Proceedings of Machine Learning Research vol 117:1–15, 2020 31st International Conference on Algorithmic Learning Theory

Distribution Free Learning with Local Queries

Galit Bary-Weisberg BARYGALIT@GMAIL.COM
Mobileye inc.

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

Shai Shalev-Shwartz SHAI.SHWARTZ@GMAIL.COM
School of Computer Science and Engineering, The Hebrew University, Jerusalem

Editors: Aryeh Kontorovich and Gergely Neu

Abstract

The model of learning with *local membership queries* interpolates between the PAC model and the membership queries model by allowing the learner to query the label of any example that is similar to an example in the training set. This model, recently proposed and studied by Awasthi et al. (2012), aims to facilitate practical use of membership queries.

We continue this line of work, proving both positive and negative results in the *distribution free* setting. We restrict to the boolean cube $\{-1,1\}^n$, and say that a query is q-local if it is of a hamming distance $\leq q$ from some training example. On the positive side, we show that 1-local queries already give an additional strength, and allow to learn a certain type of DNF formulas, that are not learnable without queries, assuming that learning decision trees is hard. On the negative side, we show that even $\left(n^{0.99}\right)$-local queries cannot help to learn various classes including Automata, DNFs and more. Likewise, q-local queries for any constant q cannot help to learn Juntas, Decision Trees, Sparse Polynomials and more. Moreover, for these classes, an algorithm that uses $\left(\log^{0.99}(n)\right)$-local queries would lead to a breakthrough in the best known running times.

Keywords: PAC Learning, Local Membership Queries, DNFs

1. Introduction

A child typically learns to recognize a cat based on two types of input. The first is given by her parents, pointing at a cat and saying “Look, a cat!”. The second is given as a response to the child’s frequent question “What is that?”. These two types of input were the basis for the learning model originally suggested by Valiant (1984). Indeed, in Valiant’s model, the learner can randomly sample labelled examples from “nature”, but it can also make a *membership query (MQ)* for the label of *any* unseen example. Today, the acronym PAC stands for the restricted model in which MQ are forbidden, while the full model is called PAC+MQ. Much work has been done investigating the limits and the strengths of MQ. In particular, membership queries were proven stronger than the vanilla PAC model (Angluin, 1987; Blum and Rudich, 1992; Bshouty, 1995; Jackson, 1994). Yet, MQ are rarely used in practice. This is commonly attributed to the fact that MQ algorithms query very artificial examples, that are uninterpretable by humans (e.g., Baum and Lang (1992)).

Awasthi et al. (2012) suggested a solution to the problem of unnatural examples. They considered a mid-way model that allows algorithms to make only *local* queries, i.e., query examples that are close to examples from the sample set. Hopefully, examples which are similar to natural examples will appear natural to humans. Awasti et al. considered the case where the instance space is $\{-1,1\}^n$ and the distance between examples is the Hamming distance. They proved positive results on learning sparse polynomials with $O(\log(n))$-local queries under what they defined as locally smooth distributions[1]. They also proposed an algorithm that learns DNF formulas under the uniform distribution in quasi-polynomial time using $O(\log(n))$-local queries.

Our work follows Awasthi et al. and investigates local queries in the *distribution free* setting, in which no explicit assumptions are made on the underlying distribution, but only on the learned hypothesis. We prove both positive and negative results in this context:

- One of the strongest and most beautiful results in the MQ model shows that automata are learnable with membership queries Angluin (1987). We show that unfortunately, this is probably not the case with local queries. Concretely, we show that even $\left(n^{0.99}\right)$-local queries cannot help to learn automata. Namely, such an algorithm will imply a standard PAC algorithm for automata. As learning automata is hard under several assumptions Kearns and Valiant (1994); Daniely and Shalev-Shwatz (2016), our result suggests that it is hard to learn automata even with $\left(n^{0.99}\right)$-local queries.

- We prove a similar result for several additional classes. Namely, we show that $\left(n^{0.99}\right)$-local queries cannot help to learn DNFs, intersection of halfspaces, decision lists, depth-d circuits for any $d \geq 2$, and depth-d threshold circuits for any $d \geq 2$. Likewise, for any constant q, q-local queries cannot help to learn Juntas, Decision Trees, Sparse polynomials, and Sparse polynomial threshold functions. In fact, we show that even $\left(\log^{0.99}(n)\right)$-local queries are unlikely to lead to polynomial time algorithms. Namely, any algorithm that uses $\left(\log^{0.99}(n)\right)$-local queries will result with a PAC algorithm whose running time significantly improves the state of the art in these well studied problems.

