k > \mathtt{k}_{\mathcal{H}}$, then there exists a learning rule $\mathcal{A}$ with $\mathtt{k}_{\mathcal{A}} = k$ which learns $\mathcal{H}$ with a negative regret in the following sense: its expected regret on every input sequence of length T is at most*

$$(p-1) \cdot \mathtt{opt}_{\mathcal{H}} + c \cdot \left(\sqrt{T \ln T}\right),$$

where $p = p(\mathcal{H}) < 1$, $c = c(\mathcal{H})$ both do not depend on T, and $\mathtt{opt}_{\mathcal{H}}$ is the number of mistakes made by the best function in $\mathcal{H}$. (Note that when $\mathtt{opt}_{\mathcal{H}} = \Omega(T)$ then the regret is indeed negative.)

2. **Nonnegative Regret.**[3] *Every learning rule $\mathcal{A}$ with $\mathtt{k}_{\mathcal{A}} = \mathtt{k}_{\mathcal{H}}$ has non-negative regret: there exists $p' = p'(\mathcal{H}) > 0$ so that for every $p \leq p'$ and every T there exists an input sequence of length T such that $\mathtt{opt}_{\mathcal{H}} \leq \lfloor p \cdot T \rfloor$ but the expected number of mistakes made by $\mathcal{A}$ is $\geq \lfloor p \cdot T \rfloor$.*

3. **Unbounded Regret.** *If in addition to Item 2, there are $h', h'' \in \mathcal{H}$ and an input x such that $h'(x), h''(x)$ are distinct lists of size $\mathtt{k}_{\mathcal{H}}$ then every learning rule $\mathcal{A}$ with $\mathtt{k}_{\mathcal{A}} = \mathtt{k}_{\mathcal{H}}$ has regret at least $c \cdot \sqrt{T}$, where $c = c(\mathcal{H})$ does not depend on T.*

Item 2 is tight in the sense that there is a class $\mathcal{H}$ satisfying the assumption in Item 2 which can be learned by a learning rule $\mathcal{A}$ with $\mathtt{k}_{\mathcal{A}} = \mathtt{k}_{\mathcal{H}}$ that has a non-positive regret. Indeed, let $\mathcal{X} = \mathbb{N}$ and $\mathcal{Y} = \{0, 1, 2, 3\}$, and consider $\mathcal{H} = \{0, 1\}^{\mathbb{N}}$. Thus, $\mathtt{k}_{\mathcal{H}} = 2$ and all functions in $\mathcal{H}$ map each $x \in \mathcal{X}$ to a single output in $\{0, 1\}$ (and hence the additional assumption in Item 3 does not apply). Finally, notice that the learning rule $\mathcal{A}$ which always predicts with the list $\{0, 1\}$ achieves regret ≤ 0.

On the other hand, there are also such classes (i.e. that satisfy the assumption in Item 2 but not the additional assumption in Item 3) for which the optimal regret does scale like $\Omega(\sqrt{T})$. One such example is $\mathcal{H} = \{0, 1\}^{\mathcal{X}} \cup \{2, 3\}^{\mathcal{X}}$. It will be interesting to refine the above theorem and characterize, for every class $\mathcal{H}$, the optimal regret achievable by learning rules $\mathcal{A}$ for which $\mathtt{k}_{\mathcal{A}} = \mathtt{k}_{\mathcal{H}}$.

The proof of Theorem 6 appears in Section A.5. It combines a variety of techniques: in Item 1 the idea is to combine two algorithms: one which uses lists of size $\mathtt{k}_{\mathcal{H}}$ and has vanishing regret, and another that uses additional $k - \mathtt{k}_{\mathcal{H}} > 0$ labels that were not chosen by the first algorithm in a way that achieves the overall regret. In fact, we show that for the second algorithm, we can simply pick the $k - \mathtt{k}_{\mathcal{H}}$ labels uniformly at random from the remaining labels. The proof of Item 2 follows by a randomized construction of a hard input sequence which witnesses the nonngegative regret. In particular we start by taking a shattered $\mathtt{k}_{\mathcal{H}}$-tree of depth T (such a tree exists for all T because $\mathtt{L}_{\mathtt{k}_{\mathcal{H}}-1}(\mathcal{H}) = \infty$), then pick a sequence corresponding to a random branch in the tree, and carefully modify this random sequence in $p \cdot T$ random locations. The proof of Item 3 also follows by a randomized construction of a hard input sequence. The argument here generalizes the classical lower bound of $\Omega(\sqrt{T})$ for any class $\mathcal{H}$ which consists of at least 2 functions. However, the proof here is a little bit more subtle.

3.4. Online List Sauer-Shelah-Perles Lemma

A central tool in the derivation of our upper bound in the agnostic setting is a covering lemma, which takes a realizable case learner for a pattern class $\mathcal{P}$ and uses it to construct a small collection of online/adaptive functions which cover $\mathcal{P}$. We begin with introducing some definitions and notations.

3. We comment that in the realizable case every learning algorithm trivially has nonnegative regret. Thus, notice that this item states a more elaborate statement which demonstrates a non-trivial sense in which the regret is non-negative.

Online Functions. An online (or adaptive) function is a mapping $f : \mathcal{X}^\star \to \mathcal{Y}$. That is, its input is a finite sequence of points $x_1, \ldots, x_{t-1}, x \in \mathcal{X}$ and its output is a label $y \in \mathcal{Y}$ which we denote by $y = f(x; x_1, \ldots, x_{t-1})$. We interpret this object as a function with memory: the points $x_1, \ldots x_{t-1}$ are the points that f has already seen and stored in its memory, and y is the label it assigns to the current point x. Similarly, a k-list online function is a mapping $f : \mathcal{X}^\star \to \binom{\mathcal{Y}}{k}$.

Definition 7 (Online Coverings) *Let $\mathcal{P}$ be a pattern class and let $T \in \mathbb{N}$. We say that a family $\mathcal{F}$ of k-list online functions is a T-cover of $\mathcal{P}$ if for every pattern $\{(x_t, y_t)\}_{t=1}^{T} \in \mathcal{P}$ of length T there exists a k-list online function $f \in \mathcal{F}$ such that*

$$(\forall t \leq T) : y_t \in f(x_t; x_1, \ldots, x_{t-1}).$$

Proposition 1 (List Sauer-Shelah-Perles Lemma) *Let $\mathcal{P}$ be a pattern class with $\mathtt{L}_k(\mathcal{P}) = d < \infty$, and let $L = |\mathcal{Y}|$. Then, for every $T \in \mathbb{N}$ there is a family $\mathcal{F}$ of k-list online functions which is a T-cover of $\mathcal{P}$ such that*

$$|\mathcal{F}| \leq \sum_{i=0}^{d} \binom{T}{i} \left(\left\lceil \frac{L-k}{k} \right\rceil k \right)^i \approx \left(e \cdot \frac{T}{d} L \right)^d.$$

The construction of the covering family $\mathcal{F}$ follows a similar approach like in Ben-David et al. (2009). In particular it relies on our lazy List SOA (lazy in the sense that it only changes its predictor when making a mistake). Then, each k-list online function in $\mathcal{F}$ simulates the List SOA, on all but at most d steps, aiming to fully cover all patterns in $\mathcal{P}$.

4. Future Research

There are plenty of directions for keep exploring list online learning, here we consider a few natural ones that arise in this work.

1. Our characterization of the agnostic case only applies when the label space is finite. (Quantitatively, our upper bound on the regret depends on the size of the label space.) Does the characterization extends to infinite label space? More generally, it will be interesting to derive tighter lower and upper bounds on the optimal regret.

2. Study specific list learning tasks. For example, for linear classification with margin, we demonstrated an upper bound using the list Perceptron algorithm. It will be interesting to determine whether this algorithm achieves an optimal mistake bound, or perhaps there are better algorithms?

3. What is the optimal expected mistake bound for randomized list learners? Is there a natural description of an optimal randomized list learner?

4. We study list online learning in the context of multiclass online prediction using the 0/1 classification loss. Extensions to weighted and continuous losses are left for future research.

5. Study the expressivity of pattern classes (see Section B).

Acknowledgments

We thank Amir Yehudayoff for insightful discussions.

