A CHARACTERIZATION OF MULTICLASS LEARNABILITY

NATALY BRUKHIM, DANIEL CARMON, IRIT DINUR,
SHAY MORAN, AND AMIR YEHUDAYOFF

ABSTRACT. A seminal result in learning theory characterizes the PAC learnability of binary classes through the Vapnik-Chervonenkis dimension. Extending this characterization to the general multiclass setting has been open since the pioneering works on multiclass PAC learning in the late 1980s. This work resolves this problem: we characterize multiclass PAC learnability through the DS dimension, a combinatorial dimension defined by Daniely and Shalev-Shwartz (2014).

The classical characterization of the binary case boils down to empirical risk minimization. In contrast, our characterization of the multiclass case involves a variety of algorithmic ideas; these include a natural setting we call *list PAC learning*. In the list learning setting, instead of predicting a single outcome for a given unseen input, the goal is to provide a short menu of predictions.

Our second main result concerns the Natarajan dimension, which has been a central candidate for characterizing multiclass learnability. This dimension was introduced by Natarajan (1988) as a barrier for PAC learning. He furthered showed that it is the only barrier, provided that the number of labels is bounded. Whether the Natarajan dimension characterizes PAC learnability in general has been posed as an open question in several papers since. This work provides a negative answer: we construct a non-learnable class with Natarajan dimension 1.

For the construction, we identify a fundamental connection between concept classes and topology (i.e., colorful simplicial complexes). We crucially rely on a deep and involved construction of *hyperbolic pseudo-manifolds* by Januszkiewicz and Świątkowski. It is interesting that hyperbolicity is directly related to learning problems that are difficult to solve although no obvious barriers exist. This is another demonstration of the fruitful links machine learning has with different areas in mathematics.

1. Introduction

Many important machine learning tasks require classification into many target classes: in image object recognition, the number of classes is the number of possible objects. In language models, the number of classes scales with the dictionary size. In protein folding prediction, the goal is to predict the 3D structures of proteins based on their 1D amino sequence. These are real-world tasks that do not admit an a priori reasonable bound on the number of classes. Multiclass classification problems, therefore, have been attracting interest both on the theoretical side and on the practical side; for further reading we refer to the introduction of [Daniely and Shalev-Shwartz, 2014] and references within.

The theoretical understanding of multiclass learnability, however, is still lacking: even in the basic Probably Approximately Correct (PAC) setting [Valiant, 1984], learnability is well-understood only when the number of classes is bounded (see e.g. [Natarajan, 1989, Ben-David, Cesabianchi, Haussler, and Long, 1995, Shalev-Shwartz and Ben-David, 2014, Daniely, Sabato, Ben-David, and Shalev-Shwartz, 2015a]).

The fundamental theorem of PAC learning asserts the equivalence between binary classification and finiteness of the Vapnik-Chervonenkis (VC) dimension [Vapnik and Chervonenkis, 1968, 1974, Blumer, Ehrenfeucht, Haussler, and Warmuth, 1989]. The works of Natarajan and Tadepalli [1988] and Natarajan [1989] extended Valiant's PAC framework to the multiclass setting. They identified two natural extensions of the VC dimension: the Natarajan dimension and the Graph dimension. The Natarajan dimension serves as a lower bound on the sample complexity of PAC learning, and the Graph dimension serves as an upper bound [Natarajan and Tadepalli, 1988, Natarajan, 1989]. When the number of classes is bounded ($|\mathcal{Y}| < \infty$), both dimensions characterize PAC learnability. In the unbounded case, however, Natarajan [1988] showed that finite Graph dimension does not characterize PAC learnability; he identified a PAC learnable class with infinite Graph dimension (see also Example 8 below). Natarajan [1989] asked whether the Natarajan dimension characterizes learnability, and explained why standard uniform convergence techniques are not sufficient to resolve this question.

In the 1990s, Ben-David, Cesabianchi, Haussler, and Long [1995] and Haussler and Long [1995] introduced a rich combinatorial framework for defining dimensions in the multiclass setting. This framework captures as special cases many other dimensions, including the Natarajan and Graph dimensions, the Pseudo-dimension [Pollard, 1990, Haussler, 1992], and Vapnik's dimension [Vapnik, 1989]. Within this framework, Ben-David, Cesabianchi, Haussler, and Long [1995] exactly identified those dimensions (called *distinguishers*) that characterize PAC learnability when the number of classes is bounded. This framework, however, does not capture learnability when the number of classes is unbounded, and they left this as an open problem.

More recently, a sequence of works studied general principles that guide learning in the multiclass setting [Rubinstein, Bartlett, and Rubinstein, 2006, Daniely, Sabato, and Shalev-Shwartz, 2012, Daniely and Shalev-Shwartz, 2014, Daniely, Sabato, Ben-David, and Shalev-Shwartz, 2015a, Daniely, Schapira, and Shahaf, 2015b]. These

works revealed a stark contrast between a bounded and an unbounded number of labels. One important example is that the celebrated empirical risk minimization (ERM) principle ceases to apply when the number of labels is unbounded [Daniely and Shalev-Shwartz, 2014].

Algorithmic ideas of Haussler, Littlestone, and Warmuth [1994] and Rubinstein, Bartlett, and Rubinstein [2006] lead Daniely and Shalev-Shwartz [2014] to identify a *universal* family of transductive learning rules called *one-inclusion graph* (OIG) algorithms. Universality means that every learnable class can be learned by OIG algorithms. This universality and the combinatorial structure of OIG algorithms guided them to a new dimension. We call this new dimension the *Daniely-Shalev-Shwartz (DS)* dimension. They proved that finite DS dimension is a necessary condition for PAC learnability. But they too left the full characterization of learnability open.

Remark. *We use standard terminology from PAC learning and standard measurability assumptions (see e.g. the textbook [Shalev-Shwartz and Ben-David, 2014] and references within). All the relevant dimensions are defined and discussed in Section 2.*

1.1. **Results.** Our main result is that the DS dimension characterizes PAC learnability in the multiclass setting.

Theorem A (Learnability ≡ Finite DS Dimension)**.** *The following are equivalent for a concept class $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$:*

  – *The class $\mathcal{H}$ is PAC learnable.*
  – *The DS dimension of $\mathcal{H}$ is finite.*

We complement this result by refuting the conjecture that the Natarajan dimension characterizes learnability.

Theorem B (Learnability ≢ Finite Natarajan Dimension)**.** *Finite Natarajan dimension does not characterize PAC learnability.*

The two theorems follow from more informative results as we describe next. Because Daniely and Shalev-Shwartz [2014] proved that finite DS dimension is a necessary condition for PAC learnability, Theorem A follows from the following algorithmic result.

Theorem 1. *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be an hypothesis class with DS dimension $d < \infty$.*

Realizable Case: *There is a learning algorithm A^{real} for $\mathcal{H}$ with the following guarantees. For every $\mathcal{H}$-realizable distribution $\mathcal{D}$, every $\delta > 0$ and every integer n, given an input sample $S \sim \mathcal{D}^n$, the algorithm A^{real} outputs an hypothesis $h = A^{real}(S)$ such that[1]*

$$\Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y] \leq \tilde{O}\left(\frac{d^{3/2} + \log(1/\delta)}{n} \right)$$

with probability at least $1 - \delta$ over S.

---

[1]The $\tilde{O}$ notation conceals polylog(n, d) factors. Logarithms in this text are always in base two.

Agnostic Case: *There is a learning algorithm A^{agn} for $\mathcal{H}$ with the following guarantees. For every distribution $\mathcal{D}$, every $\delta > 0$ and integer n, given an input sample $S \sim \mathcal{D}^n$, the algorithm A^{agn} outputs an hypothesis $h = A^{agn}(S)$ such that*

$$\Pr_{(x,y) \sim \mathcal{D}}[h(x) \neq y] \leq L_{\mathcal{D}}(\mathcal{H}) + \tilde{O}\left(\sqrt{\frac{d^{3/2} + \log(1/\delta)}{n}}\right)$$

with probability at least $1 - \delta$, where $L_{\mathcal{D}}(\mathcal{H}) = \inf_{g \in \mathcal{H}} \Pr_{(x,y) \sim \mathcal{D}}[g(x) \neq y]$.

Because finite DS dimension is a necessary condition for learnability, Theorem B boils down to the following statement.

Theorem 2. *There exists a concept class $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ with Natarajan dimension 1 and an infinite DS dimension.*

1.2. **Roadmap.** In Section 2, we define the Natarajan dimension and the DS dimension. We also introduce the reader to the DS dimension and its basic properties. The central goal is to explain the important links between the three fundamental concepts: learnability, one-inclusion graphs, and the DS dimension.

In Section 3, we review the shifting mechanism. Shifting is a combinatorial technique used by Haussler [1995] to analyze OIG algorithms in the binary setting. Rubinstein, Bartlett, and Rubinstein [2006] later extended shifting to analyze OIG algorithms in the multiclass setting. The multiclass setting introduces subtleties and difficulties compared to the binary setting (see Examples 19 and 20 below). To overcome these difficulties, we introduce a new combinatorial dimension, the *exponential* dimension, which might be interesting in its own right.

Section 4 contains the proof of the equivalence between finite DS dimension and PAC learnability. The section begins with an overview of the main challenges that arise and the algorithmic ideas used to overcome them. Specifically, we introduce and discuss the notion of *list* PAC learning, which we believe should be of independent interest.

In Section 5 we prove that the Natarajan dimension does not characterize PAC learnability. This section has two parts. One part describes a general and basic connection between concept classes and properly colored simplicial complexes. The second part uses a deep and beautiful construction by Januszkiewicz and Świątkowski [2003] of a simplicial complex that exactly meets our needs. We provide a simplified and high-level exposition to their construction.

2. The DS Dimension and One-inclusion Graphs

The prime purpose of this section is to build the bridge between the DS dimension and learnability. We start with an introduction to the DS dimension, and a description of some of its simple properties. We continue with a description of the *one-inclusion graph* algorithm. The section concludes with the story of the "duality" that links between the two.

2.1. **Dimensions and Pseudo-cubes.** All dimensions we consider follow a similar mechanism. The main part is defining a notion of "shattering" that captures some local complexity of $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$. For $S \in \mathcal{X}^n$, we consider the projection $\mathcal{H}|_S$ of $\mathcal{H}$ to S, and say that $\mathcal{H}$ shatters S if $\mathcal{H}|_S$ is "complex" in some appropriate sense. The dimension is then defined as the maximum size (n) of a shattered sequence (if $\mathcal{H}$ shatters arbitrarily large sets then the dimension is defined to be ∞).

Notation. *We consider sequences in $\mathcal{X}^n$ instead of subsets of $\mathcal{X}$, because typically inputs to learning problems are sequences not sets. For $h : \mathcal{X} \to \mathcal{Y}$ and $S = (x_1, \ldots, x_n) \in \mathcal{X}^n$, the projection $h|_S$ of h to S is thought of as the map from $[n]$ to $\mathcal{Y}$ defined by $i \mapsto h(x_i)$. The projection of $\mathcal{H}$ to S is*

$$\mathcal{H}|_S = \{h|_S : h \in \mathcal{H}\} \subseteq \mathcal{Y}^n.$$

We sometimes think of $\mathcal{Y}^n$ as words of length n over the alphabet $\mathcal{Y}$.

The first and most well-known dimension is the VC dimension. It is defined only for binary classification problems.

Definition 3 (VC dimension [Vapnik and Chervonenkis, 1968])**.** We say that $S \in \mathcal{X}^n$ is *VC-shattered* by $\mathcal{H} \subseteq \{0, 1\}^{\mathcal{X}}$ if $\mathcal{H}|_S = \{0, 1\}^n$. The VC dimension $d_{VC}(\mathcal{H})$ is the maximum size of a VC-shattered sequence.

When $|\mathcal{Y}| > 2$, there are many ways to extend the VC dimension. One of the first extensions of the VC dimension to the multiclass setting is the Natarajan dimension. The relevant shattering is "containing a copy of the Boolean cube".

Definition 4 (Natarajan dimension [Natarajan, 1989])**.** We say that $S \in \mathcal{X}^n$ is *N-shattered* by $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ if there exist $f, g : [n] \to \mathcal{Y}$ such that for every $i \in [n]$ we have $f(i) \neq g(i)$, and

$$\mathcal{H}|_S \supseteq \{f(1), g(1)\} \times \{f(2), g(2)\} \times \ldots \times \{f(n), g(n)\}.$$

The Natarajan dimension $d_N(\mathcal{H})$ is the maximum size of an N-shattered sequence.

What is the "correct analog" of the Boolean cube for larger alphabet sizes? There are many possible answers. The starting point of the definition of the DS dimension is viewing the Boolean cube as a graph. The vertex-set of the graph is $\{0, 1\}^d$. The edges of the graph are defined as follows. For every vertex $v \in \{0, 1\}^d$ and for every direction $i \in [d]$, there is a (single) neighbor u of v in direction i (that is, $u(i) \neq v(i)$ and $u(j) = v(j)$ for all $j \neq i$). This perspective can be naturally applied to non-binary concept classes.

Definition 5 (Pseudo-cube)**.** A class $\mathcal{H} \subseteq \mathcal{Y}^d$ is called a *pseudo-cube* of dimension d if it is non-empty, finite and for every $h \in \mathcal{H}$ and $i \in [d]$, there is an i-neighbor $g \in \mathcal{H}$ of h (i.e., $g(i) \neq h(i)$ and $g(j) = h(j)$ for all $j \neq i$).