- On the positive side we show that already 1-local queries are probably stronger than the vanilla PAC model. Concretely, we show that poly-sized DNF formulas in which every pair of terms contains two opposite literals, are learnable with 1-local queries. Furthermore, we show that without queries, learning this class is at least as hard as learning decision trees, that are conjectured to be hard to learn.

2. Previous Work

Membership Queries Several concept classes are known to be learnable only if membership queries are allowed: Deterministic Finite Automatons (Angluin, 1987), k-term DNF for $k = \frac{\log(n)}{\log(\log(n))}$ (Blum and Rudich, 1992), decision trees and k-almost monotone-DNF formulas (Bshouty, 1995), intersections of k-halfspaces (Baum, 1991) and DNF formulas under the uniform distribution (Jackson, 1994). The last result builds on Freund's boosting algorithm (Freund, 1995) and the Fourier-based technique for learning using membership queries due to (Kushilevitz and Mansour, 1993). We note that there are cases in which MQ do not help. E.g., in the case of learning DNF and CNF formulas (Angluin and Kharitonov, 1995), assuming that one way functions exist, and in the case of distribution free agnostic learning (Feldman, 2009).

1. A distribution is locally α-smooth for $\alpha \geq 1$ if its density function is $\log(\alpha)$-Lipschitz.

Local Membership Queries Awasthi et al. (2012) focused on $O(\log(n))$-local queries. They showed learnability of t-sparse polynomials under locally smooth distributions with $O(\log(n)+\log(t))$-local queries, and that DNF formulas are learnable under the uniform distribution in quasi-polynomial time ($n^{O(\log\log n)}$) using $O(\log(n))$-local queries. They also presented some results regarding the strength of local MQ. They proved that under standard cryptographic assumptions, $(r+1)$-local queries are more powerful than r-local queries (for every $1 \le r \le n-1$). They also showed that local queries do not always help. They showed that if a concept class is agnostically learnable under the uniform distribution using k-local queries (for constant k) then it is also agnostically learnable (under the uniform distribution) in the PAC model.

We remark that besides local queries, there were additional suggestions to solve the problem of unnatural examples. For example, to allow "I don't know" answers, or to be tolerant to some incorrect answers Angluin and Slonim (1994); Angluin et al. (1997); Blum et al. (1995); Sloan and Turán (1994); Bisht et al. (2008).

3. Setting

We consider binary classification where the instance space is $\mathcal{X} = \mathcal{X}_n = \{-1,1\}^n$ and the label space is $\mathcal{Y} = \{0,1\}$. The learner receives a *training set*

$$S = \{(\mathbf{x}_1, h^\star(\mathbf{x}_1)), (\mathbf{x}_2, h^\star(\mathbf{x}_2)), \ldots, (\mathbf{x}_m, h^\star(\mathbf{x}_m))\} \in (\mathcal{X}\times\mathcal{Y})^m$$

where the $\mathbf{x}_i$'s are sampled i.i.d. from some *unknown* distribution $\mathcal{D}$ on $\mathcal{X}$ and $h^\star : \mathcal{X} \to \mathcal{Y}$ is some *unknown* hypothesis. The learner is also allowed to make membership queries. Namely, to call an oracle, which receives as input some $\mathbf{x} \in \mathcal{X}$ and returns $h^\star(\mathbf{x})$. We say that a membership query for $\mathbf{x} \in \mathcal{X}$ is *q-local* if there exists a training example $\mathbf{x}_i$ whose Hamming distance from $\mathbf{x}$ is at most q. The learner returns (a description of) a hypothesis $\hat{h} : \mathcal{X} \to \mathcal{Y}$. The goal is to approximate $h^\star$, namely to find $\hat{h} : \mathcal{X} \to \mathcal{Y}$ with *loss* as small as possible, where the loss is defined as $L_{\mathcal{D},h^\star}(\hat{h}) = \Pr_{\mathbf{x}\sim\mathcal{D}}\left(\hat{h}(\mathbf{x}) \ne h^\star(\mathbf{x})\right)$. We will focus on the realizable case where $h^\star$ is assumed to be in a hypothesis class $\mathcal{H}$, and will require algorithms to return a hypothesis with loss $< \epsilon$ in time that is polynomial in n and $\frac{1}{\epsilon}$.