Shay Moran is a Robert J. Shillman Fellow; he acknowledges support by ISF grant 1225/20, by BSF grant 2018385, by an Azrieli Faculty Fellowship, by Israel PBC-VATAT, by the Technion Center for Machine Learning and Intelligent Systems (MLIS), and by the the European Union (ERC, GENERALIZATION, 101039692). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

References

Noga Alon, Steve Hanneke, Ron Holzman, and Shay Moran. A theory of PAC learnability of partial concept classes. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 658–671. IEEE, 2021. doi: 10.1109/FOCS52979.2021.00070. URL https://doi.org/10.1109/FOCS52979.2021.00070.

Anastasios N. Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *CoRR*, abs/2107.07511, 2021. URL https://arxiv.org/abs/2107.07511.

Peter Auer and Philip M Long. Structural results about on-line learning models with and without queries. *Machine Learning*, 36(3):147–181, 1999.

Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic online learning. December 2009. 22nd Conference on Learning Theory, COLT 2009 ; Conference date: 18-06-2009 Through 21-06-2009.

Alina Beygelzimer, Dávid Pál, Balázs Szörényi, Devanathan Thiruvenkatachari, Chen-Yu Wei, and Chicheng Zhang. Bandit multiclass linear classification: Efficient algorithms for the separable case. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 624–633. PMLR, 2019. URL http://dblp.uni-trier.de/db/conf/icml/icml2019.html#BeygelzimerPSTW19.

Nataly Brukhim, Daniel Carmon, Irit Dinur, Shay Moran, and Amir Yehudayoff. A characterization of multiclass learnability. *CoRR*, abs/2203.01550, 2022. doi: 10.48550/arXiv.2203.01550. URL https://doi.org/10.48550/arXiv.2203.01550.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Moses Charikar and Chirag Pabbaraju. A characterization of list learnability. *CoRR*, abs/2211.04956, 2022. doi: 10.48550/arXiv.2211.04956. URL https://doi.org/10.48550/arXiv.2211.04956.

Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.*, 3(null):951–991, mar 2003. ISSN 1532-4435. doi: 10.1162/jmlr.2003.3.4-5.951. URL https://doi.org/10.1162/jmlr.2003.3.4-5.951.

Amit Daniely, Sivan Sabato, Shai Ben-David, and Shai Shalev-Shwartz. Multiclass learnability and the erm principle. *J. Mach. Learn. Res.*, 16(1):2377–2404, jan 2015. ISSN 1532-4435.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Willey & Sons, New Yotk, 1973.

Elad Hazan. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*, 2019.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. In *Machine Learning*, pages 285–318, 1988.

Philip Long. On agnostic learning with $\{0, *, 1\}$-valued and real-valued hypotheses. In *Proceedings of the 14th Annual Conference on Learning Theory and 5th European Conference on Computational Learning Theory*, 2001.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958. ISSN 0033-295X. doi: 10.1037/h0042519. URL http://dx.doi.org/10.1037/h0042519.

Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *J. Mach. Learn. Res.*, 9:371–421, 2008. doi: 10.5555/1390681.1390693. URL https://dl.acm.org/doi/10.5555/1390681.1390693.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Found. Trends Mach. Learn.*, 4(2):107–194, 2012. doi: 10.1561/2200000018. URL https://doi.org/10.1561/2200000018.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 0387001522.

Appendix A. Proofs

A.1. List Perceptron

Proof [Proof of Theorem 1] We follow the classical analysis of the Perceptron by upper and lower bounding the sum $\sum_y \|w_y\|^2$ at the end of round T. This sum only changes on rounds t in which the algorithm makes a mistake. In such rounds, the vectors w_y are updated to w'_y as follows

$$\begin{aligned} w'_{y_t} &= w_{y_t} + kx_t, \\ w'_y &= w_y - x_t. \qquad \text{(for all } y \in P_t\text{)} \end{aligned}$$

Thus, the change in the potential function from round t to $t+1$ is upper bounded as follows

$$\begin{aligned}
\sum_{y\in\mathcal{Y}} \|w'_y\|^2 - \sum_{y\in\mathcal{Y}} \|w_y\|^2 &= \left(\|w'_{y_t}\|^2 - \|w_{y_t}\|^2\right) + \left(\sum_{y\in P_t} \|w'_y\|^2 - \|w_y\|^2\right) \\
&= k^2\|x_t\|^2 + 2k(w_{y_t}\cdot x_t) + \sum_{y\in P_t}(\|x_t\|^2 - 2w_y\cdot x_t) \\
&= k(k+1)\|x_t\|^2 + 2\sum_{y\in P_t}(w_{y_t} - w_y)\cdot x_t \\
&\le k(k+1)\|x_t\|^2 && (w_{y_t}\cdot x_t \le w_y\cdot x_t) \\
&\le k(k+1)R^2. && (\|x_t\|\le R)
\end{aligned}$$

Thus, if we denote by M the total number of mistakes that the Perceptron makes on S then the potential function at time T satisfies:

$$\sum_{y\in\mathcal{Y}} \|w_y\|^2 \le M\cdot k(k+1)R^2. \tag{4}$$

We now lower bound the potential function. Let $w^\star_y$ for $y\in\mathcal{Y}$ denote the unit vectors that witness that S is linearly separable with k-margin at least γ_k. Consider the sum of inner products $\sum_{y\in\mathcal{Y}} w_y\cdot w^\star_y$. This sum also changes only on rounds t when a mistake occurs, hence

$$\begin{aligned}
\sum_{y\in\mathcal{Y}} w'_y\cdot w^\star_y - \sum_{y\in\mathcal{Y}} w_y\cdot w^\star_y &= kx_t\cdot w^\star_{y_t} - \sum_{y\in P_t} x_t\cdot w^\star_y \\
&= \sum_{y\in P_t}(w^\star_{y_t} - w^\star_y)x_t \ge \gamma_k. && (\text{because } \|P_t\| = k)
\end{aligned}$$

Thus, after round T we have $M\gamma_k \le \sum_{y\in\mathcal{Y}} w_y\cdot w^\star_y$ and hence:

$$\begin{aligned}
M\gamma_k &\le \sum_{y\in\mathcal{Y}} w_y\cdot w^\star_y \le \sum_{y\in\mathcal{Y}} \|w_y\|\cdot\|w^\star_y\| && (\text{Cauchey-Schwartz}) \\
&\le \sqrt{\sum_{y\in\mathcal{Y}}\|w_y\|^2}\cdot\sqrt{\sum_{y\in\mathcal{Y}}\|w^\star_y\|^2} = \sqrt{\sum_{y\in\mathcal{Y}}\|w_y\|^2}. && (\text{Cauchey-Schwartz})
\end{aligned}$$

Hence, $\sum_{y\in\mathcal{Y}}\|w_y\|^2 \ge M^2\cdot\gamma_k^2$, which in combination with Equation 4 yields $M\le k(k+1)\frac{R^2}{\gamma_k^2}$ as required.

■

A.2. Realizable Case

The qualitative statement of Theorem 2 is just a corollary of the quantitative statement given by Theorem 4.

Proof [Proof of Theorem 4] The lower bound follows directly from the definition of mistake trees and $\mathrm{L}_k(\mathcal{P})$. Let $\mathcal{A}$ be a deterministic k-list learner, and let $\mathcal{T}$ be a shattered complete $(k+1)$-ary mistake tree of depth n. We will build an input sequence by following the adversary on the shattered

tree $\mathcal{T}$. Starting from the root, we will choose a label for each node x_i in the root-to-leaf path, associated with one edge from its $k+1$ out-going edges, so that the chosen label will not be in the predicted set of the learner $\mathcal{A}$. We can always do that because the predicted set in of size k and for each node in the tree there are $k+1$ out-going edges (with different labels). With this construction we will get a sequence of size n which is realizable by $\mathcal{P}$ and forces $\mathcal{A}$ to always err and hence to make n mistakes on the input-sequence. Therefore, for finite $(k+1)$-ary Littlestone dimension d, we can construct a realizable sequence of length d such that $\mathcal{A}$ makes at least d mistakes on S, and for infinite $(k+1)$-ary Littlestone dimension, we can construct a realizable sequence of length n, for each $n \in \mathbb{N}$ such that $\mathcal{A}$ makes at least n mistakes on S.

Corollary 8

$$\mathtt{M}_k(\mathcal{P}) \geq \mathtt{L}_k(\mathcal{P}).$$

The lower bound for randomized learners is achieved in a very similar way. We still follow a root-to-leaf path in a shattered tree, but now, instead of choosing the unchosen label by the learner, we choose a *random* label, so that at each step, the algorithm makes a mistake with probability $\frac{1}{k+1}$.