When $\mathcal{Y} = \{0, 1\}$, the two notions "Boolean cube" and "pseudo-cube" coincide: The Boolean cube $\{0, 1\}^d$ is of course a pseudo-cube. Conversely, every pseudo-cube $\mathcal{H} \subseteq \{0, 1\}^d$ is the entire Boolean cube $\mathcal{H} = \{0, 1\}^d$. When $|\mathcal{Y}| > 2$, the two notions do not longer coincide. Every copy of the Boolean cube is a pseudo-cube, but there are

pseudo-cubes that are not Boolean cubes; see Figure 3 for an example. The example in the figure uses a dual perspective. The functions (words) in the class are the edges of the graph, and the alphabet symbols are the vertices of the graph. This dual perspective is important and useful. We discuss it in more detail in Section 5.
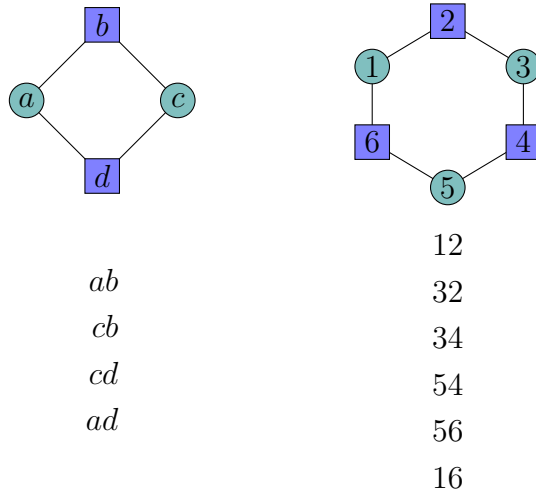
Figure 1. A 2-dimensional pseudo-cube (on the right) that is not isomorphic to the 2-dimensional Boolean cube (on the left). The labels $\mathcal{Y}$ are the vertices (4 label on the left, and 6 labels on the right). The words in $\mathcal{H} \subset \mathcal{Y}^2$ are the edges (4 words on the left, and 6 words on the right). For each word, the circle vertex appears as the first symbol, and the square appears as the second symbol.

The DS dimension is defined by containing pseudo-cubes (the original definition uses a slightly different language, but it is equivalent).

Definition 6 (DS dimension [Daniely and Shalev-Shwartz, 2014])**.** We say that $S \in \mathcal{X}^n$ is *DS-shattered* by $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ if $\mathcal{H}|_S$ contains an n-dimensional pseudo-cube. The DS dimension $d_{DS}(\mathcal{H})$ is the maximum size of a DS-shattered sequence.

How different are pseudo-cubes than Boolean cubes? Or, more formally, are there d-dimensional pseudo-cubes with Natarajan dimension $< d$? The hexagon in Figure 3 is a 2-dimensional pseudo-cube whose Natarajan dimension is 1. There are, in fact, many other such constructions, even in the 2-dimensional case.

The following example provides a complete description of 2-dimensional pseudo-cubes with Natarajan dimension 1 using the language of graph theory. We omit the proof because in Section 5 we derive generalizations to arbitrary dimensions.

Example 7. *For every bipartite graph $G = (L, R, E)$ the set*

$$B(G) := \big\{(u, v) \in L \times R : \{u, v\} \in E\big\}$$

is a 2-dimensional pseudo-cube if and only if G contains no leaves. Conversely, for every $B \subseteq \mathcal{Y}^2$, the bipartite graph

$$G(B) := \left(L = \mathcal{Y} \times \{0\}, R = \mathcal{Y} \times \{1\}, E = \left\{\{(y_0, 0), (y_1, 1)\} : (y_0, y_1) \in B\right\}\right)$$

contains no leaves if and only if B is a pseudo-cube. The claim is that a 2-dimensional pseudo-cube $B \subseteq \mathcal{Y}^2$ has Natarajan dimension 2 iff the corresponding bipartite graph $G(B)$ has a 4-cycle.

The above demonstrates that 2-dimensional pseudo-cubes are rather simple combinatorial objects. The landscape in higher dimensions is significantly richer. Figure 2 depicts a 3-dimensional pseudo-cube with Natarajan dimension 1. This pseudo-cube arises from a triangulation of the plane; a hint towards the topology that is used in Section 5 to prove Theorem 2.
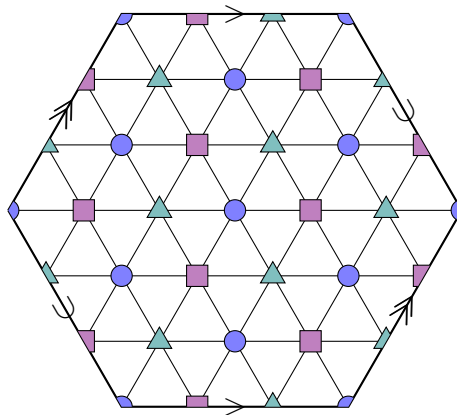

Figure 2. An example of a 3-dimensional pseudo-cube with Natarajan dimension 1. The words in the pseudo-cube are the triangles (there are 54 of them). The labels are the vertices (there are $\frac{3 \cdot 54}{6} = 27$ of them). The vertices are colored by 3 colors: circle, triangle and square. In each word, the circle vertex appears as the first symbol, the triangle vertex as the second, and the square vertex as the third. Opposite sides of the hexagon are identified, as the picture indicates. The pseudo-cube property holds because each triangle has three neighboring triangles that are obtained by switching one vertex from each color. The Natarajan dimension is 1 because there is no square (a cycle of length four) in the graph so that its vertices have alternating colors. For more details, see Section 5.

The condition that pseudo-cubes are *finite* is surprisingly important. Without it, the DS dimension does not characterize learnability, as the following example shows.

Example 8. *There is an infinite learnable class $\mathcal{H}^{tree}$ over $\mathcal{X} = \mathbb{N}$ so that for each $x \in \mathcal{X}$ and $h \in \mathcal{H}^{tree}$, there is $g \in \mathcal{H}^{tree}$ that agrees with h on all points besides x. But the DS dimension of this class is 1, so it is learnable (by our main result). This class can be thought of as a directed tree whose edges are directed towards the root; Figure 3 illustrates a similar class for the case $\mathcal{X} = [3]$. The root is the all-zeros function.*
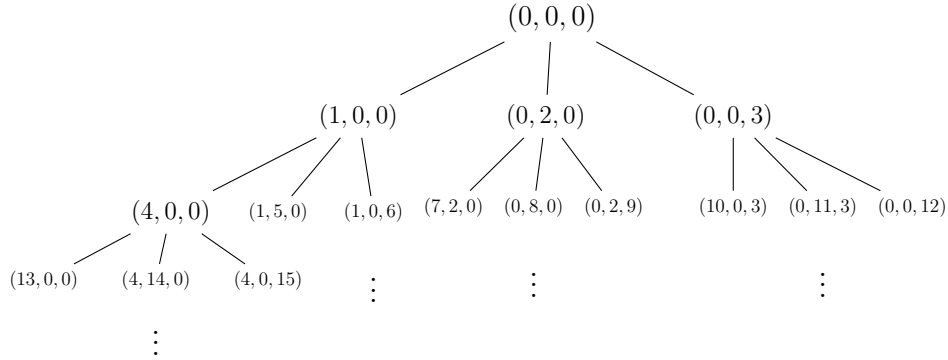
Figure 3. An example of an infinite class $\mathcal{H}^{tree} \subseteq \mathcal{Y}^{\mathcal{X}}$ with $\mathcal{X} = [3]$ and $\mathcal{Y} = \mathbb{N}$.

Each h in the tree has $\mathcal{X}$ in-going edges; for each $x \in \mathcal{X}$, there is an edge towards h from the function that is equal to h on $\mathcal{X} \setminus \{x\}$, and is equal to a new and unique alphabet symbol at x. Every alphabet symbol $y \in \mathcal{Y}$ has a depth*; it is the minimum distance from the root of a word that y appears in. The DS dimension is less than two for the following reason. For every $S \in \mathcal{X}^2$ and every finite $\mathcal{H}_0 \subset \mathcal{H}^{tree}|_S$, we can choose $y \in \mathcal{Y}$ with maximum depth among all symbols that appear in $\mathcal{H}_0$. Let h_0 be an element in $\mathcal{H}_0$ that contains y. The vertex h_0 does not have two neighbors, so $\mathcal{H}_0$ is not a pseudo-cube.*

2.2. **The One-Inclusion Graph.** This subsection introduces an important combinatorial abstraction of learning algorithms. The idea is to translate a learning problem to the language of graph theory.

Definition 9 (One-inclusion Graph [Haussler, Littlestone, and Warmuth, 1994, Rubinstein, Bartlett, and Rubinstein, 2006])**.** The *one-inclusion* graph of $\mathcal{H} \subseteq \mathcal{Y}^n$ is a hypergraph $\mathcal{G}(\mathcal{H}) = (V, E)$ that is defined as follows.[2] The vertex-set is $V = \mathcal{H}$. For each $i \in [n]$ and $f : [n] \setminus \{i\} \to \mathcal{Y}$, let $e_{i,f}$ be the set of all $h \in \mathcal{H}$ that agree with f on $[n] \setminus \{i\}$. The edge-set is

(1) $$E = \big\{(e_{i,f}, i) : i \in [n], f : [n] \setminus \{i\} \to \mathcal{Y}, e_{i,f} \neq \emptyset\big\}.$$

We say that the edge $(e_{i,f}, i) \in E$ contains the vertex v, and write $v \in (e_{i,f}, i)$, if $v \in e_{i,f}$. The size of the edge $(e_{i,f}, i)$ is defined to be $|(e_{i,f}, i)| := |e_{i,f}|$.

Remark. *The edge-set consists of pairs (e, i), where e is a set of vertices and i is the* direction *of the edge. It is convenient that the "name" of an edge also tells us its direction. Edges could be of size one, and each vertex v is contained in exactly n edges. This is not the standard structure of edges in hypergraphs, but we use this notation because it provides a better model for learning problems.*

The one-inclusion graph leads to a simple but useful toy model for transduction in machine learning.

––––––––––

[2]We use the term "one-inclusion graph" although it is actually a hypergraph.

Example 10 (Toy Model)**.** *The learning game is played over a one-inclusion graph* (V, E)*. The input to the problem is an edge. The input edge e is generated by first choosing a vertex v_* from some unknown distribution over V, and then choosing e to be a uniformly random edge containing v_*. The goal is to output a vertex u that is equal to v_* with as high probability as possible.*

Learning algorithms in this toy model are orientations.

Definition 11. An *orientation* of the hypergraph (V, E) is a mapping $\sigma : E \to V$ such that $\sigma(e) \in e$ for each edge $e \in E$.

Every (deterministic) learning algorithm defines an orientation, and vice versa. The success probability of the algorithm is determined by the out-degrees of the orientation. The *out-degree* of $v \in V$ in σ is

$$(2) \qquad \mathsf{outdeg}(v; \sigma) = |\{e : v \in e \text{ and } \sigma(e) \neq v\}|.$$

The maximum out-degree of σ is

$$(3) \qquad \mathsf{outdeg}(\sigma) = \sup_{v \in V} \mathsf{outdeg}(v; \sigma).$$

There is a certain duality between orientations and the DS dimension, as the following two lemmas demonstrate. This duality is the basic link between the DS dimension and learnability.

Lemma 12. *If $\mathcal{H} \subseteq \mathcal{Y}^d$ has DS dimension d, then $\mathsf{outdeg}(\sigma) \geq \frac{d}{2}$ for every orientation σ of $\mathcal{G}(\mathcal{H})$.*

Lemma 13. *If $\mathcal{H} \subseteq \mathcal{Y}^{d+1}$ has DS dimension d, then there exists an orientation σ of $\mathcal{G}(\mathcal{H})$ with $\mathsf{outdeg}(\sigma) \leq d$.*

Proof of Lemma 12. We prove the stronger assertion that if $\mathcal{H} \subseteq \mathcal{Y}^d$ is a pseudo-cube then every orientation σ of $\mathcal{H}$ satisfies that $\mathsf{outdeg}(\sigma) \geq \frac{d}{2}$. In a pseudo-cube, each h has a neighbor in each of the d directions, and every edge $e \in E$ has size $|e| \geq 2$ so that $|e| - 1 \geq \frac{|e|}{2}$. Even the average out-degree is at least $\frac{d}{2}$: for every orientation σ,

$$\frac{1}{|V|} \sum_{v \in V} \mathsf{outdeg}(v; \sigma) = \frac{1}{|V|} \sum_{e \in E} |e| - 1$$
$$\geq \frac{1}{|V|} \sum_{e \in E} \frac{|e|}{2}$$
$$= \frac{1}{2|V|} \cdot d|V| = \frac{d}{2}.$$

This finishes the proof because the maximum is at least the average. $\qquad \square$

Proof of Lemma 13. We start by analyzing the case that $\mathcal{H}$ is finite (similarly to [Daniely and Shalev-Shwartz, 2014]). The orientation is constructed inductively and greedily as follows. The base of the induction is the case $|\mathcal{H}| = 1$. In this case, all edges are oriented towards the single vertex, so the claim trivially holds. For the inductive step, assume $|\mathcal{H}| > 1$. Let $\mathcal{G} = (V, E)$ be the one-inclusion graph associated with

$\mathcal{H}$. Because the DS dimension of $\mathcal{H}$ is d, there must be $h \in \mathcal{H}$ so that the size of $\{e \in E : h \in e, |e| > 1\}$ is at most d. Let $\mathcal{H}'$ be $\mathcal{H}$ after deleting h. By definition, the DS dimension of $\mathcal{H}'$ is at most d. Let $\mathcal{G}' = (V', E')$ be the hyper-graph associated with $\mathcal{H}'$. Edges in E' are obtained from edges in E by deleting h. There is at least one singleton edge of size 1 that contains h in E. This edge does not appear in E'. By the induction hypothesis, there is an orientation $\sigma' : E' \to V'$ with maximum out-degree at most d. Every edge in E' corresponds to an edge in E. The only edges in E that do not have counterparts in E' are the singleton edges that contain h. Let $\sigma : E \to V$ extend σ' as follows. Every edge in E that has a counterpart in E' is directed in σ as in σ'. All other (singleton) edges are directed towards h. The out-degree of vertices in $\mathcal{G}'$ does not change, and the out-degree of h is at most d. So, the out-degree of σ is at most d as required.