Definition 1 (Membership-Query Learning Algorithm) *We say that a learning algorithm* $\mathcal{A}$ efficiently learns $\mathcal{H}$ with q-local membership queries *if*

- *There exists a function $m_{\mathcal{A}}(n,\epsilon) \le \mathrm{poly}\left(n, \frac{1}{\epsilon}\right)$, such that for every pair $(\mathcal{D}, h^\star)$ with $h^\star \in \mathcal{H}$ and every $\epsilon > 0$, if $\mathcal{A}$ is given access to membership queries, and a training sequence*

$$S = \{(\mathbf{x}_1, h^\star(\mathbf{x}_1)), (\mathbf{x}_2, h^\star(\mathbf{x}_2)), \ldots, (\mathbf{x}_m, h^\star(\mathbf{x}_m))\}$$

  where the $\mathbf{x}_i$'s are sampled i.i.d. from $\mathcal{D}$ and $m \ge m_{\mathcal{A}}(n,\epsilon)$, then with probability of at least[2] *$\frac{3}{4}$ (over the choice of S), the output $\hat{h}$ of $\mathcal{A}$ satisfies $L_{\mathcal{D},h^\star}(\hat{h}) < \epsilon$.*

- *Given a training set of size m*
  - *$\mathcal{A}$ asks at most $\mathrm{poly}(m,n)$ membership queries, all of them are q-local*
  - *$\mathcal{A}$ runs in time $\mathrm{poly}(m,n)$.*

2. The success probability can be amplified to $1-\delta$ by repetition.

– *The hypothesis returned by $\mathcal{A}$ can be evaluated in time* $\mathrm{poly}(m, n)$.

We remark that learning with 0-local queries is equivalent to PAC learning, while learning with n-local queries is equivalent to PAC+MQ learning. We will also use a more general notion of a learning problem that is not necessarily defined by a hypothesis class. Specifically, we define a *model class* as a collection $\mathcal{M}$ of pairs $(\mathcal{D}, h^\star)$, where $\mathcal{D}$ is a distribution of $\{\pm 1\}^n$ and $h^\star$ is a function from $\{\pm 1\}^n$ to $\{0, 1\}$. Learnability of a model class is defined similarly to learnability of a hypothesis class (definition 1), except that instead of requiring that $h^\star \in \mathcal{H}$, we require that $(\mathcal{D}, h^\star) \in \mathcal{M}$.

4. A class for which local queries are useful

Our first results give an example to a hypothesis class that is not efficiently learnable without membership queries (assuming that it is hard to learn decision trees), yet it is efficiently learnable with 1-local queries. We define the class of *strongly mutually exclusive DNFs*, denoted $\mathrm{SME} - \mathrm{DNF}$, as the class of functions computed by a poly-sized[3] DNF formula, in which any pair of different terms contain two opposite literals.

Theorem 2 $\mathrm{SME} - \mathrm{DNF}$ *is efficiently learnable with* 1*-local queries*

Theorem 3 *Learning* $\mathrm{SME} - \mathrm{DNF}$ *(without queries) is as hard as learning decision trees.*

As we elaborate next, our positive result (theorem 2) is valid for a more general learning problem (yet one that is not captured by a function class), which we call *learning DNFs with representative examples*. When evaluating a DNF formula on a given example, we check a few conditions corresponding to the formula's terms, and deem the example positive if one of them holds. We will consider the case that for each of these conditions, there is a chance to see a "representative example", that satisfies it in a strong or clear way. Concretely, there is a a chance to see an example for which this condition is the only one that is valid, and in a strong sense – no other condition can be made true by flipping a single coordinate. In the sequel, we denote by $h_F : \{-1, 1\}^n \to \{0, 1\}$ the function induced by a DNF formula F over n variables.

Definition 4 *Let $F = T_1 \vee T_2 \vee \ldots \vee T_d$ be a DNF formula. We say that an example $\mathbf{x} \in \{-1, 1\}^n$ is* representative *for a term T_i (with respect to the formula F) and denote $T_i(\mathbf{x}) \equiv 1$ if (i) $\mathbf{x}$ satisfies T_i and only T_i (In particular, $h_F(\mathbf{x}) = 1$), and (ii) if we denote $\mathbf{x}^{\oplus j} = (x_1, \ldots, x_{j-1}, -x_j, x_{j+1}, \ldots, x_n)$, then for every $j \in [n]$, and term $k \neq i$, $T_k(\mathbf{x}^{\oplus j}) = 0$.*