Corollary 9

$$\mathtt{R}_k(\mathcal{P}) \geq \frac{1}{k+1}\mathtt{L}_k(\mathcal{P}).$$

The upper bound relies on an adaptation of Littlestone's Standard Optimal Algorithm (SOA) to the list learning setting and to pattern class, described in Figure 4.

Claim 1 *Let $\mathcal{P}$ be a pattern class such that $\mathtt{L}_k(\mathcal{H}) < \infty$ and let $x \in \mathcal{X}$. Then the number of $y \in \mathcal{Y}$ such that $\mathtt{L}_k(\mathcal{P}_{x\to y}) = \mathtt{L}_k(\mathcal{P})$ is at most k.*

Proof Assume by contradiction that there exists a subset $I \subseteq \mathcal{Y}, |I| = k+1$ such that $\forall y \in I$, $\mathtt{L}_k(\mathcal{P}_{x\to y}) = \mathtt{L}_k(\mathcal{P}) =: d$. Then we will construct the following tree. Let $\{\mathcal{T}_y\}_{y\in I}$ be complete $(k+1)$-ary mistake trees with depth d shattered by $\{\mathcal{P}_{x\to y}\}_{y\in I}$ respectively. Construct a tree with x as a root, where its outgoing edges are labeled with the labels in I and $\{\mathcal{T}_y\}_{y\in I}$ are the subtrees under x where the subtree under each outgoing edge with label $y \in I$ is $\mathcal{T}_y$. With this construction we get a complete $(k+1)$-ary mistake tree of depth $d+1$ which is shattered by $\mathcal{P}$ which contradicts with the assumption that d is the maximal depth of a complete $(k+1)$-ary mistake tree which is shattered by $\mathcal{P}$. ■

Claim 2 *The List SOA makes at most $\mathtt{L}_k(\mathcal{P})$ mistakes on every realizable sequence.*

Proof Note that at each time step where the List SOA makes a mistake, the $(k+1)$-ary Littlestone dimension of V is decreasing. This is because of Claim 1 combined with the construction of the predicted list P_t to contain the k labels with maximal value $\mathtt{L}_k(V)$. Since the $(k+1)$-ary Littlestone dimension is bounded below by 0, it can only decrease at most $\mathtt{L}_k(\mathcal{P})$ times. Therefore, the number of mistakes the algorithm makes is bounded above by $\mathtt{L}_k(\mathcal{P})$. ■

Corollary 10

$$\mathtt{M}_k(\mathcal{P}) \leq \mathtt{L}_k(\mathcal{P}).$$

The upper bound of $\mathtt{L}_k(\mathcal{P})$ achieved by the List SOA, completes the characterization of list learnability for realizable sequences, and shows that the optimal mistake bound of deterministic learner is exactly the $(k+1)$-ary Littlestone dimension $\mathtt{L}_k(\mathcal{P})$.

For randomized learners, the lower and upper bounds of the optimal number of mistakes, provided in Theorem 4 are tight;

- There is a k-list learnable pattern class $\mathcal{P}$, such that for every k-list learner $\mathcal{A}$, there exists a realizable sequence S, with $\mathtt{M}(\mathcal{A}; S) \geq \mathtt{L}_k(\mathcal{P})$.

- There is a k-list learnable pattern class $\mathcal{P}$, and a k-list learner $\mathcal{A}$, such that for every realizable sequence S, $\mathtt{M}(\mathcal{A}; S) \leq \frac{1}{k+1} L_k(\mathcal{P})$.

See Appendix C for the construction of these extremal classes, and the analysis of their optimal mistake bounds.

■

A.3. Online Sauer-Shelah-Perles Lemma

Proof [Proof of Proposition 1] The construction of the covering family $\mathcal{F}$ follows a similar approach like in Ben-David et al. (2009). In particular it relies on our lazy List SOA (lazy in the sense that it only changes its predictor when making a mistake). Each k-list online function in $\mathcal{F}$ simulates the List SOA, on all but at most d steps, aiming to fully cover all patterns in $\mathcal{P}$.

We now describe the family $\mathcal{F}$. Since our goal is to cover realizable sequences of length T, it suffices to define each $f \in \mathcal{F}$ on input sequences in $\mathcal{X}^\star$ of size at most T.[4] In what follows, it is convenient to fix a linear order on the label set $\mathcal{Y}$ (e.g. one can think of $\mathcal{Y}$ as $\mathcal{Y} = \{0, \ldots, L-1\}$). Also, for every $A \subseteq \mathcal{Y}$, fix a covering $\{A_1, \ldots, A_m\}$ of A into $m = \lceil |A|/k \rceil$ sets, where each set has size k (e.g., the first k elements in A form the first part, the second k elements form the next part, and so on; if the last part has size l less than k, then complete it with the first $k-l$ elements to get a size of k). We will refer to this fixed cover as *the canonical k-cover of A*. Each $f \in \mathcal{F}$ is parameterized by a subset $|I| \subset [T]$ of size $|I| \leq d$ and a vector of pairs $\big((i_t, j_t)\big)_{t \in I}$, where $i_t \in \{1, \ldots, k\}$, and $j_t \in \{1, \ldots, \lceil \frac{L-k}{k} \rceil\}$. Since there are $\lceil \frac{L-k}{k} \rceil k$ ways to pick a given pair (i_t, j_t), we get

$$|\mathcal{F}| = \sum_{i=0}^{d} \binom{T}{i} \left(\left\lceil \frac{L-k}{k} \right\rceil k \right)^i.$$

Each such function f is defined as follows:

4. If $t > T$ then set $f(x_t; x_1, \ldots, x_{t-1}) = 0$ for all $f \in \mathcal{F}$ and all $x_1, \ldots, x_t \in \mathcal{X}$.

Parameters: $T \in \mathbb{N}$, $\big(I, \{(i_t, j_t)\}_{t \in I}\big)$, where $I \subseteq [T]$ and $i_t \leq k, j_t \leq \lceil \frac{L-k}{k} \rceil$.
Input: An input sequence of unlabeled examples $\boldsymbol{x} = \{x_t\}_{t=1}^{T}$ of length T.
Initialize: $S = \emptyset$. For $t = 1, 2, \ldots, T$

1. Get x_t and let $P_t = \text{ListSOA}(x_t; S)$.
2. If $t \notin I$ then:
   - Set $f(x_t; \boldsymbol{x}_{<t}) = P_t$.
   - Set y_t to be any label in $f(x_t; \boldsymbol{x}_{<t})$.
3. If $t \in I$ then:
   - Set $f(x_t; \boldsymbol{x}_{<t})$ to be set number j_t in the canonical k-cover of $\mathcal{Y} \setminus P_t$.
   - Set y_t to be the i_t'th element in $f(x_t; \boldsymbol{x}_{<t})$.
4. Update $S \leftarrow S \circ (x_t, y_t)$.

Finally, it remains to show that $\mathcal{F}$ covers $\mathcal{P}$. That is, to show that for every $S \in \mathcal{P}$ there exists $f \in \mathcal{F}$ such that $y_t \in f(x_t; x_1, \ldots, x_{t-1})$ for all $(x_t, y_t) \in S$. To this end, imagine a simulation of the List SOA on the sequence S, and denote by $I = I(S)$ the set of all indices where the List SOA made a mistake on S. Notice that $|I| \leq d$, since S is realizable by $\mathcal{P}$. For each $t \in I$, let j_t be an index of a set A_t in the canonical k-cover of $\mathcal{Y} \setminus \text{ListSOA}(x_t; S_{<t})$ such that $y_t \in A_t$, and let i_t be the ordinal of y_t in A_t.[5] Consider the function $f \in \mathcal{F}$ parameterized by $(I, \{(i_t, j_t)\}_{t \in I})$. By the laziness property of the List SOA algorithm, it follows that $f(x_t; \boldsymbol{x}_{<t}) = \text{ListSOA}(x_t; S_{<t})$ on each time step $t \notin I$. By construction, on time steps $t \in I$, the list $f(x_t; \boldsymbol{x}_{<t})$ contains the label y_t. Thus, in total f covers S as required. ■

A.4. Agnostic Case

The proof of Theorem 5 consists of an upper bound statement and a lower bound statement:

- **Upper bound.** finite $\mathtt{L}_k(\mathcal{H}) \to \mathcal{H}$ is agnostically k-list learnable.
- **Lower bound.** infinite $\mathtt{L}_k(\mathcal{H}) \to \mathcal{H}$ is not agnostically k-list learnable.