The case when $\mathcal{H}$ is infinite is handled using a compactness argument. Because we could not find a proper reference, we provide the short (but not entirely trivial) proof in Appendix B.

$\square$

2.3. **The One-Inclusion Graph Algorithm.** The one-inclusion graph captures a model for transduction in machine learning (Example 10). A key observation of Haussler, Littlestone, and Warmuth [1994] is that this model captures an essential ingredient of general PAC learnability; see also [Rubinstein, Bartlett, and Rubinstein, 2006, Daniely and Shalev-Shwartz, 2014, Alon, Hanneke, Holzman, and Moran, 2021]. In a nutshell, good orientations of the one-inclusion graph yield good learning algorithms.

Algorithm 1 The one-inclusion algorithm $\mathcal{A}_{\mathcal{H}}$ for $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$

Input: An $\mathcal{H}$-realizable sample $S = \big((x_1, y_1), \ldots, (x_n, y_n)\big)$.
Output: A hypothesis $\mathcal{A}_{\mathcal{H}}(S) = h_S : \mathcal{X} \to \mathcal{Y}$.

For each $x \in \mathcal{X}$, the value $h_S(x)$ is computed as follows.
1: Consider the class of all patterns over the *unlabeled data* $\mathcal{H}|_{(x_1,\ldots,x_n,x)} \subseteq \mathcal{Y}^{n+1}$.
2: Find an orientation σ of $\mathcal{G}(\mathcal{H}|_{(x_1,\ldots,x_n,x)})$ that minimizes the maximum out-degree.

3: Consider the edge in direction $n+1$ defined by S; let
$$e = \{h \in \mathcal{H}|_{(x_1,\ldots,x_n,x)} : \forall i \in [n] \ \ h(i) = y_i\}.$$

4: Let $h' = \sigma((e, n+1))$.
5: Set $h_S(x) = h'(n+1)$.

The one-inclusion graph (OIG) algorithm $\mathcal{A}_{\mathcal{H}}$ is presented in Algorithm 1. The algorithm gets as input a realizable training sample $S = ((x_1, y_1), \ldots, (x_n, y_n))$ as well as an additional test point x. Its goal is to provide a good prediction for the label of x. The main idea is to translate this problem to the toy model. Use the unlabelled data

$x_1, \ldots, x_n$ and x to build the one-inclusion graph of $\mathcal{H}|_{(x_1,\ldots,x_n,x)}$. The labels $y_1, \ldots, y_n$ now define an edge in the graph. An orientation of the graph provides the prediction for the label of x.

The crucial point is that an orientation with small maximum out-degree yields small error. This follows by a simple and clever *leave-one-out* argument (see e.g. [Haussler, Littlestone, and Warmuth, 1994]). The argument exploits the underlying symmetry as we now explain.

Let $S \sim \mathcal{D}^n$ be the input sample and let $(x, y) \sim \mathcal{D}$ be the test point (chosen independently of S). We can generate the joint distribution of $(S, (x, y))$ in a different way. We can choose $S' \sim \mathcal{D}^{n+1}$ and independently choose I from the uniform distribution $U(n+1)$ on $[n+1]$. Let

$$S'_{-I} = ((x'_1, y'_1), \ldots, (x'_{I-1}, y'_{I-1}), (x'_{I+1}, y'_{I+1}), \ldots, (x'_{n+1}, y'_{n+1}))$$

be the sample S' after deleting its I entry. The distribution of $(S'_{-I}, (x'_I, y'_I))$ is identical to that of $(S, (x, y))$.

Fact 14 (Leave-one-out). *Let $\mathcal{D}$ be a distribution over a set $\mathcal{Z}$ and let $n > 0$ be an integer. For every event $E \subseteq \mathcal{Z}^{n+1}$,*

$$\Pr_{(S,Z)\sim\mathcal{D}^{n+1}}\Big[(S, Z) \in E\Big] = \Pr_{(S',I)\sim\mathcal{D}^{n+1}\times U(n+1)}\Big[(S'_{-I}, S'_I) \in E\Big].$$

The one-inclusion graph together with the leave-one-out argument lead to a formal connection between good orientations and PAC prediction error.

Proposition 15. *Let $\mathcal{D}$ be an $\mathcal{H}$-realizable distribution and let $n > 0$ be an integer. Let M be an upper bound on the maximum out-degree of all orientations chosen by $\mathcal{A}_\mathcal{H}$. The prediction error can be bounded as*

$$\Pr_{(S,(x,y))\sim\mathcal{D}^{n+1}}\big[h_S(x) \neq y\big] \leq \frac{M}{n+1},$$

where $h_S = \mathcal{A}_\mathcal{H}(S)$.

Proof. By Fact 14,

$$\Pr\big[h_S(x) \neq y\big] = \Pr\big[h_{S'_{-I}}(x'_I) \neq y'_I\big].$$

The prediction error is small, as long as the maximum out-degree is small: for every fixed $S' = ((x'_1, y'_1), \ldots, (x'_{n+1}, y'_{n+1}))$,

$$\begin{aligned}
\Pr_I\big[h_{S'_{-I}}(x'_I) \neq y'_I\big] &= \frac{1}{n+1}\sum_{i=1}^{n+1} 1\big[h_{S'_{-i}}(x'_i) \neq y'_i\big] \\
&= \frac{1}{n+1}\sum_{i=1}^{n+1} 1\big[\sigma(e_i) \neq y'_i\big] \\
&= \frac{\mathsf{outdeg}(y'; \sigma)}{n+1},
\end{aligned}$$

where $y' = (y'_1, \ldots, y'_{n+1})$ is a vertex the one-inclusion graph, and e_i is the edge in the i'th direction containing y'. $\qquad\square$

The final piece we present in this section is that a bound on the DS dimension leads to non-trivial prediction guarantees for PAC learning. This rather weak prediction capability is the starting point of our general PAC learning algorithm. It will be significantly enhanced in Section 4 below.

Claim 16. *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be so that $d = d_{DS}(\mathcal{H}) < \infty$. Let $\mathcal{A}_{\mathcal{H}}$ be Algorithm 1. For every $\mathcal{H}$-realizable sample $S' = ((x'_1, y'_1), \ldots, (x'_{d+1}, y'_{d+1}))$, there exists $i \in [d+1]$ such that $h_{S'_{-i}}(x'_i) = y'_i$, where $h_{S'_{-i}} = \mathcal{A}_{\mathcal{H}}(S'_{-i})$.*

Proof. Let $\mathcal{H}' = \mathcal{H}|_{(x'_1, \ldots, x'_{d+1})}$. Think of $y' = (y'_1, \ldots, y'_{d+1})$ as a vertex in $\mathcal{G}(\mathcal{H}')$. Let σ denote the orientation that minimizes the maximum out-degree of $\mathcal{G}(\mathcal{H}')$ chosen by $\mathcal{A}_{\mathcal{H}}$. Lemma 13 implies that the maximum out-degree of σ is at most d. Let e_i be the edge in the i'th direction containing y'. For every $i \in [d+1]$, we have $h_{S'_{-i}}(x'_i) \neq y'_i \Leftrightarrow \sigma(e_i) \neq y'$. So,

$$\sum_{i=1}^{d+1} 1\left[h_{S'_{-i}}(x'_i) \neq y'_i\right] = \sum_{i=1}^{d+1} 1\left[\sigma(e_i) \neq y'\right] = \mathsf{outdeg}(y'; \sigma) \leq d.$$

It follows that there must exist i such that $h_{S'_{-i}}(x'_i) = y'_i$. $\square$

3. Shifting and Orientations

In this section we use a powerful combinatorial technique called shifting to derive good orientations. This links the general discussion of one-inclusion graphs from the previous section, with the learning algorithm we use to prove Theorem A in the next section. The main result of this section is that the out-degree of optimal orientations can be controlled by the Natarajan dimension and the number of labels.[3]

Lemma 17. *Let $\mathcal{H} \subseteq [p]^n$ be a class with Natarajan dimension $d_N < \infty$. Then, there exists an orientation σ of $\mathcal{G}(\mathcal{H})$ with maximum out-degree*

$$\mathsf{outdeg}(\sigma) \leq 20 d_N \log p.$$

The key technique we use is shifting [Haussler, 1995, Rubinstein, Bartlett, and Rubinstein, 2006]. Shifting is a way to simplify the structure of a hypothesis class, while controlling important properties. Intuitively, it is the operation of "pushing a concept class downward". Think of $[p]$ as totally ordered by the standard order on $\mathbb{N}$. The set $[p]^n$ becomes a poset with the partial order $h \leq g$ iff $h(i) \leq g(i)$ for all i.

Definition 18 (Shifting). Let $\mathcal{H} \subseteq [p]^n$ and let $i \in [n]$. The shifting operator in the i'th direction $\mathbb{S}_i$ maps $\mathcal{H}$ to its shifted version $\mathbb{S}_i(\mathcal{H})$ as follows. Shifting is first defined on edges. For $f : [n] \setminus \{i\} \to [p]$, let e_f be the collections of $h \in \mathcal{H}$ that agree with f on $[n] \setminus \{i\}$. The shifting $\mathbb{S}_i(e_f)$ is obtained by "pushing e_f downward"; namely, $\mathbb{S}_i(e_f)$ is the collection of all $g \in [p]^n$ that agree with f on $[n] \setminus \{i\}$ and $1 \leq g(i) \leq |e_f|$. The shifting of $\mathcal{H}$ is the union of all shifted edges

$$\mathbb{S}_i(\mathcal{H}) = \bigcup_f \mathbb{S}_i(e_f) \subseteq [p]^n.$$

---

[3]Here and below we did not attempt to optimize the constants.

Let us provide a different view point on this important operation. Fix $j \neq i$, and partition all edges in the j'th direction according to their projection to $[n] \setminus \{i, j\}$. Fix $f \in \mathcal{H}|_{[n] \setminus \{i,j\}}$, and consider all vertices that agree with f on $[n] \setminus \{i, j\}$. Encode this data by the $p \times p$ Boolean matrix M_f defined by $M_f(a, b) = 1$ iff adding a, b to f in positions i, j leads to a word in $\mathcal{H}$. The 1-entries in the matrix correspond to words in $\mathcal{H}$ that agree with f. Every row in the matrix corresponds to the (possibly empty or singleton) set of words that differ in the j'th coordinate. Rows with at least one 1-entry correspond to edges in the one-inclusion graph. The matrix offers a nice viewpoint on shifting. Shifting is performed by pushing all the 1-entries "upwards". Here is an example of shifting six words over an alphabet of size four:

$$
\begin{bmatrix} 1 & & 1 & \\ & & & \\ & & 1 & \\ 1 & & 1 & 1 \end{bmatrix} \implies \begin{bmatrix} 1 & & 1 & 1 \\ 1 & & 1 & \\ & & 1 & \\ & & & \end{bmatrix}
$$

Repeatedly applying the shifting operator in various directions leads to a fixed point $\mathcal{H}_*$ of these operators; that is, $\mathbb{S}_i(\mathcal{H}_*) = \mathcal{H}_*$ for all i. This must happen in a finite number of steps, because when a change is made the total sum of all entries strictly decreases. The fixed points of shifting are classes that are closed downwards (that is, if h is in a fixed-point $\mathcal{H}_* \subseteq [p]^n$ and $g \leq h$ then $g \in \mathcal{H}_*$).

In the binary setting, Haussler [1995] proved that shifting does not increase the VC dimension, and that it does not decrease the number of edges in the one-inclusion graph. This allows to elegantly bound from above the edge density by the VC dimension.

In the multiclass setting, Rubinstein, Bartlett, and Rubinstein [2006] used the Pollard dimension [Pollard, 1990] to control the behavior of multiclass shifting; the Pollard dimension provides a natural mechanism for moving from the multiclass setting to the binary setting. But the Pollard dimension and other standard dimensions can grow during shifting; see Example 19 below. In addition, the number of edges and their total size can decrease; see Example 20.

Example 19 (Dimensions Increase)**.**

$$
\begin{array}{lcl}
(1,1) & & (1,1) \\
(1,0) & & (0,0) \\
(0,1) & \implies & (0,1) \\
(2,0) & \mathbb{S}_1 & (1,0) \\
(0,2) & & (0,2)
\end{array}
$$

Before shifting all three dimensions—Natarajan, DS and Pollard—are 1. *After shifting they are* 2.

Example 20 (Edges Decrease)**.**

$$
\begin{array}{lcl}
(2,2) & & (0,2) \\
(1,1) & \implies & (0,1) \\
(1,0) & \mathbb{S}_1 & (0,0) \\
(2,0) & & (1,0)
\end{array}
$$

Before shifting, the three non-singleton edges are $\{(2,2),(2,0)\}$, $\{(1,1),(1,0)\}$, *and* $\{(1,0),(2,0)\}$, *and the sum of their sizes is* 6. *After shifting, there are two non-singleton edges* $\{(0,0),(0,1),(0,2)\}$ *and* $\{(0,0),(1,0)\}$, *and the sum of their sizes is* 5. *In the binary case, the sum of the sizes of edges is equivalent to the average degree, and it does not decrease during shifting.*

These examples show that the analysis of multiclass shifting is not a direct extension of the arguments in the binary case. We now identify two quantities that are similar to VC dimension and average degree, but can be controlled during shifting.

Because multiclass shifting is "complex", we seek the simplest possible dimension so that we can keep track of it.

Definition 21 (Exponential Dimension). We say that $S \in \mathcal{X}^n$ is E-shattered by $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ if $|\mathcal{H}|_S| \geq 2^n$. The exponential dimension $d_E(\mathcal{H})$ is the maximum size of an E-shattered sequence.

The following claim shows that the exponential dimension is not increased during shifting.