Definition 5 *Let $h^\star$ be a function from $\{-1, 1\}^n$ to $\{0, 1\}$, let $\mathcal{D}$ be a distribution on $\{-1, 1\}^n$, and let $F = T_1 \vee T_2 \vee \ldots \vee T_d$ be a DNF formula. We say that $(\mathcal{D}, h^\star)$ is* realized by F with representative examples *if $h^\star = h_F$ and for every term T_i, $\Pr_{\mathbf{x} \sim \mathcal{D}} (T_i(\mathbf{x}) \equiv 1 | T_i(\mathbf{x}) = 1) \geq \frac{1}{\mathrm{poly}(n)}$. We denote by* RDNF *the collection of pairs $(\mathcal{D}, h^\star)$ that are realized by a poly-sized DNF formula with representative examples.*

Theorem 6 RDNF *is efficiently learnable with* 1*-local queries.*

3. The actual polynomial can be any polynomial. All the results are valid for any choice a polynomial. This comment applies to any further definition that involves a polynomial without a concrete specification.

It is not hard to see that if $h^\star \in \mathrm{SME-DNF}$ then for any $\mathcal{D}$, $(\mathcal{D}, h^\star) \in \mathrm{RDNF}$. Hence, theorem 6 implies theorem 2.

We next demonstrate that DNFs with representative example goes beyond $\mathrm{SME-DNF}$. Indeed, the following example shows that under the uniform distribution, read once DNFs[4] with terms of size $\log_2(n)$ are realized by a poly-sized DNFs with representative examples. We note that such formuals are known to be PAC learnable under the unform distribution (e.g. Schapire (1994)), but with quite complex algorithms and analysis. Example 1 implies that once we have 1-local queries at our disposal, we get a simple algorithm for the problem with a simple analysis.

Example 1 (Read once DNFs under the uniform distribution) *Let $F = T_1 \vee \ldots \vee T_d$ be a read once DNF formula over n variables, where each T_i has $\log_2(n)$ literals. For uniform $\mathbf{x} \in \{\pm 1\}^n$, we have*

$$\begin{aligned}
\Pr\left(T_1(\mathbf{x}) \equiv 1 | T_1(\mathbf{x}) = 1\right) &= \prod_{i=2}^{d} \Pr\left(\mathbf{x} \textit{ falsifies at least } 2 \textit{ literals in } T_i\right) \\
&= \left(1 - \frac{1}{n} - \frac{\log_2(n)}{n}\right)^{d-1} \\
&\geq \left(1 - \frac{1}{n} - \frac{\log_2(n)}{n}\right)^{n} \\
&= \left(e^{-1} + o(1)\right)^{\log_2(n)} \\
&= n^{-\log_2(e) + o(1)} \\
&\geq \frac{1}{\mathrm{poly}(n)}
\end{aligned}$$

4.1. An algorithm – proof of theorem 6

Theorem 7 *Algorithm 1 learns with 1-local-queries poly-sized* DNF*s with representative examples.*

Idea The algorithm is based on the following claim that follows easily from definition 4.

Claim 1 *Let $F = T_1 \vee T_2 \vee \ldots \vee T_d$ be a* DNF *formula over $\{-1, 1\}^n$. For every $\mathbf{x} \in \{-1, 1\}^n$ that satisfies a term T_i representatively (with respect to F), and for every $j \in [n]$ it holds that:*

$$h_F(\mathbf{x}^{\oplus j}) = 1 \iff \textit{the term } T_i \textit{ does not contain the variable } x_j$$

By this claim, if $\mathbf{x}$ is a representative example for a certain term T, one can easily reconstruct T. Indeed, by flipping the value of each variable and checking if the label changes, one can infer which variables appear in T. Furthermore, the sign of these variables can be inferred from their sign in $\mathbf{x}$. Hence, after seeing a representative example for all terms, one can have a list of terms containing all terms in the DNF. This list might have terms that are not part of the DNF. Yet, such terms can be thrown away later by testing if they make wrong predictions.