The proof of the upper bound proof follows a similar approach as in the agnostic-to-realizable reduction by Ben-David et al. (2009); Daniely et al. (2015): the idea is to cover the set of all sequences realizable by $\mathcal{H}$ using a small set of adaptive experts, and then run a vanishing regret algorithm on top of these experts. We break the proof into a few propositions. Proposition 2 shows that each finite class of k-list online functions is agnostically k-list learnable, with regret $O(\sqrt{T \ln n})$, where n is the number of functions in the class. This follows by a standard application of vanishing regret algorithms, here we use *Multiplicative Weights*. In Propositions 3 and 4 we show that a general multi-labeled hypothesis class with finite $(k+1)$-ary Littlestone dimension is k-list learnable (with a matching upper bound), relying on Proposition 2 for finite class of experts, and the list version of SSP lemma 1 from previous section.

5. I.e. there are $i_t - 1$ elements in A_t that are smaller than y_t with respect to the assumed ordering on $\mathcal{Y}$.

Multiplicative Weights Algorithm

Parameter: $\gamma \in (0,1)$.
Input: A finite class $\mathcal{H}$ of experts.
Initialize: $\forall f \in \mathcal{H}$ initialize $W_1(f) = 1$.
For $t = 1, 2, \ldots$

1. Receive unlabeled example $x_t \in \mathcal{X}$.
2. Predict $P_t = \sum_{f \in \mathcal{H}} \frac{W_t(f)}{W_t} \cdot f(x_t; x_1, \ldots, x_{t-1})$ where $W_t := \sum_{f \in \mathcal{H}} W_t(f)$.
3. Receive correct label $y_t \in \mathcal{Y}$.
4. $\forall f \in \mathcal{H}$, update $W_{t+1}(f) = \begin{cases} W_t(f) & y_t \in f(x_t) \\ (1-\gamma) W_t(f) & y_t \notin f(x_t) \end{cases}$

Figure 5: Multiplicative Weights Algorithm

A.4.1. FINITE CLASS OF EXPERTS

Proposition 2 *Let $\mathcal{H}$ be a finite class of k-list online functions with size $|\mathcal{H}| = n$. Also let $S = \{(x_i, y_i)\}_{i=1}^{T}$ be an input sequence. Then there exists a k-list online learner $\mathcal{A}$ such that $\mathtt{R}(\mathcal{A}; S) \leq 2\sqrt{T \ln n}$ respecting to the class $\mathcal{H}$.*

We will construct the algorithm $\mathcal{A}$ using the randomized multiplicative weights method, so the desired algorithm will be a randomized algorithm. A randomized k-list algorithm, is an algorithm which for each input sample returns some probability distribution over all subsets of $\mathcal{Y}$ of size k. For our convenience in describing and analyzing the algorithm, we will use a vector representation of the prediction P_t, where the prediction P_t is a vector of length $L = |\mathcal{Y}|$, and each entry i represents the probability that i is in the predicted set. For a deterministic k-list algorithm, the value of each entry in the vector representation of P_t will be 0 or 1, and the number of 1's will be k. For a randomized k-list algorithm, the value of each entry in the vector representation of P_t will be between 0 and 1, and the summation over all entries will be k. With this point of view we can also look at a randomized algorithm as a deterministic algorithm, where the loss function (i.e. the probability to make a mistake) will be $l_t = 1 - P_t(y_t)$, which is exactly the probability that y_t will not be in the randomized-chosen predicted set P_t.

Proof The proof uses the multiplicative weights algorithm with parameter $\gamma \in (0,1)$, described in Figure 5. This algorithm runs over a finite class of experts, where each expert is a k-list online function. The computation of the prediction P_t uses the vector representation of each expert $f \in \mathcal{H}$, so that P_t is a convex combination of those vectors and hence a legal vector representation of a randomized k-list algorithm's prediction i.e. each entry is between 0 and 1, and the summation over all entries is k. The analysis of the multiplicative weights algorithm does not care of the type of the experts in class. In particular, the analysis does not depend neither on the number of labels L, nor on the list's size k. Let $S = \{(x_t, y_t)\}_{t=1}^{T}$ be any input sequence, let $L_T = \sum_{t=1}^{T} l_t$ be the accumulated loss until time T, and set $\mathtt{opt}_{\mathcal{H}} := \min_{f \in \mathcal{H}} \mathtt{M}(f; S)$. The sum of all weights at time $t+1$ equals to

the following expression:

$$W_{t+1} = l_t W_t(1-\gamma) + (1-l_t)W_t = W_t(1-\gamma l_t)$$

Hence, if we denote the best expert in $\mathcal{H}$ by f^* and note that $W_1 = n$, we get the following relation:

$$(1-\gamma)^{\texttt{opt}_{\mathcal{H}}} = W_T(f^*) \leq W_T = n\prod_{t=1}^{T}(1-\gamma l_t) \leq n\exp\left(-\gamma\sum_{t=1}^{T} l_t\right) = n\exp\left(-\gamma L_T\right)$$

Thus,

$$(1-\gamma)^{\texttt{opt}_{\mathcal{H}}} \leq n\exp\left(-\gamma L_T\right)$$

Claim 3 $1 - x \geq \exp(-x - x^2)\ \forall x \in [0, \frac{1}{2}]$

Proof By the Taylor series, it is known that $\forall x \in [0,1)$, $\frac{1}{1-x} = 1 + x + x^2 + x^3 + \ldots$, hence we get:

$$\begin{aligned} 1 - x &= \exp(\ln(1-x)) = \exp(-\int \frac{1}{1-x}) = \exp(-\int (1 + x + x^2 + \ldots)) \\ &= \exp(-(x + \frac{x^2}{2} + \frac{x^3}{3} + \ldots)) \underset{x \leq \frac{1}{2}}{\geq} \exp(-(x + \frac{x^2}{2} + \frac{x^2}{2})) \end{aligned}$$

■

Using Claim 3 with $x = \gamma$, we get:

$$\begin{aligned} \exp(\texttt{opt}_{\mathcal{H}}(-\gamma - \gamma^2)) \leq (1-\gamma)^{\texttt{opt}_{\mathcal{H}}} &\leq n\exp(-\gamma L_T) \\ \Rightarrow\ -\texttt{opt}_{\mathcal{H}}(\gamma + \gamma^2)) &\leq \ln n - \gamma L_T \\ \Rightarrow \gamma(L_T - \texttt{opt}_{\mathcal{H}}) &\leq \ln n + \texttt{opt}_{\mathcal{H}} \cdot \gamma^2 \\ \Rightarrow \texttt{R}(\mathcal{A}; S) = L_T - \texttt{opt}_{\mathcal{H}} \leq \frac{\ln n}{\gamma} + \texttt{opt}_{\mathcal{H}} \cdot \gamma &\leq \frac{\ln n}{\gamma} + T\gamma \end{aligned}$$

This gives us an upper bound for the regret, depending on γ, and by setting $\gamma = \sqrt{\frac{lnn}{T}}$ we will get the desired bound from the theorem:

$$\texttt{R}(\mathcal{A}; S) = L_T - \texttt{opt}_{\mathcal{H}} \leq 2\sqrt{Tlnn}.$$

Note that the above result is valid only for large values of T when $\gamma = \sqrt{\frac{lnn}{T}} \leq \frac{1}{2}$. But we can overcome this obstacle easily. If $\sqrt{\frac{lnn}{T}} > \frac{1}{2}$ then $T < 4\ln n$. Hence $T^2 < 4T\ln n$ and by taking the square root of both sides, $T < 2\sqrt{T\ln n}$. Since the regret is bounded by the length of the input sequence T, we will get that if $\sqrt{\frac{lnn}{T}} > \frac{1}{2}$ then,

$$\texttt{R}(\mathcal{A}; S) \leq T \leq 2\sqrt{Tlnn}.$$

■

A.4.2. GENERAL CLASS

The previous section showed that any finite class of k-list online experts is agnostic online learnable in the k-list setting with a bound on the regret of $2\sqrt{T \ln n}$, where n is the number of experts and T is the length of the input sequence. We now want to generalize this result to a multi-labeled hypothesis class $\mathcal{H}$ with no restriction on the size of the class. For that we will use the list SSP lemma to cover the class $\mathcal{H}$ by a finite set of k-list online functions, and then applying the multiplicative weights algorithm over the finite covering set.

Proposition 3 *Let $\mathcal{H}$ be a (multi-labeled) hypothesis class, with $\mathtt{L}_k(\mathcal{H}) = d < \infty$, and let $S = \{(x_t, y_t)\}_{t=1}^T$ be an input sequence. Then there exists an algorithm $\mathcal{A}$ such that*

$$\mathtt{R}(\mathcal{A}; S) < 2\sqrt{dT + dT \ln\Big(\frac{T}{d}\Big\lceil\frac{L-k}{k}\Big\rceil k\Big)}$$

with respect to the class $\mathcal{H}$.