Claim 22 (Shifting Does Not Increase Projections). *Let* $\mathcal{H} \subseteq [p]^n$ *and let* $i \in [n]$. *For every* $S \in [n]^k$,

$$\left|\mathbb{S}_i(\mathcal{H})|_S\right| \leq \left|\mathcal{H}|_S\right|.$$

Proof. Without loss of generality, assume that S has k distinct entries. Recall that $\mathcal{H}|_S$ is a subset of $[p]^k$. We assume that $k > 1$; the proof when $k = 1$ is similar. If i does not appear in S, shifting does not change the projection. If i appears in S, argue as follows. Let S' be S after deleting i, so that $\mathcal{H}|_{S'}$ is a subset of $[p]^{k-1}$. For each $f \in \mathcal{H}|_{S'}$, let e_f be the set of $h \in \mathcal{H}|_S$ that agree with f on $[k-1]$, so that

$$|\mathcal{H}|_S| = \sum_f |e_f|.$$

Let e'_f be the set of $h \in \mathbb{S}_i(\mathcal{H})|_S$ that agree with f on $[k-1]$. Because $\mathcal{H}|_{S'}$ is equal to $\mathbb{S}_i(\mathcal{H})|_{S'}$, we similarly have

$$|\mathbb{S}_i(\mathcal{H})|_S| = \sum_f |e'_f|.$$

For fixed f, the size $|e'_f|$ is equal to the maximum size of an edge in the i'th direction of $\mathcal{H}$ whose elements agree with f. This holds because there is an edge in the i'th direction of $\mathbb{S}_i(\mathcal{H})$ of size $|e'_f|$ whose elements agree with f, and the sizes of edges do not change during shifting. It follows that $|e_f| \geq |e'_f|$.

$\square$

Corollary 23. *For every* $\mathcal{H} \subseteq [p]^n$ *and* $i \in [n]$,

$$d_E(\mathbb{S}_i(\mathcal{H})) \leq d_E(\mathcal{H}).$$

We would like to control the structure of edges during shifting. The most obvious measure to keep track of is the average degree (with respect to non-singleton edges).

Definition 24 (Average Degree). Let $\mathcal{G}(\mathcal{H}) = (V, E)$ be the one-inclusion graph of $\mathcal{H} \subseteq [p]^n$. The average degree of $\mathcal{H}$ is

$$\mathsf{avd}(\mathcal{H}) = \frac{1}{|V|} \sum_{v \in V} \mathsf{deg}(v) = \frac{1}{|V|} \sum_{e \in E : |e| > 1} |e|,$$

where $\mathsf{deg}(v) = |\{e \in E : v \in e, |e| > 1\}|$.

Example 20 shows that the number of edges and average degree may decrease during shifting (which is bad for our purposes). The correct measure to keep track of turns out to be the following.

Definition 25 (Shifting Average Degree). Let $\mathcal{G}(\mathcal{H}) = (V, E)$ be the one-inclusion graph of $\mathcal{H} \subseteq [p]^n$. Define

$$\mathsf{avd}'(\mathcal{H}) = \frac{1}{|V|} \sum_{e \in E} (|e| - 1).$$

Claim 26 (Shifting Does Not Decrease avd'). *For every $\mathcal{H} \subseteq [p]^n$ and $i \in [n]$,*

$$\mathsf{avd}'(\mathbb{S}_i(\mathcal{H}))) \geq \mathsf{avd}'(\mathcal{H}).$$

Proof. Since $|V|$ does not change during shifting, we only need to record the changes in the edges. Let i denote the direction of shifting. The sizes of all edges in the i'th direction do not change during shifting. We need to understand the behavior in the other directions. We shall use the perspective explained after Definition 18. Fix $j \neq i$, and partition all edges in the j'th direction according to their projection to $[n] \setminus \{i, j\}$. Fix $f \in \mathcal{H}|_{[n] \setminus \{i,j\}}$, and consider all vertices that agree with f on $[n] \setminus \{i, j\}$. Encode this data by the $p \times p$ Boolean matrix M_f defined by $M_f(a, b) = 1$ iff adding a, b to f in positions i, j leads to a word in $\mathcal{H}$. The example we saw earlier helps to digest the scenario we are operating in (shifting six words over an alphabet of size four):

$$\begin{bmatrix} 1 & & 1 & \\ & & & \\ & & 1 & \\ 1 & & 1 & 1 \end{bmatrix} \implies \begin{bmatrix} 1 & & 1 & 1 \\ 1 & & 1 & \\ & & 1 & \\ & & & \end{bmatrix}$$

The sum of $|e| - 1$ over all edges e in the j'th direction that agree with f can be expressed as the total number of 1-entries in the matrix minus the number of non-zero rows. This is true for $\mathcal{H}$ as well for $\mathbb{S}_i(\mathcal{H})$. The total number of 1-entries remains fixed. The number of non-zero rows can not increase during shifting, because the number of non-zero rows after shifting is equal to the maximum number of 1-entries in a single column. It follows that this sum over $|e| - 1$ can not decrease, as claimed. □

The control of the exponential dimension and of avd' allows to bound the average degree.

Proposition 27 (Average Degree is Bounded by Exponential Dimension). *For every $\mathcal{H} \subseteq [p]^n$,*

$$\mathsf{avd}(\mathcal{H}) \leq 4d_E(\mathcal{H}).$$

Proof. Apply shifting on $\mathcal{H}$ until a fixed point $\mathcal{H}_*$ is reached. Because

$$\mathsf{avd}(\mathcal{H}) \leq 2\mathsf{avd}'(\mathcal{H}) \leq 2\mathsf{avd}'(\mathcal{H}_*) \leq 2\mathsf{avd}(\mathcal{H}_*),$$

it suffices to prove that $\mathsf{avd}(\mathcal{H}_*) \leq 2d_E(\mathcal{H}_*)$. This holds by induction. The induction base $|\mathcal{H}_*| = 1$ trivially holds. The induction step is justified as follows. Let h_0 be the concept in $\mathcal{H}_*$ so that no concept in $\mathcal{H}_*$ is larger than h_0 with respect to the natural partial order. Let $|h_0|$ be the number of entries that are larger than 1 in h_0. The fixed point property of $\mathcal{H}_*$ implies that it is closed downwards. There are $2^{|h_0|}$ words under h_0 in $\mathcal{H}_*$. It follows that $\mathsf{deg}(h_0) \leq |h_0| \leq d_E(\mathcal{H}_*)$. Remove h_0 from $\mathcal{H}_*$. This removal does not increase the exponential dimension, and the resulting class is still closed downwards. The inductive assumption completes the proof, because the sum of the degrees is reduced by at most $2\mathsf{deg}(h_0)$. $\qquad\square$

The bound on the average degree immediately allows to build good orientations.

Corollary 28 (Exponential Dimension Leads to Orientations)**.** *For every $\mathcal{H} \subseteq [p]^n$, there is an orientation of $\mathcal{G}(\mathcal{H})$ with maximum out-degree at most $4d_E(\mathcal{H})$.*

Proof. Proposition 27 implies that every induced sub-graph of $\mathcal{G}(\mathcal{H})$ has a vertex of degree at most $4d_E(\mathcal{H})$. The beginning of the proof of Lemma 13 produces the needed orientation; the orientation is constructed "greedily" by picking a vertex of degree at most $4d_E(\mathcal{H})$, removing it from the graph and proceeding recursively. $\qquad\square$

The last piece of the puzzle is to relate the exponential dimension to the Natarajan dimension. This is achieved via a generalization of Sauer's lemma by Haussler and Long [1995].

Lemma 29 (Controlling the Exponential Dimension)**.** *For every $\mathcal{H} \subseteq [p]^n$ with $d_N = d_N(\mathcal{H})$ and $d_E = d_E(\mathcal{H}) < \infty$,*

$$d_E \leq 5d_N \log(p).$$

Proof. Corollary 5 in [Haussler and Long, 1995] says that for every $S \subseteq [n]$ of size m, the size of $\mathcal{H}|_S$ is at most

$$\sum_{i=1}^{d_N} \binom{m}{i}\binom{p}{2}^i \leq \Big(\frac{p^2 em}{2d_N}\Big)^{d_N}.$$

By definition, the exponential dimension satisfies

$$2^{d_E} \leq \Big(\frac{p^2 e d_E}{2d_N}\Big)^{d_N}.$$

This implies the lemma because $p \geq 2$. $\qquad\square$

Remark. *Corollary 28 and Lemma 29 imply Lemma 17.*

4. Learnability ≡ Finite DS Dimension

Here we prove the characterization of multiclass PAC learnability via the DS dimension (Theorem 1). Our main contribution is algorithmic. We develop a learning algorithm for any class $\mathcal{H}$ with finite DS dimension.

4.1. **Outline.** The starting point is the OIG algorithm by Haussler, Littlestone, and Warmuth [1994]; see Section 2.3 above for a reminder. The finiteness of the DS dimension translates to a non-trivial guarantee on the OIG algorithm (as we saw in Claim 16). The output hypothesis of this algorithm has expected prediction error at most $1 - \frac{1}{d+1}$. This error is pretty high, but the crucial point is that it is uniformly bounded away from 1. The OIG algorithm forms a kind of a (very) weak PAC learner.

It is tempting to try to improve the error by boosting. But standard boosting turns out to be useless in our context. The traditional assumption for boosting in the binary setting requires error below $\frac{1}{2}$. The above error guarantee is too weak and does not meet the minimal requirements for boosting. And even if multiclass boosting was available, known techniques have sample complexity that scales with $\log|\mathcal{Y}|$; see [Schapire and Freund, 2012, Brukhim, Hazan, Moran, and Schapire, 2021]. This factor could be infinite in our setting. To circumvent this obstacle we introduce the framework of list PAC learning.

List PAC learning. In the standard PAC setting, the goal is to provide a single prediction on an unseen data point. In list PAC learning, the goal is to provide a short menu of predictions. Given a sample $S \sim \mathcal{D}^n$ from a realizable $\mathcal{D}$, the goal is to output a menu μ that maps elements of $\mathcal{X}$ to a small subset of $\mathcal{Y}$ so that $y \in \mu(x)$ with high probability over a new test point $(x, y) \sim \mathcal{D}$. List PAC learning is discussed in greater detail in Section 4.2.

Rather than boosting the weak OIG algorithm to a strong PAC learner, we use it to derive a list PAC learning algorithm. We show that every class $\mathcal{H}$ with a finite DS dimension admits a list PAC learner (see Algorithm 2). This list-learner gathers information from several OIG algorithms to produce a good menu. Its analysis is based on the leave-one-out symmetrization argument. The list-learner allows to eliminate the vast majority of a priori possible labels. Instead of all of $\mathcal{Y}$, we can safely use the menu $\mu(x)$ as the "local alphabet for x". Menus can be thought of as tools for alphabet reduction. Once we have a list PAC learner, it is natural to try to reduce the learning task to one in which the number of labels is bounded.

Did we just reduce the infinite alphabet case to the finite case? The short answer is no. Even with a good menu μ, the subclass $\mathcal{H}|_\mu = \{h \in \mathcal{H} : \forall x \in \mathcal{X} \ h(x) \in \mu(x)\}$ of $\mathcal{H}$ may be completely useless. For example, let $\mathcal{H} \subseteq \{0, 1, 2\}^{\mathbb{N}}$ be the set of all functions h such that $|h^{-1}(\{1, 2\})| < \infty$, and let μ be the menu such that $\mu(x) = \{1, 2\}$ for all $x \in \mathbb{N}$. The menu-subclass $\mathcal{H}|_\mu$ is just empty. At the same time, every finitely supported distribution $\mathcal{D}$ with labels in $\{1, 2\}$ is both realizable by $\mathcal{H}$ and consistent with μ. This simple example indicates that in order to restrict to a subclass of $\mathcal{H}$ without losing essential information, at least some knowledge on the support of the target distribution is needed. Learning the support of a distribution, however, is a much harder task than PAC learning.

Let us make a quick comment. In this work, list PAC learning serves as a tool for proving Theorem 1. However, we think it is a natural setting and interesting in its own right (and we prove further motivation in Section 4.2).
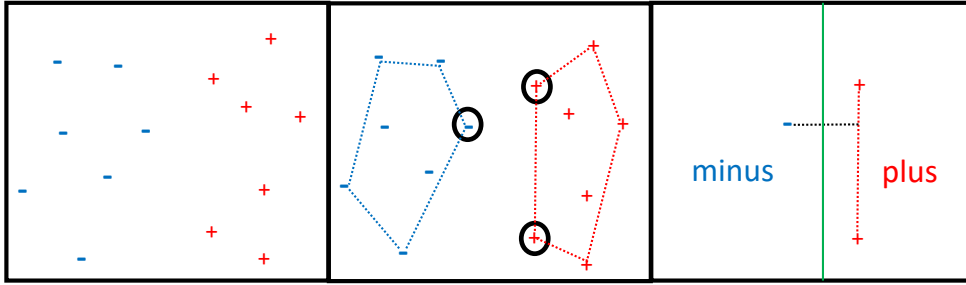
Figure 4. An illustration of Support Vector Machine in 2D as a sample compression scheme. The realizable sample S consists of negative and positive points. The algorithm outputs a separating line that maximizes the margin. This line is determined by the support vectors (circled).

List PAC Learning $\Rightarrow$ PAC Learning. Our solution is based on the fact that the OIG algorithm is exactly suitable for situations in which the learning task is not defined by a concept class, but by a set of allowable samples. The main property of OIG algorithms is their *locality*. To make a prediction on x, they just use the part of $\mathcal{H}$ that is relevant to the training data S, and do not require any global access to $\mathcal{H}$.

An alternative way to model learning with a menu is via partial concept classes [Alon, Hanneke, Holzman, and Moran, 2021]. Instead of all maps in $\mathcal{H}$ that are consistent with the menu μ, we can consider all partial maps that agree with both the class $\mathcal{H}$ and the menu μ. We chose not to use this formalism here in order to use as standard language as possible. The partial concept class perspective does not really help to solve the problem. The focus of Alon, Hanneke, Holzman, and Moran [2021] was on binary-classification, which is significantly simpler than the multiclass setting. Generalizing the analysis of the one-inclusion graph from the binary setting to the multiclass setting turns out to be a subtle (and somewhat confusing) task. Natural attempts to do so fail; see Section 3 for a full discussion.