4. That is, DNF formulas in which any variable appears in at most a single term.

Algorithm 1

Input: $S_1, S_2 \in (\{-1,1\}^n \times \{0,1\})^m$

Output: A DNF formula H

```
Start with an empty DNF formula $H$
for all positive examples $(\mathbf{x}, y) \in S_1$ do
  Define $T = x_1 \wedge \overline{x_1} \wedge x_2 \wedge \overline{x_2} \wedge \ldots \wedge x_n \wedge \overline{x_n}$
  for $1 \le j \le n$ do
    Query $\mathbf{x}^{\oplus j}$ to get $h^\star(\mathbf{x}^{\oplus j})$
    if $h^\star(\mathbf{x}^{\oplus j}) = 1$ then
      Remove $x_j$ and $\overline{x_j}$ from $T$
    if $h^\star(\mathbf{x}^{\oplus j}) = 0$ then
      if $x_j = 1$ then
        Remove $\overline{x_j}$ from $T$
      if $x_j = 0$ then
        Remove $x_j$ from $T$
  $H = H \vee T$
for all $T$ in $H$ do
  if $T(\mathbf{x}) = 1$ but $y = 0$ for some $(\mathbf{x}, y) \in S_2$ then
    Remove $T$ from $H$
Return H
```

Proof (of Theorem 7) We will show that algorithm 1 learns with 1-local-queries any function that is realized by a DNF of size $\le n^2$ with representative examples. Adapting the proof to hold with n^c instead of n^2, for any $c > 0$, is straight-forward. Likewise, we assume that for every term T_i, $\Pr_{\mathbf{x}\sim\mathcal{D}}(T_i(\mathbf{x}) \equiv 1 | T_i(\mathbf{x}) = 1) \ge \frac{1}{n}$. Again, adapting the proof to hold with n^c instead of n^2, for any $c > 0$, is straight-forward. First, it is easy to see that this algorithm is efficient. Now, fix a distribution $\mathcal{D}$ and let $h^\star : \{-1,1\}^n \to \{0,1\}$ be a hypothesis that is realized, w.r.t. $\mathcal{D}$, by a DNF formula $F = T_1 \vee T_2 \vee \ldots \vee T_d,\ d \le n^2$ with representative examples. Let $\epsilon > 0$, and suppose we run the algorithm on two indepent samples from $\mathcal{D}$, denoted $S_1 = \{(\mathbf{x}_i, h^\star(\mathbf{x}_i))\}_{i=1}^{m_1}$ and $S_2 = \{(\mathbf{x}'_i, h^\star(\mathbf{x}'_i))\}_{i=1}^{m_2}$. We will show that if $m_1 \ge \frac{32n^3}{\epsilon}\log\frac{32n^2}{\epsilon} \ge \frac{32nd}{\epsilon}\log\frac{32d}{\epsilon}$ and $m_2 \ge \frac{32m_1}{\epsilon}\log\frac{32m_1}{\epsilon}$ then with probability of at least $\frac{3}{4}$, the algorithm will return a function $\hat{h}$ with $L_{\mathcal{D},h^\star}(\hat{h}) < \epsilon$. Let H be the DNF formula returned by the algorithm, and let $\hat{h}$ be the function induced by H. We have that

$$\begin{aligned} L_{\mathcal{D},h^\star}(\hat{h}) &= \Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x}) \ne \hat{h}(\mathbf{x})\right) \\ &= \Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x}) = 1 \text{ and } \hat{h}(\mathbf{x}) = 0\right) + \Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x}) = 0 \text{ and } \hat{h}(\mathbf{x}) = 1\right) \end{aligned}$$

The proof will now follow from claims 2 and 3.

Claim 2 *With probability at least $\frac{7}{8}$ over the choice of S_1, S_2 we have*

$$\Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x}) = 1 \textit{ and } \hat{h}(\mathbf{x}) = 0\right) \le \frac{\epsilon}{2} \tag{1}$$

Proof We first note that if there is a representative example $\mathbf{x}$ for a term T_i in S_1, then T_i will be in the output formula. Indeed, in the for-loop that go over the examples in S_1, when processing the example $(\mathbf{x}, h^\star(\mathbf{x}))$, it is not hard to see that T_i will be added. We furthermore claim that the term T_i won't be removed at the for loop that tests the terms collected in the first loop. Indeed, if for some $(\mathbf{x}, h^\star(\mathbf{x})) \in S_2$ we have $T_i(\mathbf{x}) = 1$, it must be the case that $h^\star(\mathbf{x}) = 1$. Now, we say that the term T_i is *revealed* if we see a representative example for this term in S_1. We also denote $p_i = \Pr_{\mathbf{x}\sim\mathcal{D}}\left(T_i(\mathbf{x}) = 1\right)$. We have

$$\begin{aligned}\Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x}) = 1 \text{ and } \hat{h}(\mathbf{x}) = 0\right) &\le \sum_{i=1}^{d} \Pr_{\mathbf{x}\sim\mathcal{D}}\left(T_i(\mathbf{x}) = 1 \text{ and } \hat{h}(\mathbf{x}) = 0\right) \\ &\le \sum_{i:T_i \text{ is not revealed}} p_i\end{aligned}$$