Proof Let $\mathcal{F}$ be a family of $n \leq \binom{T}{\leq d}\Big(\lceil\frac{L-k}{k}\rceil k\Big)^d$ k-list online functions which is a T-cover of $\mathcal{P}(\mathcal{H})$. Namely, that covers all realizable sequences by $\mathcal{H}$ of length T. Apply the multiplicative weights algorithm $\mathcal{A}$ described in the proof of Proposition 2 on the class $\mathcal{F}$, and get that

$$\begin{aligned}
\mathtt{R}(\mathcal{A}; S) &\leq 2\sqrt{T \ln n} \\
&= 2\sqrt{T \ln\Big(\binom{T}{\leq d}\Big(\Big\lceil\frac{L-k}{k}\Big\rceil k\Big)^d\Big)} \\
&\leq 2\sqrt{T \ln\Big(\Big(\frac{eT}{d}\Big)^d\Big(\Big\lceil\frac{L-k}{k}\Big\rceil k\Big)^d\Big)} \\
&= 2\sqrt{dT + dT \ln\Big(\frac{T}{d}\Big\lceil\frac{L-k}{k}\Big\rceil k\Big)} =: r(T)
\end{aligned}$$

where the regret is with respect to the class $\mathcal{F}$. This means that

$$\mathtt{M}(\mathcal{A}; S) - \min_{f\in\mathcal{F}} \mathtt{M}(f; S) \leq r(T).$$

Since $\min_{f\in\mathcal{F}} \mathtt{M}(f; S) \leq \min_{h\in\mathcal{H}} \mathtt{M}(h; S)$ we get that

$$\mathtt{M}(\mathcal{A}; S) - \min_{h\in\mathcal{H}} \mathtt{M}(h; S) \leq \mathtt{M}(\mathcal{A}; S) - \min_{f\in\mathcal{F}} \mathtt{M}(f; S) \leq r(T).$$

Hence,

$$\mathtt{R}(\mathcal{A}; S) \leq r(T)$$

with respect to the class $\mathcal{H}$. ■

Notice that the randomized algorithm above achieves a sub-linear bound on the regret while assuming that the length of the input sequence is known. We now want to remove this assumption, and construct an algorithm that also achieves the same bound on the regret. We will rely on the

construction of the above algorithm in order to build the desired algorithm that learns $\mathcal{H}$ without assuming that the length of the input sequence is given. The method we use is called the doubling trick (Shalev-Shwartz, 2012). The idea is to separate the input sequence of examples into subsequences with increasing size and then apply the algorithm for a known size of input sequence repeatedly over these subsequences.

Proposition 4 *Let $\mathcal{H}$ be a (multi-labeled) hypothesis class with $\mathtt{L}_k(\mathcal{H}) = d < \infty$, and also let S be an input sequence. Then there exists an algorithm $\mathcal{A}$ such that*

$$\mathtt{R}(\mathcal{A}; S) < \frac{2\sqrt{2}}{\sqrt{2}-1}\sqrt{dT + dT\ln\left(\frac{T}{d}\left\lceil\frac{L-k}{k}\right\rceil k\right)}$$

with respect to the class $\mathcal{H}$, where $T = |S|$.

Proof We will use the algorithm $\mathcal{A}$ described on the proof of Proposition 3, which knows the length of the input sequence, and apply it on subsequences of the original input sequence of length T as follows: The first subsequence S_1 will be the first sample in S: (x_1, y_1), the second subsequence S_2 will be the next two samples: $(x_2, y_2), (x_3, y_3)$, the third subsequence S_3 will be the next four samples: $(x_4, y_4), (x_5, y_5), (x_6, y_6), (x_7, y_7)$ and so on. We will get a sequence of subsequences $\{S_i\}_{i\geq 1}$ with increasing lengths: $\{T_i\}_{i\geq 1} = 1, 2, 4, 8, \ldots$ such that the length of the i'th sequence S_i will be $|S_i| = T_i = 2^{i-1}$. [6] On each of these subsequences with a known length, we can now apply the algorithm $\mathcal{A}$, to get the following bound on the regret:

$$2\sqrt{dT_i + dT_i\ln\left(\frac{T_i}{d}\left\lceil\frac{L-k}{k}\right\rceil k\right)}.$$

For each step i, we got a bound on the regret for a specific pattern $h_i \in \mathcal{H}$, so

$$\min_{h\in\mathcal{H}} \mathtt{M}(h; S) \geq \sum_{i=1}^{\lfloor \log T\rfloor+1} \min_{h\in\mathcal{H}} \mathtt{M}(h; S_i),$$

since the optimal "global" hypothesis h may make more mistakes on S_i then the optimal "local" hypothesis h_i. Hence,

$$\mathtt{R}(\mathcal{A}; S) \leq \sum_{i=1}^{\lfloor \log T\rfloor+1} \mathtt{R}(\mathcal{A}_i; S_i),$$

6. The length of the last sequence may be smaller, but it can only improve the bound.

where $\mathcal{A}_i$ is the algorithm applied on the subsequence S_i, with the assumption of length T_i, and with some calculations we will get

$$\begin{aligned}
\mathtt{R}(\mathcal{A}; S) &\leq \sum_{i=1}^{\lfloor \log T \rfloor + 1} \mathtt{R}(\mathcal{A}_i; S_i) \\
&= \sum_{i=1}^{\lfloor \log T \rfloor + 1} 2\sqrt{dT_i + dT_i \ln\left(\frac{T_i}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)} \\
&= \sum_{i=1}^{\lfloor \log T \rfloor + 1} 2\sqrt{d2^{i-1} + d2^{i-1} \ln\left(\frac{2^{i-1}}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)} \\
&\leq \sum_{i=1}^{\lfloor \log T \rfloor + 1} 2\sqrt{d2^{i-1} + d2^{i-1} \ln\left(\frac{2^{\log T}}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)} \\
&= 2\sqrt{d + d\ln\left(\frac{T}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)} \sum_{i=1}^{\lfloor \log T \rfloor + 1} 2^{i-1} \\
&= 2\sqrt{d + d\ln\left(\frac{T}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)} \frac{\sqrt{2}^{\lfloor \log T \rfloor + 1} - 1}{\sqrt{2} - 1} \\
&\leq 2\sqrt{d + d\ln\left(\frac{T}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)} \frac{\sqrt{2}^{\lfloor \log T \rfloor + 1}}{\sqrt{2} - 1} \\
&\leq 2\sqrt{d + d\ln\left(\frac{T}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)} \frac{\sqrt{2T}}{\sqrt{2} - 1} \\
&= \frac{2\sqrt{2}}{\sqrt{2} - 1}\sqrt{dT + dT\ln\left(\frac{T}{d}\left\lceil \frac{L-k}{k} \right\rceil k\right)}.
\end{aligned}$$

■

The following Proposition shows that a class with infinite $(k+1)$-ary Littlestone dimension is not k-list learnable in the agnostic setting.

Proposition 5 *Let $\mathcal{H}$ be a (multi-labeled) hypothesis class with infinite $(k+1)$-ary Littlestone dimension $\mathtt{L}_k(\mathcal{H}) = \infty$. Then for each (maybe randomized) k-list algorithm $\mathcal{A}$ and $T \in \mathbb{N}$ there exists an input sequence S of length T such that the regret is*

$$\mathtt{R}(\mathcal{A}; S) = \Omega(T).$$

Proof The proof repeats the same argument as in the lower bound proof of $\mathtt{L}_k(\mathcal{P})$ for randomized learners over the number of mistakes; let $\mathcal{A}$ be a k-list online learner and let $T \in \mathbb{N}$. Since $\mathtt{L}_k(\mathcal{H}) = \infty$, there is a complete $(k+1)$-ary mistake tree of depth T, which is shattered by $\mathcal{H}$. Follow a root-to-leaf path in the shattered tree, and choose one of the $k+1$ labels associated to the $k+1$ outgoing edges from the current node in the path, uniformly at random. At each step, the learner $\mathcal{A}$

will make a mistake with probability $\frac{1}{k+1}$. The sequence S of length T generated by this process, is realizable by $\mathcal{H}$, which means $\min_{h\in\mathcal{H}} \mathtt{M}(h;S) = 0$. Further, $\mathcal{A}$ makes $\frac{1}{k+1}T$ mistakes on S in expectation. Thus, the regret will be $\frac{1}{k+1}T$ which means $\mathtt{R}(\mathcal{A};S) = \Omega(T)$. ■

Proof [proof of Theorem 5] The proof of the upper bound was given in a series of propositions: Proposition 2, Proposition 3 and Proposition 4, while the proof of the lower bound was given by Proposition 5.