Sample Compression Schemes. The algorithm we develop is best thought of as a *sample compression scheme* [Littlestone and Warmuth, 1986]. A sample compression scheme (Definition 35) is an abstraction of a common property to many learning algorithms; see Figure 4. It can be viewed as a two-party protocol between a *compresser* and a *reconstructor*. Both players know the underlying concept class $\mathcal{H}$. The compresser gets as input an $\mathcal{H}$-realizable sample S. The compresser picks a small subsample S' of S and sends it to the reconstructor. The reconstructor outputs an hypothesis h. The correctness criteria is that h needs to correctly classify *all* examples in the input sample S.

One advantage of using the sample compression schemes framework is that the proofs are typically cleaner, because in contrast to the probabilistic nature of the PAC framework, sample compression is a deterministic task. At the same time, sample compression schemes are known to represent good PAC learning algorithms [Littlestone and Warmuth, 1986].

Classical sample compression algorithms usually boil down to a simple one-shot encoding scheme (e.g. Figure 4). Our compression scheme is more involved and is comprised of two main components. The first component is a variant of sample compression that fits into the list-learning framework (Definition 38). The second component incorporates the menu derived by the first component together with a minimax-based sample compression as in [David, Moran, and Yehudayoff, 2016]. All of this is described in Section 4.4.

A high-level outline of the full algorithm is presented in Figure 5, after all the needed ingredients are prepared and the subtleties are discussed.

4.2. **List PAC learning.**

List PAC learning is a model for providing a short menu of likely predictions. It extends the standard PAC model by allowing the learning algorithm more freedom.

Relaxing the demand of a single output to a list of outputs is a common and useful paradigm in computer science. One notable example is the notion of *list-decoding* in coding theory, which is important both as a tool and as a goal.

Let us start with a few examples for list learning. In medical contexts, list-learning can offer physicians a menu of likely diagnoses. In technical contexts, list-learning can provide a short menu of possible solutions that are meant to assist clients. List-learning can also provide the menu of preferences of consumers. One can easily imagine other scenarios where list-learning is useful.

Our main motivation for developing this model is reasoning on the first component of our multiclass learning algorithm. But this basic model naturally fits into many scenarios, and we plan to investigate it further in future works.

The goal of list PAC learning is to compute good menus.

Definition 30 (*p*-menu)**.** A menu of size $p \in \mathbb{N}$ is a function $\mu : \mathcal{X} \to \{Y \subseteq \mathcal{Y} : |Y| \leq p\}$.

List PAC learning is the following natural version of standard PAC learning.

Definition 31 (List PAC Learner)**.** An algorithm A with sample size n and list size p is a *list PAC learner* with success probability $\alpha > 0$ for the concept class $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ if for every $\mathcal{H}$-realizable distribution $\mathcal{D}$,

$$\Pr_{(S,(x,y))\sim\mathcal{D}^{n+1}}\big[y \in \mu_S(x)\big] \geq \alpha,$$

where $\mu_S = A(S)$ is always a p-menu.

Remark. *In the "noisy" case, when the label y has entropy given x, list learning can potentially lead to zero error even though in the standard PAC setting zero error is not achievable.*

The main result of this section is the development of a list PAC learner for every class of finite DS dimension (see Algorithm 2). The list PAC learner can be thought of as a brute-force extension of the one-inclusion learning rule.

Algorithm 2 List PAC learner $\mathcal{L}_{\mathcal{H},t}$ for $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ with $d_{DS}(\mathcal{H}) = d$ and $t \in \mathbb{N}$

Input: Data $S \in (\mathcal{X} \times \mathcal{Y})^n$ where $n = d + t$.
Output: A p-menu μ_S for $p = \binom{n}{t}$.
1: Let $S_1, \ldots, S_p$ denote all subsamples of S of size d.
2: Let $h_{S_i} = \mathcal{A}_{\mathcal{H}}(S_i)$ denote the hypothesis output of Algorithm 1 on input sample S_i.
3: Return the menu defined by
$$\mu_S(x) = \{h_{S_1}(x), \ldots, h_{S_p}(x)\}.$$

Proposition 32 (Finite DS Dimension implies List PAC Learning). *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class with DS dimension $d < \infty$ and let $t \in \mathbb{N}$. The algorithm $\mathcal{L}_{\mathcal{H},t}$ is a list PAC learner for $\mathcal{H}$ with sample size $n = d + t$, list size $p = \binom{n}{t}$ and success probability $\alpha = \frac{t+1}{d+t+1}$.*

Proof. Let $\mu_S = \mathcal{L}_{\mathcal{H},t}(S)$ be the menu generated by the algorithm with data S. By the leave-one-out symmetrization argument (Fact 14),
$$\Pr_{(S,(x,y))\sim\mathcal{D}^{n+1}}\left[y \in \mu_S(x)\right] = \Pr_{(S',I)\sim\mathcal{D}^{n+1}\times U([n+1]])}\left[y'_I \in \mu_{S'_{-I}}(x'_I)\right].$$

It hence suffices to show that every realizable sample S' of size $n + 1$ satisfies

(4)
$$\Pr_{I\sim U([n+1]])}\left[y'_I \in \mu_{S'_{-I}}(x'_I)\right] \geq \frac{t+1}{d+t+1}.$$

Let us call an index $i \in [n+1]$ *good* if $y'_i \in \mu_{S'_{-i}}(x'_i)$. We need to show that there are at least $t+1$ good indices. By Claim 16, at least one of the indices in $[d+1]$ is good. Denote this good index by i_1. Again, by Claim 16, at least one of the indices in $[d+2] \setminus \{i_1\}$ is good. Denote this good index by i_2. Repeat this process to obtain the needed $t+1$ good indices. $\square$

4.3. **Learning Natarajan Classes From Menus.** We now move towards the second component of our algorithm. The objective is to use the good menu that was generated by the first component to effectively reduce the number of labels. The algorithm we develop in this sub-section is a weak PAC learner, but under a strong assumption. Several such weak learners will be combined later on to get the full sample compression scheme.

The learning algorithm now has two pieces of knowledge: the underlying class $\mathcal{H}$ and the menu μ. Trusting that the first component delivered on its promise, it assumes that the data is consistent with the menu. This is captured by the following definition.

Definition 33 (Menu Realizability). A sample $S \in (\mathcal{X} \times \mathcal{Y})^n$ is *realizable* by the menu μ if $y \in \mu(x)$ for every (x, y) in S. A distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ is *realizable* by μ if for every $m \in \mathbb{N}$, a random sample $S \sim \mathcal{D}^m$ is realizable by μ with probability 1.

This definition captures the ideal scenario that we have a menu that completely captures the unknown distribution $\mathcal{D}$. It is basically impossible to generate a single

menu that captures all of $\mathcal{D}$. Nevertheless, this idealization is a useful sub-goal that we need to deal with later on.

Algorithm 3 One-inclusion algorithm $\mathcal{A}_{\mathcal{H},\mu}$ for a class $\mathcal{H}$ and menu μ

Input: A sample $S = \big((x_1, y_1), \ldots, (x_n, y_n)\big)$ realizable by $\mathcal{H}$ and μ.
Output: A hypothesis $h_S : \mathcal{X} \to \mathcal{Y}$.

For each $x \in \mathcal{X}$, the value $h_S(x)$ is computed as follows.
1: Consider the class $\mathcal{H}' \subseteq \mathcal{Y}^{n+1}$ of all patterns on the unlabelled data that are realizable by both $\mathcal{H}$ and μ. That is, it is the set of all $h \in \mathcal{H}|_{(x_1,\ldots,x_n,x)}$ so that $h(n+1) \in \mu(x)$ and $h(i) \in \mu(x_i)$ for $i \in [n]$.
2: Find an orientation σ of $\mathcal{G}(\mathcal{H}')$ that minimizes the maximum out-degree.
3: Consider the edge in direction $n+1$ that is consistent with S. Let

$$e = \big\{ h \in \mathcal{H}' : \forall i \in [n]\ h(i) = y_i \big\}.$$

4: Let $h' = \sigma((e, n+1))$.
5: Set $h_S(x) = h'(n+1)$.

The main result of this sub-section is a PAC learning algorithm for menu-realizable distributions (Algorithm 3). The sample complexity is controlled by the size of the menu μ as well the Natarajan dimension of $\mathcal{H}$. This is pretty good news because we controlled the size of the menu, and the Natarajan dimension is the smallest among all dimensions.

Proposition 34 (PAC Learning Given a Menu). *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class with Natarajan dimension $d_N < \infty$ and let μ be a p-menu. For every distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ that is realizable by both $\mathcal{H}$ and by μ, and for all integers $n > 0$,*

$$\Pr_{(S,(x,y))\sim\mathcal{D}^{n+1}}\big[h_S(x) \neq y\big] \leq \frac{20 d_N \log(p)}{n},$$

where $h_S = \mathcal{A}_{\mathcal{H},\mu}(S)$.

The proposition is not the end of the story. The menu μ generated by the first component allows to make good list-predictions, but it has no chance to capture all of the unknown distribution $\mathcal{D}$. It is extremely unlikely that $\mathcal{D}$ is realizable by μ. The removal of this realizability assumption is postponed to the next section.

The high-level idea behind the proof of the proposition is to use the p-menu to reduce the label-set from the unbounded $\mathcal{Y}$ to a label-set of size p. This is beneficial because PAC learning with p many labels can be achieved with sample complexity order $d_N \log(p)$. In fact, any proper ERM algorithm with this sample complexity is a PAC learner.

Trying to implement this strategy raises a subtle challenge. The task of learning a distribution realizable by a class $\mathcal{H}$ and a menu μ *cannot* be reduced to PAC learning

the sub-class $\mathcal{H}|_\mu$ of $\mathcal{H}$ that is consistent with the menu. The sub-class $\mathcal{H}|_\mu$ could even be empty; see Section 4.1 for a simple example.

The solution is based on the unique locality feature of the OIG algorithm. To make a prediction on an unseen data point $x \in \mathcal{X}$, the OIG algorithm just uses x and the unlabelled part of the sample S. This local view of $\mathcal{X}$ suffices to make a prediction.

Proof of Proposition 34. Let $\mathcal{D}$ be a distribution that is realizable by $\mathcal{H}$ and μ. By the leave-one-out symmetrization argument (Fact 14),

$$\Pr_{(S,(x,y))\sim\mathcal{D}^{n+1}}\left[h_S(x) \neq y\right] = \Pr_{(S',I)\sim\mathcal{D}^{n+1}\times U(n+1)}\left[h_{S'_{-I}}(x'_I) \neq y'_I\right],$$

where $h_S = \mathcal{A}_{\mathcal{H},\mu}(S)$. It therefore suffices to show that for every sample S' that is realizable by $\mathcal{H}$ and μ,

$$(5) \qquad\qquad \Pr_{I\sim U(n+1)}\left[h_{S'_{-I}}(x'_I) \neq y'_I\right] \leq \frac{20d_N \log(p)}{n}.$$

Fix S' that is realizable by $\mathcal{H}$ and μ for the rest of the proof. The class $\mathcal{H}' = \mathcal{H}|_{(x'_1,\ldots,x'_{n+1})}$ constructed by the algorithm $\mathcal{A}_{\mathcal{H},\mu}$ for S'_{-I} and x'_I is the same for all values of I. The class $\mathcal{H}'$ is realizable by μ. The Natarajan dimension of $\mathcal{H}'$ is at most that of $\mathcal{H}$. Denote by σ the orientation of $\mathcal{G}(\mathcal{H}')$ that the algorithm chooses. Lemma 17 tells us that the maximum out-degree of σ is at most $20d_N \log(p)$. Let y' denote the vertex in $\mathcal{G}(\mathcal{H}')$ defined by $y'_1, \ldots, y'_{n+1}$. Finally,

$$\Pr_{I\sim U(n+1)}\left[h_{S'_{-I}}(x'_I) \neq y'_i\right] = \frac{\mathsf{outdeg}(y';\sigma)}{n+1} \leq \frac{20d_N \log(p)}{n+1}. \qquad\qquad \square$$

4.4. **The Algorithm.** We are ready to describe the full algorithm. It is convenient to think of it as a sample compression scheme.

Definition 35 (Sample Compression Scheme [Littlestone and Warmuth, 1986])**.** Let $\mathcal{H} \subseteq \mathcal{Y}^\mathcal{X}$ and let $r \leq n$ be integers. An $n \to r$ *sample compression scheme* consists of a *reconstruction* function

$$\rho : (\mathcal{X} \times \mathcal{Y})^r \to \mathcal{Y}^\mathcal{X}$$

such that for every $\mathcal{H}$-realizable $S \in (\mathcal{X} \times \mathcal{Y})^n$, there exists $S' \in (\mathcal{X} \times \mathcal{Y})^r$ whose elements appear in S such that for every (x,y) in S we have $h(x) = y$, where $h = \rho(S')$.

The main goal of this section is to construct a sample compression scheme for classes with finite DS dimension.

Theorem 36 (DS Classes are Compressible)**.** *Let $\mathcal{H} \subseteq \mathcal{Y}^\mathcal{X}$ be a class with DS dimension $d_{DS} < \infty$ and Natarajan dimension d_N. For every integers $n, t > 0$, there exists an $n \to r$ sample compression scheme for $\mathcal{H}$ with*

$$r \leq \left(\frac{d_{DS}+t+1}{t+1}(d_{DS}+t) + 10^3 d_N \log\left(\binom{d_{DS}+t+1}{t+1}\log(2n)\right)\right)\log(2n).$$

When d_{DS} and t are thought of as constants and n as tending to infinity, the value of r becomes $r \le \tilde{O}((\frac{(d_{DS}+t)^2}{t} + d_{DS}t)\log n)$. For $t = \lceil d_{DS}^{1/2} \rceil$, this becomes

$$r \le \tilde{O}(d_{DS}^{3/2} \log n).$$

As we explain in the next section, a standard "compression implies generalization" argument implies that $\tilde{O}(d_{DS}^{3/2})$ samples are sufficient for PAC learning.