Now, by the assumption that $h^\star$ is realized with representative examples, the probability (over the choice of S_1, S_2) that T_i is not revealed is at most $\left(1 - \frac{p_i}{n}\right)^{m_1}$. Hence, if we denote by A_i the event that T_i is not revealed, we have

$$\begin{aligned}\mathop{\mathbb{E}}_{S_1\sim\mathcal{D}^{m_1}}\left[\Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x}) = 1 \text{ and } \hat{h}(\mathbf{x}) = 0\right)\right] &\le \mathbb{E}_{S_1\sim\mathcal{D}^{m_1}}\left[\sum_{i=1}^{d} p_i \cdot 1_{A_i}\right] \\ &= \sum_{i=1}^{d} p_i \mathbb{E}_{S_1\sim\mathcal{D}^{m_1}}[1_{A_i}] \\ &= \sum_{i=1}^{d} p_i \Pr_{S_1\sim\mathcal{D}^{m_1}}[A_i] \\ &= \sum_{i=1}^{d} p_i \left(1 - \frac{p_i}{n}\right)^{m_1} \\ &= \sum_{i|p_i<\frac{\epsilon}{32d}} p_i \left(1 - \frac{p_i}{n}\right)^{m_1} + \sum_{i|p_i\ge\frac{\epsilon}{32d}} p_i \left(1 - \frac{p_i}{n}\right)^{m_1} \\ &\le \sum_{i|p_i<\frac{\epsilon}{32d}} \frac{\epsilon}{32d} + \sum_{i|p_i\ge\frac{\epsilon}{32d}} \left(1 - \frac{p_i}{n}\right)^{m_1} \\ &\le d \cdot \frac{\epsilon}{32d} + \sum_{i|p_i\ge\frac{\epsilon}{32d}} e^{-\frac{m_1 p_i}{n}} \\ &\le \frac{\epsilon}{32} + d \cdot e^{-\frac{m_1 \epsilon}{32dn}}\end{aligned}$$

Since $m_1 \ge \frac{32dn}{\epsilon} \log \frac{32d}{\epsilon}$ the last expression is bounded by $\frac{\epsilon}{16}$. By Markov's inequality we conclude that the probabilty over the coice of S_1, S_2 that (1) does not lold is less than $\frac{1}{8}$. □

Claim 3 *With probability at least $\frac{7}{8}$ over the choice of S_1, S_2 we have*

$$\Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x}) = 0 \textit{ and } \hat{h}(\mathbf{x}) = 1\right) \le \frac{\epsilon}{2} \tag{2}$$

Proof Let $\hat{T}_1, \ldots, \hat{T}_r$ be the terms that were added to H at the first for-loop. Denote

$$q_i = \Pr_{\mathbf{x}\sim\mathcal{D}}\left(\hat{T}_i(\mathbf{x}) = 1 \text{ and } h^\star(\mathbf{x}) = 0\right)$$

We have

$$\Pr_{\mathbf{x}\sim\mathcal{D}}\left(h^\star(\mathbf{x})=0 \text{ and } \hat{h}(\mathbf{x})=1\right) \quad\leq \sum_{i\,:\,\hat{T}_i \text{ is not removed}} q_i$$

Now, the probability that $\hat{T}_i$ is not removed is $(1-q_i)^{m_2}$. Hence, using an argument similar to the one used in the proof of claim 2, and since $m_2 \geq \frac{32m_1}{\epsilon}\log\frac{32m_1}{\epsilon} \geq \frac{32r}{\epsilon}\log\frac{32r}{\epsilon}$, the claim follows. □

□

4.2. A lower bound – proof of theorem 3

In this section we provide evidence that the use of queries in our algorithm is crucial. We will show that the problem of learning poly-sized decision trees can be reduced to the problem of learning strongly mutually exclusive DNFs. As learning decision trees is conjectured to be intractable, this reduction serves as an indication that learning strongly mutually exclusive DNFs (and hence learning DNFs with representative examples)is hard without membership queries. In the sequel we denote by TREE the class of functions from $\{\pm 1\}^n$ to $\{0,1\}$ that are realized by a poly-sized decision tree.