■

A.5. Proof of Theorem 6

Proof Let $\mathcal{H}$ be a multi-labeled hypothesis class, $|\mathcal{H}| > 1$, and let $1 \leq \mathtt{k}_{\mathcal{H}} \leq |\mathcal{Y}|$ be the minimal number such that $\mathtt{L}_{\mathtt{k}_{\mathcal{H}}}(\mathcal{H}) < \infty$ [7]. By Theorem 5, an agnostic list-learning rule $\mathcal{A}$ for $\mathcal{H}$ exists if and only if $\mathtt{k}_{\mathcal{H}} \leq \mathtt{k}_{\mathcal{A}}$, where $\mathtt{k}_{\mathcal{A}}$ is the size of the lists used by $\mathcal{A}$.

Proof [Proof of Item 1 (Negative Regret)]

Let $\mathtt{k}_{\mathcal{H}} < k \leq |\mathcal{Y}|$. We will show that there exists a learning rule $\mathcal{A}$ with $\mathtt{k}_{\mathcal{A}} = k$ which learns $\mathcal{H}$ with a negative regret in the following sense: its expected regret on every input sequence of length T is at most

$$(p-1)\cdot \mathtt{opt}_{\mathcal{H}} + c\cdot\left(\sqrt{T\ln T}\right),$$

where $p = p(\mathcal{H}) < 1$, $c = c(\mathcal{H})$ both do not depend on T, and $\mathtt{opt}_{\mathcal{H}}$ is the number of mistakes made by the best function in $\mathcal{H}$. The idea is to use a list-learning rule, denoted by $\mathcal{B}$, that uses lists of size $\mathtt{k}_{\mathcal{H}}$ and has vanishing regret ($O(\sqrt{d\cdot T\ln T})$) where $d = \mathtt{L}_{\mathtt{k}_{\mathcal{H}}}(\mathcal{H})$ (such a learning rule exists by Theorem 5), and in each time-step add to its list $\mathtt{k}_{\mathcal{A}} - \mathtt{k}_{\mathcal{H}}$ labels that are chosen uniformly at random from the remaining labels. To analyze the regret, notice that on every fixed input sequence $S = \{(x_t, y_t)\}_{t=1}^{T}$ of length T, the accumulated loss $\mathcal{A}$ suffers is equal to the accumulated loss $\mathcal{B}$ suffers times $p < 1$, where $p = \frac{|\mathcal{Y}|-k}{|\mathcal{Y}|-\mathtt{k}_{\mathcal{H}}}$. Indeed, in every time-step t, let $\ell_t^{\mathcal{B}}$ denote the expected loss of $\mathcal{B}$ in this round, that is, $\ell_t^{\mathcal{B}}$ is the probability that the random list of size $\mathtt{k}_{\mathcal{H}}$ does not include y_t, and also let $\mathcal{Y}_{add}$ denote the (random) set of labels that are added to the list $\mathcal{B}$ predicts (so $|\mathcal{Y}_{add}| = k - \mathtt{k}_{\mathcal{H}}$). Then,

$$\begin{aligned}\ell_t^{\mathcal{A}} &= \ell_t^{\mathcal{B}}\cdot \Pr\left[y_t \notin \mathcal{Y}_{add} | y_t \notin \mathcal{B}\left(x_t; \{(x_i,y_i)\}_{i<t}\right)\right] \\ &= \ell_t^{\mathcal{B}}\left(1 - \frac{k-\mathtt{k}_{\mathcal{H}}}{|\mathcal{Y}|-\mathtt{k}_{\mathcal{H}}}\right) = \ell_t^{\mathcal{B}}\left(\frac{|\mathcal{Y}|-k}{|\mathcal{Y}|-\mathtt{k}_{\mathcal{H}}}\right) = \ell_t^{\mathcal{B}}\cdot p.\end{aligned}$$

7. Notice that the minimum always exists if $\mathcal{Y}$ is finite since $\mathtt{L}_{|\mathcal{Y}|}(\mathcal{H}) < \infty$.

And by summing over the whole input sequence S, we get

$$\begin{aligned}
\mathtt{R}(\mathcal{A};S) &= \mathtt{M}(\mathcal{A};S) - \min_{h\in\mathcal{H}} \mathtt{M}(h;S) \\
&= \sum_{t=1}^{T} \ell_t^{\mathcal{A}} - \mathtt{opt}_{\mathcal{H}} \\
&= \sum_{t=1}^{T} \ell_t^{\mathcal{B}} \cdot p - \mathtt{opt}_{\mathcal{H}} \\
&= p \sum_{t=1}^{T} \ell_t^{\mathcal{B}} - \mathtt{opt}_{\mathcal{H}} \\
&\leq p(\mathtt{opt}_{\mathcal{H}} + c \cdot \left(\sqrt{T \ln T}\right)) - \mathtt{opt}_{\mathcal{H}} \\
&= (p-1)\mathtt{opt}_{\mathcal{H}} + p \cdot c\left(\sqrt{T \ln T}\right) \\
&\leq (p-1)\mathtt{opt}_{\mathcal{H}} + c\left(\sqrt{T \ln T}\right).
\end{aligned}$$

■

Proof [Proof of Item 2 (Nonnegative Regret)] Assume that $\mathcal{A}$ is a list-learning rule with $\mathtt{k}_{\mathcal{A}} = \mathtt{k}_{\mathcal{H}}$. We will show that $\mathcal{A}$ has a non-negative regret, by showing that for a sufficient small $p > 0$, there is an input sequence S with arbitrarily large length T, such that $\mathtt{opt}_{\mathcal{H}} \leq \lfloor p \cdot T \rfloor$ but the expected number of mistakes made by $\mathcal{A}$ is at least $\lfloor p \cdot T \rfloor$. We will show that this holds for any $p \leq \frac{1}{\mathtt{k}_{\mathcal{H}}+1}$. To simplify presentation we assume here that $p \cdot T \in \mathbb{N}$.

Let $T \in \mathbb{N}$. Since $\mathtt{L}_{\mathtt{k}_{\mathcal{H}}-1}(\mathcal{H}) = \infty$, there is a complete $\mathtt{k}_{\mathcal{H}}$-ary mistake tree $\mathcal{T}$ of length T which is shattered by $\mathcal{H}$. We construct the hard sequence S randomly as follows:

1. First draw a branch B in $\mathcal{T}$ uniformly at random by starting at the root and at each step proceed on an outgoing edge which is chosen uniformly at random.

2. Let S_B denote the sequence of T examples corresponding to the random branch, and let L_t be the set of labels corresponding to the outgoing edges from the t'th node in the branch B. (In particular, notice that $|L_t| = \mathtt{k}_{\mathcal{H}}$ and $y_t \in L_t$.)

3. Pick uniformly a random subsequence of $p \cdot T$ examples from S_B, and for each of the examples (x_t, y_t) in the subsequence, replace the label y_t with a random label y'_t, chosen uniformly such that $y'_t \notin L_t$.

4. Denote by S the resulting random sequence.

Notice that $\mathbb{E}[\mathtt{opt}_{\mathcal{H}}] \leq p \cdot T$; indeed, S_B is realizable and S differs from it in $p \cdot T$ places.