The sample compression scheme consists of two components. The first component provides list-learning guarantees. It produces a good menu that is passed to the second component. The second component is a regular sample compression scheme but only for menu-realizable samples. To accommodate this mechanism, we introduce two variants of sample compression schemes.

Definition 37 (List Sample Compression Scheme). *An $n \to r$ list sample compression scheme with menu size p consists of a reconstruction function*

$$\rho : (\mathcal{X} \times \mathcal{Y})^r \to \{Y \subseteq \mathcal{Y} : |Y| \le p\}^{\mathcal{X}}$$

such that for every $\mathcal{H}$-realizable $S \in (\mathcal{X} \times \mathcal{Y})^n$, there exists $S' \in (\mathcal{X} \times \mathcal{Y})^r$ whose elements appear in S such that for every (x, y) in S we have $y \in \mu(x)$, where $\mu = \rho(S')$.

Definition 38 (Sample Compression Scheme for a Menu). *An $n \to r$ sample compression scheme for a class $\mathcal{H}$ and a menu μ consists of a reconstruction function*

$$\rho : (\mathcal{X} \times \mathcal{Y})^r \to \mathcal{Y}^{\mathcal{X}}$$

such that for every $S \in (\mathcal{X} \times \mathcal{Y})^n$ that is realizable by both $\mathcal{H}$ and μ, there exists $S' \in (\mathcal{X} \times \mathcal{Y})^r$ whose elements appear in S such that for every (x, y) in S we have $h(x) = y$, where $h = \rho(S')$.

The following two lemmas summarize the two components of the construction.

Lemma 39 (List Sample Compression Scheme). *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class with DS dimension $d_{DS} < \infty$. For every integers $n, t > 0$, there exists an $n \to r_1$ list sample compression scheme for $\mathcal{H}$ with menu size p, where*

$$r_1 \le \frac{d_{DS} + t + 1}{t + 1}(d_{DS} + t)\log(2n)$$

and

$$p \le \binom{d_{DS} + t + 1}{t + 1}\log(2n).$$

Lemma 40 (Sample Compression Given a Menu). *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class with Natarajan dimension $d_N < \infty$ and let μ be a p-menu. For every integer $n > 0$, there exists an $n \to r_2$ sample compression scheme for $\mathcal{H}$ and μ with*

$$r_2 \le 10^3 d_N \log(p) \log(2n).$$

The two lemmas, which are proved below, complete the proof of Theorem 36.

High-level Outline of the Algorithm

realizable case over $\mathcal{H}$ with $d = d_{DS}(\mathcal{H}) < \infty$

Input: A sample $S \in (\mathcal{X} \times \mathcal{Y})^n$.

1: Find $m' \approx \sqrt{d} \log n$ subsamples $S'_1, \ldots, S'_{m'}$ of S, each of size d, so that the menu μ of size $p \approx 2^{\sqrt{d}} \cdot m'$ they define realizes S.

2: Using the menu μ, find $m'' \approx \log n$ subsamples $S''_1, \ldots, S''_{m''}$ of S, each of size $\approx d^{3/2}$, so that the majority vote h over the m'' functions they define correctly classifies S.

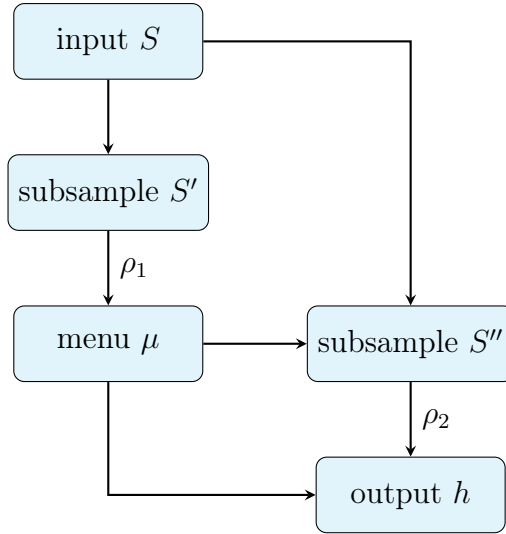**Output:** The function $h : \mathcal{X} \to \mathcal{Y}$.

Figure 5. The outline of the algorithm. The m' subsamples in step one are found using the OIG algorithm of the class $\mathcal{H}$. The m'' subsamples in step two are found using the OIG algorithm of the class $\mathcal{H}$ and the menu μ.

Proof of Theorem 36. The high-level outline is presented in Figure 5. Let S be an $\mathcal{H}$-realizable sample of size n. Lemma 39 tells us that there is a reconstruction ρ_1 that produces p-menus, and a sequence S' of r_1 examples from S such that S is μ-realizable where $\mu = \rho_1(S')$. Lemma 40 applied to $\mathcal{H}$ and μ implies that there is a reconstruction ρ_2, and a sequence S'' of r_2 examples from S such that $\rho_2(S'')$ correctly classifies the entire sample S. The composition of the two schemes is an $n \to r_1 + r_2$ sample compression scheme for $\mathcal{H}$. $\square$

The List Compression Scheme.

Proof of Lemma 39. We begin by describing the reconstruction function ρ_1. Let $\ell = \lfloor \frac{d_{DS}+t+1}{t+1} \log(2n) \rfloor$. Given an $\mathcal{H}$-realizable sample S' of size $r_1 = (d_{DS} + t)\ell$,

partition it into ℓ contiguous subsequences $S_1', \ldots, S_\ell'$, each of size $d_{DS} + t$. Define $\mu = \rho(S')$ as

$$\mu(x) = \bigcup_{j=1}^{\ell} \mu_j(x),$$

where $\mu_j(x) = \mathcal{L}_\mathcal{H}(S_j')$ is the $\binom{d_{DS}+t}{t}$-menu outputted by Algorithm 2 on input sample S_j'. The menu μ has list-size $p \leq \binom{d_{DS}+t+1}{t+1} \log(2n)$.

It remains to show that there is S' so that the reconstruction on S' achieves its goal. The sample S' is chosen via the probabilistic method. Let U denote the uniform distribution over the n examples in S and let $\alpha = \frac{t+1}{d_{DS}+t+1}$. By Proposition 32 applied to the distribution U, for a random sample $S_1' \sim U^{d_{DS}+t}$, in expectation at least αn of the examples (x_i, y_i) in S satisfy

$$y_i \in \mu_1(x_i),$$

where $\mu_1 = \mathcal{L}_{\mathcal{H},t}(S_1')$. In particular, there exists S_1' for which the above holds. Remove from S all examples (x_i, y_i) such that $y_i \in \mu_1(x_i)$ and repeat the same reasoning on the remaining sample. This way at each step j we find a sample S_j' and a menu $\mu_j = \mathcal{L}_{\mathcal{H},t}(S_j')$ that covers at least an α-fraction of the remaining examples. After ℓ steps, all examples in S are covered because $(1-\alpha)^\ell n < 1$. Setting S' to be the concatenation of $S_1', \ldots, S_\ell'$ finishes the proof. $\square$

Learning From a Menu.

Proof of Lemma 40. We begin by describing the reconstruction function ρ_2. Let $\ell = \lfloor 8 \log(2n) \rfloor$ and $m = \lceil 100 d_N \log(p) \rceil$. Given a sequence of $r_2 = m\ell$ examples S' that are realizable by $\mathcal{H}$ and μ, partition it into ℓ contiguous sub-sequences $S_1', \ldots, S_\ell'$, each of size m. Define $h(x) = \rho_2(S')$ as

$$h(x) = \mathsf{plurality}\big(h_1(x), \ldots, h_\ell(x)\big)$$

where $h_j = \mathcal{A}_{\mathcal{H},\mu}(S_j')$ is the hypothesis outputted by Algorithm 3 on input sample S_j', and $\mathsf{plurality}(y_1, \ldots, y_\ell)$ is the label that appears most frequently (breaking ties arbitrarily).

It remains to explain how to choose S' from a sample S that is realizable by $\mathcal{H}$ and μ. The existence of S' follows from the probabilistic method. This time we also rely on von Neumann's minimax theorem [von Neumann, 1928].

We first claim that there exists a distribution $\mathcal{P}$ over sequences T of size m with elements from S such that for every example (x, y) in S,

(6) $$\Pr_{T \sim \mathcal{P}}\big[h_T(x) \neq y\big] \leq \frac{1}{4},$$

where $h_T = \mathcal{A}_{\mathcal{H},\mu}(T)$. Consider a zero-sum game between two players Minnie and Max. Max's pure strategies are examples (x, y) in S. Minnie's pure strategies are sequences T of m elements from S. The payoff matrix L is defined by $L_{(x,y),T} = 1_{h_T(x) \neq y}$. Let $\mathcal{Q}$

be a mixed strategy of Max. Namely, $\mathcal{Q}$ is a distribution over examples in S. The distribution $\mathcal{Q}$ is realizable by both $\mathcal{H}$ and μ. Proposition 34 implies

$$\Pr_{(T,(x,y))\sim\mathcal{Q}^{m+1}}\left[h_T(x) \neq y\right] \leq \frac{20d_N\log(p)}{m} \leq \frac{1}{4}.$$

In words, for every mixed strategy of Max, there is a strategy for Minnie with cost at most $\frac{1}{4}$. By the minimax theorem, there is a mixed strategy for Minnie that guarantees cost at most $\frac{1}{4}$ for every strategy of Max. This mixed strategy is the required $\mathcal{P}$.

The existence of S' can finally be proved. Let $S'_1, \ldots, S'_\ell$ be i.i.d. samples from $\mathcal{P}$. Standard concentration of measure implies that for each $(x,y) \in S$,

$$\Pr\left[\frac{1}{\ell}\sum_{j=1}^{\ell} 1_{h_{S'_j}(x)\neq y} \geq \frac{1}{2}\right] \leq \exp\left(-\frac{\ell}{8}\right) < \frac{1}{n}.$$

The union bound implies that with positive probability, for every (x,y) in S we have $\frac{1}{\ell}\sum_{j=1}^{\ell} 1_{h_{S'_j}(x)\neq y} < \frac{1}{2}$. In particular, there exist $S'_1, \ldots, S'_\ell$ such that the plurality vote over the $h_{S'_1}, \ldots, h_{S'_\ell}$ correctly classifies all of S. The concatenation is the required S'. $\qquad\square$

4.5. Wrapping-up.

Proof of Theorem 1. Theorem 36 with $t = \lceil d^{1/2} \rceil$ states the existence of an $n \to r$ sample compression scheme for $\mathcal{H}$ where $r \leq O(d^{3/2}\log(n))$. The analysis of the sample compression scheme relies on previous results on multiclass compression. Theorems 3.1 and 3.3 in [David, Moran, and Yehudayoff, 2016] imply that if $\mathcal{H}$ admits an $n \to r$ sample compression scheme, then the same compression scheme is a learning rule A^{real} with the following guarantee. For every $\mathcal{H}$-realizable distribution $\mathcal{D}$, every $\delta > 0$ and every integer $n > 0$, with probability at least $1-\delta$ over $S \sim \mathcal{D}^n$, the output hypothesis $h = A^{real}(S)$ satisfies

$$(7) \qquad \Pr_{(x,y)\sim\mathcal{D}}\left[h(x) \neq y\right] \leq O\left(\frac{r\log\left(\frac{n}{r}\right) + \log(1/\delta)}{n}\right).$$

In the agnostic case, they prove the existence of a related learning rule A^{agn} with the following guarantee. For every distribution $\mathcal{D}$, every $\delta > 0$, and every integer $n > 0$, with probability at least $1-\delta$ over sampling $S \sim \mathcal{D}^n$, the output hypothesis $h = A^{agn}(S)$ satisfies

$$(8) \qquad \Pr_{(x,y)\sim\mathcal{D}}\left[h(x) \neq y\right] \leq L_\mathcal{D}(\mathcal{H}) + O\left(\sqrt{\frac{r\log\left(\frac{n}{r}\right) + \log(1/\delta)}{n}}\right). \qquad\square$$

5. Learnability $\not\equiv$ Finite Natarajan Dimension

The goal of this section is to prove that the Natarajan dimension does not characterize PAC learnability (Theorems B and 2). That is, to construct a concept class $\mathcal{H}$ that has Natarajan dimension 1 but DS dimension ∞.

5.1. **Outline.** The class $\mathcal{H}$ lives between two opposing conditions. On one hand, there should be no non-trivial copy of the Boolean cube inside $\mathcal{H}$. On the other hand, it should contain pseudo-cubes of arbitrary large dimension. Pseudo-cubes of large dimension imply that learning $\mathcal{H}$ is difficult. No copies of the Boolean cube indicates that "locally $\mathcal{H}$ looks like it is easy to learn". The barrier to learning $\mathcal{H}$ is not local but global. An analogy is a graph of large girth and large chromatic number; locally the graph is 2-colorable, and the coloring-difficulty stems from a global obstacle.

Our goal is, essentially, to find pseudo-cubes of arbitrary large dimension that do not contain any non-trivial copy of the Boolean cube. The proof begins by translating the problem from the realm of concept classes to the realm of simplicial complexes (see Section 5.2). We show that any concept class can be identified with a colorful simplicial complex (and vice versa).

What about the pseudo-cube condition and the Natarajan dimension in the realm of simplicial complexes? The pseudo-cube conditions turns out to be quite natural; it is reminiscent of the notion of a *pseudo-manifold*. The Natarajan dimension 1 condition is almost identical to the *flag-no-square* condition; this condition was studied in many works as a local combinatorial criteria for hyperbolicity.