Theorem 8 *Learning* $\mathrm{SME}-\mathrm{DNF}$ *is as hard as learning* TREE.

We denote by h_T the function induced by a decision tree T. The proof will use the following claim:

Claim 4 *There exists a mapping (a reduction)* $\varphi : \{-1,1\}^n \to \{-1,1\}^{2n}$*, that can be evaluated in* $poly(n)$ *time so that for every decision tree* $\mathcal{T}$ *over* $\{-1,1\}^n$ *there exists a strongly mutually exclusive* DNF *formula* F *over* $\{-1,1\}^{2n}$ *such that the following holds:*

1. *The number of terms in* F *is upper bounded by the number of leaves in* $\mathcal{T}$
2. $h_{\mathcal{T}} = h_F \circ \varphi$

Proof Define φ as follows:

$$\forall \mathbf{x}=(x_1,x_2,\ldots,x_n)\in\mathcal{X}_n \qquad \varphi(x_1,x_2,\ldots,x_n)=(x_1,x_1,x_2,x_2,\ldots,x_n,x_n)$$

Now, for every tree $\mathcal{T}$, we will build the desired DNF formula F as follows: First we build a DNF formula F' over $\{-1,1\}^n$. Every leaf labeled '1' in $\mathcal{T}$ will define the following term- take the path from the root to that leaf and form the logical AND of the literals describing the path. F' will be a disjunction of these terms. Now, for every term T in F' we will define a term $\phi(T)$ over $\mathcal{X}_{2n}$ in the following way: Let $P_T=\{i\in[n] : x_i \textit{ appear in } T\}$ and $N_T=\{i\in[n] : \overline{x_i} \textit{ appear in } T\}$. So

$$T=\bigwedge_{j\in P_T} x_j \bigwedge_{j\in N_T}\overline{x_j}$$

Define

$$\phi(T)=\bigwedge_{j\in P_T} x_{2j-1}\bigwedge_{j\in P_T} x_{2j}\bigwedge_{j\in N_T}\overline{x_{2j-1}}\bigwedge_{j\in N_T}\overline{x_{2j}}$$

Finally, define F to be the DNF formula over $\mathcal{X}_{2n}$ given by

$$F = \bigvee_{T \in F'} \phi(T)$$

We will now prove that φ and F satisfy the required conditions. First, φ can be evaluated in linear time in n. Second, it is easy to see that $h_{\mathcal{T}} = h_F \circ \varphi$, and as every term in F matches one of $\mathcal{T}$'s leaves, the number of terms in F cannot exceed the number of leaves in $\mathcal{T}$. It is left to show that F is strongly mutually exclusive. Indeed, if $\varphi(T_1)$ and $\varphi(T_2)$ are two terms in F, then T_1 and T_2 corresponds to two root-to-leaf paths $\mathcal{P}_1, \mathcal{P}_2$ in $\mathcal{T}$. Now, the two variables in $\{\pm 1\}^{2n}$ that corresponds to the node in which $\mathcal{P}_1$ and $\mathcal{P}_2$ fork, appear in both $\varphi(T_1)$ and $\varphi(T_2)$, but with opposite signs. $\square$

We are now ready to prove theorem 8.

Proof [of theorem 8] Suppose that $\mathcal{A}$ learns size-n strongly mutually exclusive DNFs. Using the reduction from claim 4 we will build an efficient algorithm $\mathcal{B}$ that learns size-n decision trees. For every training set

$$S = \{(\mathbf{x}_1, h^\star(\mathbf{x}_1)), (\mathbf{x}_2, h^\star(\mathbf{x}_2)), \ldots, (\mathbf{x}_m, h^\star(\mathbf{x}_m))\} \in (\mathcal{X}_n \times \{0,1\})^m$$

we define

$$\varphi(S) := \{(\varphi(\mathbf{x}_1), h^\star(\mathbf{x}_1)), (\varphi(\mathbf{x}_2)), h^\star(\mathbf{x}_2)), \ldots, (\varphi(\mathbf{x}_m), h^\star(\mathbf{x}_m))\} \in (\mathcal{X}_{2n} \times \{0,1\})^m$$