It remains to show that the expected number of mistakes made by $\mathcal{A}$ is $\geq p \cdot T$. By linearity of expectation, it suffices to show that the probability that $\mathcal{A}$ makes a mistake at each step is $\geq p$. For each $y \in \mathcal{Y}$ and $t \leq T$, let $p_t(y)$ be the probability that the label y is in the predicted list of $\mathcal{A}$ at time-step t and let $q_t(y) = 1 - p_t(y)$ be the probability that y is not in the predicted list of $\mathcal{A}$ in time t. Let $\alpha = \sum_{y \in L_t} p_t(y)$ and $\beta = \sum_{y \in \mathcal{Y} \setminus L_t} p_t(y)$; thus, $\alpha + \beta = |L_t| = \mathtt{k}_{\mathcal{H}}$. Then, the probability

that $\mathcal{A}$ makes a mistake at the time-step t is:

$$\begin{aligned} l_t(\mathcal{A}) &= (1-p)\cdot\sum_{y\in L_t} q_t(y)\cdot\frac{1}{|L_t|} + p\cdot\sum_{y\notin L_t} q_t(y)\cdot\frac{1}{|\mathcal{Y}\setminus L_t|} \\ &= (1-p)\cdot\frac{\mathtt{k}_{\mathcal{H}}-\alpha}{\mathtt{k}_{\mathcal{H}}} + p\cdot\frac{|\mathcal{Y}|-\mathtt{k}_{\mathcal{H}}-\beta}{|\mathcal{Y}|-\mathtt{k}_{\mathcal{H}}} \\ &= (1-p)\cdot\frac{\beta}{\mathtt{k}_{\mathcal{H}}} + p\cdot\frac{|\mathcal{Y}|-\mathtt{k}_{\mathcal{H}}-\beta}{|\mathcal{Y}|-\mathtt{k}_{\mathcal{H}}}. \end{aligned}$$

It suffices to show that $l_t(\mathcal{A}) \geq p$ for every possible value of $\beta \in [0, \mathtt{k}_{\mathcal{H}}]$. Indeed:

- If $\beta = 0$, then $l_t(\mathcal{A}) = p$.
- If $\beta \geq 1$, then $l_t(\mathcal{A}) \geq (1-p)\frac{\beta}{\mathtt{k}_{\mathcal{H}}} \geq (1-p)\frac{1}{k_{\mathcal{H}}} \geq p$, because $p \leq \frac{1}{\mathtt{k}_{\mathcal{H}}+1}$.
- If $0 < \beta < 1$, then since $l_t(\mathcal{A})$ is a linear function in β, getting values $\geq p$ for $\beta \in \{0,1\}$ then also for the middle range, $l_t(\mathcal{A}) \geq p$.

To conclude this analysis, for each value of p smaller than $1/(\mathtt{k}_{\mathcal{H}}+1)$, each list-learning rule $\mathcal{A}$, with list size $\mathtt{k}_{\mathcal{H}}$, achieves a non-negative regret for some input sequence with arbitrarily large length T.

■

Proof [Proof of Item 3 (Unbounded Regret)] Let $\mathcal{A}$ be a list learning rule with $\mathtt{k}_{\mathcal{A}} = \mathtt{k}_{\mathcal{H}}$, and let $h', h'' \in \mathcal{H}$ and $x \in \mathcal{X}$ such that $h'(x), h''(x)$ are distinct lists of size $\mathtt{k}_{\mathcal{H}}$. Produce a random sequence of length T whose examples are of the form (x, y_t), where y_t is a random label drawn uniformly from $h'(x) \cup h''(x)$. Notice that the expected number of mistakes of $\mathcal{A}$ is at least $T \cdot \left(1 - \frac{\mathtt{k}_{\mathcal{H}}}{U}\right)$, where $U = |h'(x) \cup h''(x)| > \mathtt{k}_{\mathcal{H}}$. Thus it remains to show that $\mathtt{opt}_{\mathcal{H}}$ does better. Consider the random variable Δ counting how many of the y_t's belong to the symmetric difference $h'(x)\Delta h''(x)$. If y_t is in the intersection of $h'(x)$ and $h''(x)$ then both of the functions do not make a mistake on (x, y_t), and therefore such examples do not contribute to $\mathtt{opt}_{\mathcal{H}}$. Otherwise, if y_t is in the symmetric difference $h'(x)\Delta h''(x)$ then the probability that y_t belongs to $h'(x)$ is equal to the probability it belongs to $h''(x)$, and both are equal to $1/2$. Thus, conditioned on Δ, the number of mistakes made by h' or h'' behaves like a binomial random variables X', X'' with parameters $n = \Delta, p = 1/2$, where $X' + X'' = \Delta$ and $\mathbb{E}[\mathtt{opt}_{\mathcal{H}}|\Delta]$ is at most $\mathbb{E}[\min\{X', X''\}|\Delta] = \mathbb{E}[\min\{X', \Delta - X'\}|\Delta]$:

$$\mathbb{E}[\mathtt{opt}_{\mathcal{H}}|\Delta] \leq \mathbb{E}[\min\{X', \Delta - X'\}|\Delta] = \frac{\Delta}{2} - \mathbb{E}\left[\left|X' - \frac{\Delta}{2}\right||\Delta\right].$$

Now, notice that

$$\mathbb{E}\left[\left|X' - \frac{\Delta}{2}\right||\Delta\right] = \sqrt{\mathsf{Var}(X')} = \frac{\sqrt{\Delta}}{2},$$

and by linearity of expectation,

$$\mathbb{E}[\mathtt{opt}_{\mathcal{H}}] = \mathbb{E}[\mathbb{E}[\mathtt{opt}_{\mathcal{H}}|\Delta]] \leq \frac{1}{2}\mathbb{E}[\Delta] - \frac{1}{2}\mathbb{E}[\sqrt{\Delta}] \leq \mathbb{E}[L_T(\mathcal{A})] - \frac{1}{2}\mathbb{E}[\sqrt{\Delta}],$$

where the last inequality holds because

$$\mathbb{E}[\Delta] = T\cdot\frac{U - |h'(x)\cap h''(x)|}{U} = T\cdot\frac{U-(2\mathtt{k}_{\mathcal{H}}-U)}{U} = 2T\cdot\frac{U-\mathtt{k}_{\mathcal{H}}}{U} \leq 2\,\mathbb{E}[L_T(\mathcal{A})].$$

It thus remains to show that $\mathbb{E}[\sqrt{\Delta}] = \Omega(\sqrt{T})$. Indeed, since Δ is a binomial random variable with horizon T and probability $p = 2 \cdot \frac{U - \mathtt{k}_{\mathcal{H}}}{U}$,

$$\mathbb{E}[\sqrt{\Delta}] \geq \sqrt{T \cdot p} - \frac{1-p}{2\sqrt{T \cdot p}} = \Omega(\sqrt{T}).$$

The latter follows because $\sqrt{x} \geq 1 + \frac{x-1}{2} - \frac{(x-1)^2}{2}$ and therefore for any nonnegative random variable X, $\mathbb{E}[\sqrt{X}] \geq \sqrt{\mathbb{E}[X]}\left(1 - \frac{\mathsf{Var}(X)}{2\,\mathbb{E}[X]^2}\right)$. Further, this bound can be expressed in terms of $\mathtt{k}_{\mathcal{H}}$, by the observation that

$$p = 2 \cdot \frac{U - \mathtt{k}_{\mathcal{H}}}{U} \geq \frac{2}{\mathtt{k}_{\mathcal{H}} + 1}, \qquad (\mathtt{k}_{\mathcal{H}} + 1 \leq U \leq 2 \cdot \mathtt{k}_{\mathcal{H}})$$

and thus,

$$\mathbb{E}[\sqrt{\Delta}] \geq \sqrt{T \cdot p} - \frac{1-p}{2\sqrt{T \cdot p}} \geq \sqrt{T \cdot \frac{2}{\mathtt{k}_{\mathcal{H}} + 1}} - \frac{1 - \frac{2}{\mathtt{k}_{\mathcal{H}}+1}}{2\sqrt{T \cdot \frac{2}{\mathtt{k}_{\mathcal{H}}+1}}} = \Omega(\sqrt{T}).$$

■

■

Appendix B. Pattern Classes vs. Hypothesis Classes

In this section we compare between hypothesis classes and pattern classes in terms of their expressivity to model learning tasks.

Recall that a pattern class $\mathcal{P}$ is a collection of finite sequences of examples, which is downward closed: for every $S \in \mathcal{P}$, if $S' \subseteq S$ is a subsequence of S then $S' \in \mathcal{P}$, and also that the induced pattern class $\mathcal{P}(\mathcal{H})$ of an hypothesis class $\mathcal{H}$ is:

$$\mathcal{P}(\mathcal{H}) = \{S \in \mathcal{Z}^\star : S \text{ is consistent with some } h \in \mathcal{H}\}.$$

There are two basic properties which seem to capture the gap between pattern classes and hypothesis classes. These two properties hold in any induced pattern class $\mathcal{P}(\mathcal{H})$, but (as shown below) they do not necessarily hold in a general pattern class $\mathcal{P}$.

Contradiction-free. A pattern class $\mathcal{P}$ is called contradiction-free if every pattern in it has no contradictions, namely it does not contain the same example twice with different labels. Clearly, $\mathcal{P}(\mathcal{H})$ is contradiction-free for every hypothesis class $\mathcal{H}$. Thus, any pattern class $\mathcal{P}$ which contains a contradiction cannot be extended by an hypothesis class $\mathcal{H}$. Below we focus only on contradiction-free pattern classes.