As the abstract of [Januszkiewicz and Świątkowski, 2003] indicates, simplicial complexes in the spirit we need were conjectured not to exist (by Moussong), or at least to require difficult number theory (by Gromov). However, Januszkiewicz and Świątkowski [2003] built a simplicial complex that exactly meets our needs (see Section 5.3).

The difficulty of the construction explains the fact that Natarajan's question was open for so many years. For example, for $d = 2$, the smallest concept class with Natarajan dimension 1 we know of has size 6; see Figure 3. For $d = 3$, the size grows to 54; see Figure 2. For $d = 4$, the size jumps to 118,098. This large complex is not the complex suggested in [Januszkiewicz and Świątkowski, 2003]. The high-level structure of the construction is similar, but the complex we found is smaller. We found the construction and verified it with a computer (using [GAP, 2021]). See Section 9 of [Januszkiewicz and Świątkowski, 2003] for more details on the "complexity" of their construction.

5.2. **Pseudo-cubes and Simplicial Complexes.** We begin with a brief introduction to simplicial complexes. Simplicial complexes are combinatorial abstractions of triangulations of topological spaces. A family C of finite subsets of a set V is called a simplicial complex if it is downward closed. That is, for every $f \in C$, if $g \subset f$ then $g \in C$. A member of C is called a simplex or a face. The *dimension* of a face $f \in C$ is defined to be $\dim(f) = |f| - 1$ and the dimension of the complex C is $\dim(C) = \max_{f \in C} \dim(f)$. A simplicial complex is called *pure* if all of its maximal faces have the same dimension. The 1-*skeleton* of a simplicial complex C is a graph whose vertices are the elements of V and whose edges are all the 1-dimensional faces of C. Every face in C thus corresponds to a clique in its 1-skeleton.

We also need our complexes to be properly colored. A *proper coloring* of a complex C is a proper coloring of the 1-skeleton of C with $\dim(C) + 1$ colors. That is, it is an

assignment $r : V \to [\dim(C) + 1]$ such that $r(u) \neq r(v)$ for every distinct u, v so that $\{u, v\} \in C$.

Our first goal in this subsection is to express the notion "pseudo-cube" in the language of simplicial complexes. This is captured by the following definitions. We say that a complex C satisfies *replacement* if for every simplex $f \in C$ and for every vertex $v \in f$ there exists a vertex $u \neq v$ such that $(f \setminus \{v\}) \cup \{u\} \in C$.

Definition 41 (Good Complex). A simplicial complex C is *good* if it is finite, pure, has a proper coloring, and satisfies replacement.

The following proposition summarizes the equivalence between pseudo-cubes and good simplicial complexes. Figure 2 may help in digesting this equivalence.

Proposition 42 (Concept Classes $\equiv$ Good Complexes). *For every d-dimensional good complex C and a proper coloring r of C, there is a $(d + 1)$-dimensional pseudo-cube $B = B(C, r)$. Conversely, for every d-dimensional pseudo-cube B, there is a $(d - 1)$-dimensional good simplicial complex $C = C(B)$.*

Remark. *The pseudo-cube $B(C, r)$ and the complex $C(B)$ are explicitly constructed in the proof.*

Proof. Good complex $\implies$ pseudo-cube. Let C be a good d-dimensional complex over V and let $r : V \to [d + 1]$ be a proper coloring of C. We define a $(d + 1)$-dimensional pseudo-cube $B \subseteq V^{d+1}$ as follows. Each face $f \in C$ of maximum size $|f| = d + 1$ defines the word $(v_1, \ldots, v_{d+1}) \in B$ such that for each $i \in [d + 1]$, the vertex $v_i \in f$ is the unique vertex in f with color $r(v_i) = i$. It remains to verify that B is a pseudo-cube. The class B is finite because C is finite. For every $(v_1, \ldots, v_{d+1}) \in B$ and $i \in [d + 1]$, the following holds. The set $\{v_1, \ldots v_{d+1}\}$ is a face in C. By the replacement property, there exists $u \neq v_i$ such that $\{v_1, \ldots v_{i-1}, u, v_{i+1}, \ldots v_{d+1}\}$ is a face in C. Because r is a proper coloring, it must be that $r(u) = i$. So, $(v_1, \ldots v_{i-1}, u, v_{i+1}, \ldots v_{d+1})$ is in B as well.

Pseudo-cube $\implies$ good complex. Given a d-dimensional pseudo-cube $B \subseteq \mathcal{Y}^d$, define a simplicial complex C as follows. The vertex-set of C consists of all $(y, i) \in \mathcal{Y} \times [d]$ so that y appears as the i'th letter of some word in B. Each $(y_1, \ldots, y_d) \in B$ defines the maximal face $\{(y_i, i) : i \in [d]\}$ of C. It remains to verify that C is good. The complex C is finite because B is finite. The complex C is pure and all its maximal faces have dimension $d - 1$. Consider the coloring $r : V \to [d]$ defined by $r((y, i)) = i$. It is a proper coloring because all faces contain at most one vertex of each color. Because B is a pseudo-cube, for each $i \in [d]$ there exists a word $(y_1, \ldots, y_{i-1}, y'_i, y_{i+1}, \ldots, y_d) \in B$ with $y'_i \neq y_i$. In other words, C satisfies the replacement property. $\square$

The remaining of this section is about translating the Natarjan dimension condition to the language of simplicial complexes. A *square $v_0 \to v_1 \to v_2 \to v_3 \to v_0$* in a simplicial complex C is a sequence of four distinct vertices that form a cycle of length four in the 1-skeleton of C.

Proposition 43 (Natarajan Dimension for Colored Complex). *Let C be a d-dimensional good complex and let r be a proper coloring of C. Let $B = B(C, r)$ be the pseudo-cube that is defined by C, r. The two following properties are equivalent:*

(1) There exists a square $v_0 \to v_1 \to v_2 \to v_3 \to v_0$ in C such that $r(v_0) = r(v_2)$ and $r(v_1) = r(v_3)$.

(2) The Natarajan dimension of B is at least 2.

Proof. First, assume that there exists a square $v_0 \to v_1 \to v_2 \to v_3 \to v_0$ in C such that $i := r(v_0) = r(v_2)$ and $j := r(v_1) = r(v_3)$. Because r is proper, we know $i \neq j$. Without loss of generality, assume $i < j$. It suffices to show that $B|_{\{i,j\}}$ contains all 4 patterns

$$(v_0, v_1), (v_2, v_1), (v_0, v_3), (v_2, v_3).$$

This follows because each of the patterns (v_a, v_b) for $a \in \{0, 2\}$ and $b \in \{1, 3\}$ corresponds to an edge $\{v_a, v_b\}$ in C. This edge is contained in a maximal d-dimensional face $f_{a,b} \in C$. By the definition of B, the face $f_{a,b}$ corresponds to a word in B which realizes the pattern (v_a, v_b) on $\{i, j\}$.

In the other direction, assume that the Natarajan dimension of B is at least 2. Let $i < j$ be a pair of coordinates such that there exist labels v_0, v_1, v_2, v_3 such that the following 4 patterns belong to $B|_{\{i,j\}}$:

$$(v_0, v_1), (v_2, v_1), (v_0, v_3), (v_2, v_3).$$

The definition of B thus implies that $r(v_0) = r(v_2) = i$, that $r(v_1) = r(v_3) = j$, and that $\{v_0, v_1\}, \{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_0\} \in C$. So, $v_0 \to v_1 \to v_2 \to v_3 \to v_0$ is the desired square. $\square$

Typically, simplicial complexes are not colorful. So, it is helpful to have a version of Proposition 43 that does not require a proper coloring. An *empty square* in a simplicial complex C is a square $v_0 \to v_1 \to v_2 \to v_3 \to v_0$ so that both $\{v_0, v_2\}$ and $\{v_1, v_3\}$ are not edges in the 1-skeleton of C. In other words, an empty square is a square so that the induced graph on its vertices is the same square (somewhat confusingly this is also known as a *full* square in some contexts).

Corollary 44 (Natarajan Dimension for Complex). *If there are no empty squares in a good simplicial complex C of dimension d then for every proper coloring r of C, the Natarajan dimension of $B(C, r)$ is at most 1.*

Proof. By Proposition 43, if the Natarajan dimension of $B(C, r)$ is at least 2, then there is a square $v_0 \to v_1 \to v_2 \to v_3 \to v_0$ in C such that $r(v_0) = r(v_2)$ and $r(v_1) = r(v_3)$. Because r is a proper coloring, the square must be empty. $\square$

5.3. **The Simplicial Complex.** The goal of this section is to state the construction by Januszkiewicz and Świątkowski [2003] of the simplicial complexes we need.

How can we build a complex C, that is pure, has a proper coloring and satisfies replacement? This is quite easy, and we shall return to it below. The reason is that we did not insist that C is finite. The challenge is to have all these properties in a finite object.

A baby version of this difficulty appears already in graph theory. It is fairly easy to build an infinite regular tree, but constructing finite regular graphs is more challenging. Group theory provides a fundamental and powerful mechanism to "fold" the infinite tree to a finite regular graph. If the infinite tree is thought of as a Cayley graph of some group F, and N is a normal subgroup of F of finite index, then the "modulo N" operation allows to fold the tree to a finite graph. Many useful constructions of finite graphs are obtained via this mechanism.

Coming back to an infinite complex that is pure, properly colored and satisfies replacement, we can simply start with a face of dimension d, connect it to $d + 1$ new faces by adding new vertices, and keep going indefinitely. This construction corresponds to an infinite regular tree (see also Example 8). It is easy to build, but utterly useless for us. The real difficulty is to "fold" it to be finite. What does "fold" even mean? The solution is again algebraic, but it uses the more abstract language of coset complexes.

Let F be a group (finite or infinite). A coset of a subgroup $H \leq F$ is a set of the form $gH = \{gh : h \in H\}$. The *coset complex* defined by subgroups $H_1, \ldots, H_d \leq F$ is the simplicial complex $C = C_F(H_1, \ldots, H_d)$ that is defined as follows. The vertices of C are the cosets of the groups $H_1, \ldots, H_d$, and a set of cosets σ is a simplex in C if and only if the intersection of all cosets in σ is non-empty: $\sigma \in C \iff \bigcap_{L \in \sigma} L \neq \emptyset$. Stated differently, the complex is the *nerve* of the set of all cosets.

The following theorem states the existence of the coset complexes we need.

Theorem 45 (Januszkiewicz and Świątkowski [2003]). *For every integer $d > 1$, there exists a finite group F, and d subgroups $H_1, \ldots, H_d \leq F$ such that the following hold:*

(1) For every $i \in [d]$, we have $(\cap_{j \neq i} H_j) \setminus H_i \neq \emptyset$.

(2) The coset complex $C_F(H_1, \ldots, H_d)$ does not contain empty squares.

Theorem 45 is a consequence of a deep construction by Januszkiewicz and Świątkowski [2003] which combines tools and ideas from algebra and topology that are beyond the scope of our work. In Appendix A we formally derive Theorem 45 using results stated in [Januszkiewicz and Świątkowski, 2003]. It is rather a formality, because all ideas are already in that paper, but the exact result we need, unfortunately, is not explicitly stated. This derivation is *not* self-contained and uses concepts that are defined in [Januszkiewicz and Świątkowski, 2003].

Nevertheless, let us provide a simplified and high-level description of their approach. The proof of Theorem 45 is by induction on d. The group F is generated by d involutions $Z = \{z_1, \ldots, z_d\}$. The involution condition $z^2 = 1$ corresponds to having a *single* neighbor in each coordinate in the corresponding concept class. This is a strong version of the pseudo-cube condition. The subgroup H_i is generated by the $d - 1$ involutions $Z \setminus \{z_i\}$.

How can we apply induction? Imagine that each H_i plays the role of F. So, H_i is finite group and it has $d - 1$ subgroups that yield a pseudo-cube of dimension $d - 1$. These $d - 1$ subgroups of H_i are generated by $Z \setminus \{z_i, z_j\}$ for each $j \in [d] \setminus \{i\}$. In other words, we have d constructions for dimension $d - 1$ that are somehow "glued together using algebra".

All these finite groups naturally live inside one big group $\mathcal{F}$. This group $\mathcal{F}$ is the free product of the H_i's modulo the "obvious" relations that are there because e.g. z_1 is the same inside H_2 and inside H_3 (i.e., $\mathcal{F}$ is the free product with amalgamation). The group $\mathcal{F}$, however, is infinite. We obtain an infinite simplicial complex $C_{\mathcal{F}} = C_{\mathcal{F}}(H_1, \ldots, H_d)$. Again, an infinite object that we need to make finite. To do so, we need to carefully identify a normal subgroup of $\mathcal{F}$ of finite index, so that after we divide by it "everything still works". Where should we look for this magical subgroup?

One key idea is to replace the no empty square condition by a stronger algebraic condition that the groups H_i and their subgroups satisfy. This condition is called *extra retractibility*. It states the existence of certain homomorphisms between various subgroups of the H_i's. Isolating the extra retractibility condition is a major and difficult step. Even verifying that extra retractibility implies the no empty square condition is not at all trivial. But the real question is: what did we gain from this algebraic move?

The gain is that we can try to prove that the infinite complex $C_{\mathcal{F}}$ is again extra retractible. This is not the end of the story, but it is a start. Proving that $C_{\mathcal{F}}$ is extra retractible uses topology as well as the "universality" of free products. Topological properties of the complex $C_{\mathcal{F}}$ (e.g., it is connected and simply connected) allow to represent it in a "non-obvious" way. The inductive hypothesis shows that extra retractibility "locally holds" on $C_{\mathcal{F}}$. Universality now implies that it also holds globally, due to the topological properties.

The infinite complex is extra retractible. So what? The algebraic nature of extra retractibility serves as a guide in the search for the magical normal subgroup. Extra retractibility states the existence of certain homomorphisms (to finite groups). We can identify a finite index subgroup $\mathcal{K}$ of $\mathcal{F}$ that is contained in all kernels of these homomorphisms. Taking the normal core $\mathcal{N} = \bigcap_{g \in \mathcal{F}} g\mathcal{K}g^{-1}$ yields a normal subgroup of finite index (!) that is contained in "all kernels". This latter property of $\mathcal{N}$ tells us that even after we divide by it, extra retractibility still holds. We can finally choose $F = \mathcal{F}/\mathcal{N}$ and complete the inductive step.

Let us return to the main goal of this section, deducing the needed concept class from the construction of Januszkiewicz and Świątkowski [2003].

Proposition 46 (There is a Good Complex with No Empty Squares). *Let F and $H_1, \ldots, H_d$ be as in Theorem 45. The coset complex $C = C_F(H_1, \ldots, H_d)$ has dimension $d - 1$, is good and has no empty squares.*

Proof. Because F is finite, C is finite as well.

Let us prove that C is pure of dimension $d-1$. If σ is a face, then there is $g \in \bigcap_{L \in \sigma} L$. Because every two distinct cosets of the same subgroup are disjoint, the face σ is of the form $\{gH_i : i \in I\}$ for some $I \subseteq [d]$. The face σ is contained in the maximal face $\{gH_i : i \in [d]\}$ which has dimension $d - 1$.

There is a straightforward proper coloring of C with d colors. Color each vertex of the form gH_i by the color i. This is a coloring because a subset of a group can be a coset of at most one subgroup. This is a proper coloring because two distinct cosets of the same subgroup are disjoint.

Finally, we prove that C satisfies replacement. Let $\sigma \in C$. As above, we can write $\sigma = \{gH_i : i \in I\}$ for some $I \subseteq [d]$. Consider a vertex gH_k inside σ. By construction of C, let $z \in (\cap_{j \neq k} H_j) \setminus H_k$. For every $j \neq k$, we have $gzH_j = gH_j$. Because $z \notin H_k$, we can conclude $gzH_k \neq gH_k$. It follows that $(\sigma \setminus \{gH_k\}) \cup \{gzH_k\} = \{gzH_i : i \in I\}$ is also a face.

$\square$

5.4. **Wrapping up.**

Proof of Theorem 2. Theorem 45, Proposition 46 and Corollary 44 imply that for every d, there exists a d-dimensional pseudo-cube $B_d \subseteq Y_d^{X_d}$ where $|X_d| = d$ with Natarajan dimension 1. We may assume that the label-sets Y_d's are pairwise disjoint, and that the domains X_d's are pairwise disjoint.

Construct the "disjoint union" of all these classes. Let $\mathcal{X} = \bigcup_d X_d$. Each $h \in B_d$ is a partial map on $\mathcal{X}$ because it is defined only on X_d. Let $\star$ be a new label such that $\star \notin \bigcup_d Y_d$. Extend each such h by setting it to be $\star$ outside X_d. Denote by H_d the collection of extensions of maps in B_d. Finally, let

$$\mathcal{H} = \bigcup_d H_d.$$

By construction, the DS dimension of $\mathcal{H}$ is at least d for every integer d because it contains a copy of B_d.

It remains to prove that the Natarajan dimension of $\mathcal{H}$ is 1. It is at least 1 because $|\mathcal{H}| \geq 2$. The last thing to verify is that the Natarajan dimension is less than 2. Let $x_1, x_2 \in \mathcal{X}$ be a pair of distinct points and assume towards contradiction that $\{x_1, x_2\}$ is N-shattered by $\mathcal{H}$. If $x_1 \in X_{d_1}$ and $x_2 \in X_{d_2}$ for $d_1 \neq d_2$, then every function $h \in \mathcal{H}$ satisfies that $\star \in \{h(x_1), h(x_2)\}$ and therefore $\{x_1, x_2\}$ is not N-Shattered by $\mathcal{H}$. The last remaining case is that $x_1, x_2 \in X_d$ for the same d. In this case, every function $h \in \mathcal{H} \setminus H_d$ satisfies that $h(x_1) = h(x_2) = \star$ and there is no function $h \in \mathcal{H}$ such that $\{h(x_1), h(x_2)\} = \{\star, y\}$ for $y \neq \star$. It follows that $\{x_1, x_2\}$ must be N-shattered by H_d which is a contradiction because the Natarajan dimension of B_d is 1. $\square$

6. Acknowledgements

References

Noga Alon, Steve Hanneke, Ron Holzman, and Shay Moran. A theory of PAC learnability of partial concept classes. *arXiv:2107.08444*, 2021.

Shai Ben-David, Nicolo Cesabianchi, David Haussler, and Philip M Long. Characterizations of learnability for classes of {0,...,n}-valued functions. *Journal of Computer and System Sciences*, 50(1):74–86, 1995.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4): 929–965, 1989.

Nataly Brukhim, Elad Hazan, Shay Moran, and Robert E. Schapire. Multiclass boosting and the cost of weak learning. In *NIPS*, 2021.

Amit Daniely and Shai Shalev-Shwartz. Optimal learners for multiclass problems. In *COLT*, pages 287–316, 2014.

Amit Daniely, Sivan Sabato, and Shai Shalev-Shwartz. Multiclass learning approaches: A theoretical comparison with implications. In *NIPS*, pages 494–502, 2012.

Amit Daniely, Sivan Sabato, Shai Ben-David, and Shai Shalev-Shwartz. Multiclass learnability and the ERM principle. *The Journal of Machine Learning Research*, 16: 2377–2404, 2015a.

Amit Daniely, Michael Schapira, and Gal Shahaf. Inapproximability of truthful mechanisms via generalizations of the vc dimension. In *STOC*, pages 401–408, 2015b.

Ofir David, Shay Moran, and Amir Yehudayoff. Supervised learning through the lens of compression. In *NIPS*, pages 2784–2792, 2016.

GAP. The gap group, gap – groups, algorithms, and programming. 2021.

David Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inf. Comput.*, 100(1):78–150, 1992.

David Haussler. Sphere packing numbers for subsets of the boolean n-cube with bounded Vapnik-Chervonenkis dimension. *Journal of Combinatorial Theory, Series A*, 69(2):217–232, 1995.

David Haussler and Philip M. Long. A generalization of Sauer's lemma. *J. Comb. Theory, Ser. A*, 71(2):219–240, 1995.

David Haussler, Nick Littlestone, and Manfred K Warmuth. Predicting {0, 1}-functions on randomly drawn points. *Information and Computation*, 115(2):248–292, 1994.

Tadeusz Januszkiewicz and Jacek Świątkowski. Hyperbolic coxeter groups of large dimension. *Commentarii Mathematici Helvetici*, 78(3):555–583, 2003.

Nick Littlestone and Manfred Warmuth. Relating data compression and learnability. *Unpublished manuscript*, 1986.

Balas K. Natarajan. Some results on learning. *Unpublished manuscript*, 1988.

Balas K. Natarajan. On learning sets and functions. *Machine Learning*, 4(1):67–97, 1989.

Balas K. Natarajan and Prasad Tadepalli. Two new frameworks for learning. In *ICML*, pages 402–415, 1988.

David Pollard. Empirical processes: Theory and applications. *NSF-CBMS Regional Conference Series in Probability and Statistics*, 2:1–86, 1990.

Benjamin Rubinstein, Peter Bartlett, and J Hyam Rubinstein. Shifting, one-inclusion mistake bounds and tight multiclass expected risk bounds. In *NIPS*, pages 1193–1200, 2006.

Robert E Schapire and Yoav Freund. *Boosting: Foundations and algorithms*. Cambridge University Press, 2012.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

Leslie G. Valiant. A theory of the learnable. In *STOC*, pages 436–445, 1984.

Vladimir Vapnik. Inductive principles of the search for empirical dependences (methods based on weak convergence of probability measures). In *COLT*, pages 3–21, 1989.

Vladimir Vapnik and Alexey Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Proc. USSR Acad. Sci.*, 1968.

Vladimir Vapnik and Alexey Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974.

John von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100 (1):295–320, 1928.

Appendix A. The Simplicial Complex

This parts explains how to deduce Theorem 45 from the results that are stated in [Januszkiewicz and Świątkowski, 2003]. This appendix uses definitions and theorems from that paper. The key idea is to construct a *development* L_d of an *extra retractible* complex of finite groups on the poset $\{0, 1\}^{[d]}$; see Section 6 in [Januszkiewicz and Świątkowski, 2003]. The complex of groups consists of a monotone mapping from subsets of $[d]$ to finite groups. Each $A \subseteq [d]$ is assigned a finite group F_A such that $F_A \leq F_B$ whenever $A \subseteq B$. Extra retractibility further implies that

$$\forall A, B \subseteq [d] \quad F_A \cap F_B = F_{A \cap B} \tag{9}$$

$$F_\emptyset = \{1\} \tag{10}$$

Equation 9 follows from Propositions 3.2 and 4.1 in [Januszkiewicz and Świątkowski, 2003]. Equation 10 follows because an extra retractible complex of groups is *reduced*; see Definitions 4.4 and 5.8 in Januszkiewicz and Świątkowski [2003].

In the construction, the groups are generated by involutions. For each $i \in [d]$ the group $F_{\{i\}}$ is $\{1, z_i\}$ where $z_i^2 = 1$. To prove Theorem 45, we set

$$F = F_{[d]} \text{ and } H_i = F_{[d] \setminus \{i\}}.$$

With these choices, the development L_d is isomorphic to the coset complex $C = C(H_1, \ldots, H_d)$.

To justify Item 1, we shall prove that $z_i \in (\cap_{j \neq i} H_j) \setminus H_i$. Because $z_i \in F_{\{i\}}$ and by Equation 9,

$$F_{\{i\}} \cap H_i = F_{\{i\}} \cap F_{[d] \setminus \{i\}} = F_\emptyset = \{1\} \implies z_i \notin H_i.$$

On the other hand, for $j \neq i$,

$$z_i \in F_{\{i\}} \subseteq F_{[d] \setminus \{j\}} = H_j.$$

Finally, Proposition 5.12 in [Januszkiewicz and Świątkowski, 2003] asserts that C contains no empty squares.

Appendix B. Orientations for infinite graphs

Here we complete the proof of Lemma 13 for infinite graphs. Let $\mathcal{G} = (V, E)$ be the one-inclusion graph of $\mathcal{H}$. Let $\mathcal{Z}$ be the set of pairs $z = (v, e) \in V \times E$ so that $v \in e$. Let $\mathcal{K} = \{0, 1\}^{\mathcal{Z}}$. Tychonoff's theorem says that $\mathcal{K}$ is compact with respect to the product topology.

An orientation corresponds to choosing for each $e \in E$ a single $v \in V$. In other words, each orientation can be thought of as an element of $\mathcal{K}$, where $\kappa_{(v,e)} = 1$ means that e is oriented towards v, and $\kappa_{(v,e)} = 0$ means that e is not oriented towards v.

For every $v \in V$, let $A_v \subseteq \mathcal{K}$ be the set of all $\kappa \in \mathcal{K}$ so that there are at most d edges $e \ni v$ so that $\kappa_{(v,e)} = 0$. For $j \in [d+1]$, let $B_{v,j} \subseteq \mathcal{K}$ be the set of all $\kappa \in \mathcal{K}$ so that for the edge $e = e_j$ that is in the j'th direction of v, there is at most one u so that $\kappa_{(u,e)} = 1$.

The complement of the set A_v is open because it is $\{\kappa \in \mathcal{K} : \forall i \in [d+1] \; \kappa_{(v,e_i)} = 0\}$, where e_i is the edge in the i'th direction of v. The complement of the set $B_{v,j}$ is open because it is the union over all sets $\{w_1, w_2\}$ of two vertices that are contained in $e = e_j$ of $\{\kappa \in \mathcal{K} : \kappa_{(w_1,e)} = \kappa_{(w_2,e)} = 1\}$. The set $\Sigma_v = A_v \cap \bigcap_{j \in [d+1]} B_{v,j}$ is hence closed.

We now claim that for every finite $U \subset V$ the set $\bigcap_{v \in U} \Sigma_v$ is non-empty. The finite hyper-graph $\mathcal{G}_U$ that $\mathcal{G}(\mathcal{H})$ induces on U has an orientation σ with maximum out-degree at most d. The orientation σ defines an element κ in $\bigcap_{v \in U} \Sigma_v$ as follows. There are two types of edges: edges in $\mathcal{G}(\mathcal{H})$ that correspond to edges in $\mathcal{G}_U$, and edges that "disappear" with the projection to U. The former type of edges are oriented in κ exactly as in σ. The latter type of edges are edges e that contain at most one vertex in U and possibly other vertices outside U. If e contains a single vertex u from U then orient e towards u, and otherwise orient e to an arbitrary element (not in U).

Compactness now implies that the intersection of all the Σ_v's is non-empty. In particular, there is κ^* in $\bigcap_{v \in V} \Sigma_v$. This κ^* can be thought of as a partial orientation, because every edge e contains some vertex (and so there is at most one $u \in e$ so that $\kappa^*_{(u,e)} = 1$). For each $v \in V$, there are at most d edges $e \ni v$ so that $\kappa^*_{(v,e)} = 0$. Complete κ^* to a full orientation σ^* by arbitrarily orienting all edges that are not oriented in κ^*. The out-degree of σ^* is still at most d, because the final move from κ^* to σ^* does not increase out-degrees.

DEPARTMENT OF COMPUTER SCIENCE, PRINCETON
E-mail address: nbrukhim@princeton.edu

DEPARTMENT OF MATHEMATICS, TECHNION-IIT
E-mail address: daniel.carmon91@gmail.com

DEPARTMENT OF COMPUTER SCIENCE, WEIZMANN INSTITUTE
E-mail address: irit.dinur@weizmann.ac.il

DEPARTMENT OF MATHEMATICS, TECHNION–IIT, AND GOOGLE RESEARCH
E-mail address: smoran@technion.ac.il

DEPARTMENT OF MATHEMATICS, TECHNION–IIT
E-mail address: amir.yehudayoff@gmail.com