Symmetry. A pattern class $\mathcal{P}$ is called symmetric if it is closed under taking permutations: if $S \in \mathcal{P}$ then $\pi(S) \in \mathcal{P}$ for every permutation π of the elements in S. Clearly, $\mathcal{P}(\mathcal{H})$ is symmetric for every hypothesis class $\mathcal{H}$. Non-symmetric pattern classes can be naturally used to express adaptive/dynamic hypotheses (or hypotheses with memory), which are functions of the form $h : \mathcal{X}^\star \to \mathcal{Y}$ mapping a finite sequence $x_1, \ldots, x_t$ to an output $y_t \in \mathcal{Y}$. (x_t is thought of as the current point, y_t is its label, and $x_1, \ldots, x_{t-1}$ are the previous points.) Thus, non-symmetric pattern classes allows us to study the learnability of adaptive hypothesis classes.

This suggests the following question: *Can learnable pattern classes which are contradiction-free (but possibly non-symmetric) be extended by a learnable hypothesis class?* I.e. is it the case that for every contradiction-free pattern class $\mathcal{P}$ with $\mathtt{L}(\mathcal{P}) < \infty$ there exists an hypothesis class $\mathcal{H}$ such that $\mathcal{P}(\mathcal{H}) \supseteq \mathcal{P}$ and $\mathtt{L}(\mathcal{H}) < \infty$?

The answer to this question is no, as the following example shows. Let $\mathcal{X}$ be an infinite set and $\mathcal{Y} = \{0, 1\}$. Consider the pattern class $\mathcal{P} \subset (\mathcal{X} \times \mathcal{Y})^*$ containing all patterns $\{(x_i, y_i)\}_{i=1}^n$ such that their restriction to $\mathcal{Y}$, $\{y_i\}_{i=1}^n$, is non-increasing: if $i < j$ then $y_i \geq y_j$, and such that $y_i \neq y_j \Rightarrow x_i \neq x_j$ (i.e. contradiction-free property).

Not hard to see that $\mathcal{P}$ is online learnable by the learner that predicts the label 1, until she makes a mistake (namely, the true label is 0), and then predicts the label 0 for the rest of the input sequence of examples.

Now assume that $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ is an hypothesis class, such that $\mathcal{P} \subseteq \mathcal{P}(\mathcal{H})$. We will show that $\mathcal{H}$ cannot be online learnable, by showing that $\mathtt{L}(\mathcal{H}) \geq d$ for all $d \in \mathbb{N}$. Let $d \in \mathbb{N}$, we will show that there is a complete binary mistake tree of depth d which is shattered by $\mathcal{H}$. In fact, we will show a stronger result; we will show that almost *any* complete binary mistake tree of depth d is shattered by $\mathcal{H}$, with the only restriction of different values of $x \in \mathcal{X}$ in each root-to-leaf path in the tree, to avoid contradictions. Let $\mathcal{T}$ be such a tree, and let S be the induced sequence of some path in $\mathcal{T}$. We claim that S is realized by $\mathcal{H}$. Indeed, reorder the elements of S to get a sequence S' with non-increasing order of the labels. I.e., permute S to get a sequence $S' \in \mathcal{P} \subseteq \mathcal{P}(\mathcal{H})$. Since $\mathcal{P}(\mathcal{H})$ is symmetric it follows that $S \in \mathcal{P}(\mathcal{H})$ and hence S is realized by $\mathcal{H}$. Thus, $\mathcal{T}$ is shattered by $\mathcal{H}$ as claimed.

What about learnable pattern classes $\mathcal{P}$ which are both free of contradictions and symmetric? Is there always a learnable hypothesis class $\mathcal{H}$ such that $\mathcal{P} \subset \mathcal{P}(\mathcal{H})$? This remains an interesting open question for future research. (We remark that an equivalent open question was asked by Alon et al. (2021).)

Appendix C. Tightness of Bounds for Randomized Learners

Here we give two examples for pattern classes, one class $\mathcal{P}_1$ with $\mathtt{R}_k(\mathcal{P}_1) = \frac{1}{k+1}\mathtt{L}_k(\mathcal{P}_1)$ and second class $\mathcal{P}_2$ with $\mathtt{R}_k(\mathcal{P}_2) = \mathtt{L}_k(\mathcal{P}_2)$.

Lower bound is tight. The first class is the induced pattern class $\mathcal{P}_1 = \mathcal{P}(\mathcal{H})$ of the hypothesis class of all functions from $[d]$ to $[k+1]$:

$$\mathcal{H} = [k+1]^{[d]}.$$

Notice that $\mathtt{L}_k(\mathcal{P}_1) = d$.

Now, we will define a randomized algorithm $\mathcal{A}$ which makes at most $\frac{1}{k+1}\mathtt{L}_k(\mathcal{P}_1) = \frac{d}{k+1}$ mistakes in expectation on every realizable sequence. Notice that in a realizable sequence $S =$

$\{(x_i, y_i)\}_{i=1}^T$ by $\mathcal{P}_1$, if $x_i = x_j$ then $y_i = y_j$. Define the algorithm $\mathcal{A}$ so that for each input example x_i, the prediction will be a list of k labels, chosen uniformly at random. If the algorithm already seen the example x_i, then the predicted set will consists of the matching known label and completed to a list of size k arbitrarily. for each $x \in [d]$, the algorithm $\mathcal{A}$ may make a mistake on x in at most one single time step. The probability that the learner will make a mistake on x at this time step is $\frac{1}{k+1}$. Hence, the expected number of mistakes of $\mathcal{A}$ on S is at most $\frac{d}{k+1}$.

Upper bound is tight. The second class is the induced pattern class $\mathcal{P}_2 = \mathcal{P}(\mathcal{H})$ of the following hypothesis class $\mathcal{H} \subset \{0, 1, \ldots, k\}^{\mathbb{N}}$:

$$\mathcal{H} = \{h \in \{0, 1, \ldots, k\}^{\mathbb{N}} : |h^{-1}(0)| \leq d\}.$$

Notice that $\mathcal{P}_2$ is online learnable by the learner that always predicts the list $\{1, 2, \ldots, k\}$, and that $\mathtt{L}_k(\mathcal{P}_2) = d$. Indeed, take a complete $(k+1)$-ary mistake tree of depth d, where all nodes with same depth are associated with the same $x \in \mathbb{N}$ and each two nodes with different depths are associated with different items from $\mathbb{N}$. This tree is shattered by $\mathcal{P}_2$, hence $\mathtt{L}_k(\mathcal{P}_2) \geq d$. In addition, there is no complete shattered tree with depth larger than d, since a path with all zeros labels must have size of at most d, hence $\mathtt{L}_k(\mathcal{P}_2) \leq d$.

Now let $\mathcal{A}$ be a randomized learner. We will show that for each $\epsilon > 0$, there is a realizable sequence $S = \{(x_t, y_t)\}_{t=1}^T$ of length $T(\epsilon)$, such that $\mathcal{A}$ makes at least $d - \epsilon$ mistakes on S in expectation. First, we choose the x_t's such that they are all distinct, e.g. $x_t = t$. The labels y_t are defined as follows: Let $\epsilon > 0$. Let P_t be the (maybe random) predicted set of $\mathcal{A}$ at time t. If $\Pr(0 \in P_t) \leq \frac{\epsilon}{d}$, then we will set $y_t = 0$. If we already set $y_t = 0$ d times, then we will choose a different value for y_t, uniformly at random from $\{1, \ldots, k\}$, to keep the realizability property of the input sequence. If $\Pr(0 \in P_t) > \frac{\epsilon}{d}$, then we will choose the label y_t uniformly at random from the set $\{1, \ldots, k\}$. By this construction of the input sequence, we get the following result; in the case that the input sequence contains d entries with $y_t = 0$, it means that $\mathcal{A}$ makes at least $d\left(1 - \frac{\epsilon}{d}\right) = d - \epsilon$ mistakes in expectation. In the case that input sequence contains less than d entries with $y_t = 0$, then for input sequence with length $T(\epsilon) \geq \frac{kd^2}{\epsilon} + d - 1$, $\mathcal{A}$ will make at least $\frac{kd^2}{\epsilon}\left(1 - \frac{1}{k}\left(k - \frac{\epsilon}{d}\right)\right) = d$ mistakes in expectation.