The algorithm $\mathcal{B}$ will work as follows: Given a training set S, $\mathcal{B}$ will run $\mathcal{A}$ on $\varphi(S)$, and will return $\hat{h} \circ \varphi$, where $\hat{h}$ is the hypothesis returned by $\mathcal{A}$. Since φ can be evaluated in $poly(n)$ time and $\mathcal{A}$ is efficient, $\mathcal{B}$ is also efficient. We will prove that $\mathcal{B}$ learns size-n trees. Since $\mathcal{A}$ learns size-n strongly mutually exclusive DNFs, there exists a function $m_{\mathcal{A}}(n, \epsilon) \le \text{poly}\left(n, \frac{1}{\epsilon}\right)$, such that if $\mathcal{A}$ is given a training sequence

$$S = \{(\mathbf{x}_1, h^\star(\mathbf{x}_1)), (\mathbf{x}_2, h^\star(\mathbf{x}_2)), \ldots, (\mathbf{x}_m, h^\star(\mathbf{x}_m))\} \in (\mathcal{X}_n \times \{0,1\})^m$$

where the $\mathbf{x}_i$'s are sampled i.i.d. from a distribution $\mathcal{D}$, $h^\star$ is realized by a poly-sized strongly mutually exclusive DNF, and $m \ge m_{\mathcal{A}}(n, \epsilon)$, then with probability of at least $\frac{3}{4}$ (over the choice of S), the output $\hat{h}$ of $\mathcal{A}$ satisfies $L_{\mathcal{D}, h^\star}(\hat{h}) \le \epsilon$. Let $\mathcal{D}$ be a distribution on $\mathcal{X}_n$ and let $h_{\mathcal{T}}$ be a hypothesis that can be realized by a tree with $\le n$ leafs. Define a distribution $\tilde{\mathcal{D}}$ on $\mathcal{X}_{2n}$ by,

$$(\tilde{\mathcal{D}})(\mathbf{z}) = \begin{cases} \mathcal{D}(\mathbf{x}) & \text{if } \exists \mathbf{x} \in \mathcal{X}_n \text{ such that } \mathbf{z} = \varphi(\mathbf{x}) \\ 0 & \text{otherwise} \end{cases}$$

Now, since $h_{\mathcal{T}}$ is realized by $\mathcal{T}$, from the conditions that φ satisfies, we get that $h_{\mathcal{T}} = h \circ \varphi$, where h is realized by a strongly mutually exclusive DNF of size $\le n$. Now if $S \in (\mathcal{X}_n \times \{0,1\})^m$ is an i.i.d. sample with $m \ge m_{\mathcal{A}}(2n, \epsilon)$ we have that with probability of at least $\frac{3}{4}$ it holds that

$$\begin{aligned}
L_{\mathcal{D},h_{\mathcal{T}}}(\mathcal{B}(S)) &= L_{\mathcal{D},h_{\mathcal{T}}}(\hat{h}\circ\varphi) \\
&= \Pr_{\mathbf{x}\sim\mathcal{D}}[h_{\mathcal{T}}(\mathbf{x}) \neq \hat{h}\circ\varphi(\mathbf{x})] \\
&= \Pr_{\mathbf{x}\sim\mathcal{D}}[h\circ\varphi(\mathbf{x}) \neq \hat{h}\circ\varphi(\mathbf{x})] \\
&= \Pr_{\mathbf{z}\sim\tilde{\mathcal{D}}}[h(\mathbf{z}) \neq \hat{h}(\mathbf{z})] \\
&= L_{\tilde{\mathcal{D}},h}(\hat{h}) \\
&= L_{\tilde{\mathcal{D}},h}(\mathcal{A}(\varphi(S))) < \epsilon
\end{aligned}$$

$\square$

5. Conclusion and Future Work

We have shown that in the distribution free setting, for many hypothesis classes, local queries are not useful. As our proofs show, this stems from the fact that learning these classes without queries can be reduced to a case where local queries are pointless, in the sense that the answer to them is either always 1, or the label of the closest training example. On the other hand, the learning problem of DNFs with representative examples circumvents this property. Indeed, the underlying assumption enforces that local changes can change the label in a non-trivial manner. While this assumption might be intuitive in some cases, it is certainly very restrictive. Therefore, a natural future research direction is to seek less restrictive assumptions, that still posses this property.

More concrete direction arising from our work concern classes for which we have shown that $\left(\log^{0.99}(n)\right)$-local queries are unlikely to lead to efficient algorithms. We conjecture that for some $a > 0$ even (n^a)-local queries won't lead to efficient distribution free algorithms for these classes.

References

Dana Angluin. Learning regular sets from queries and counterexamples. *Information and computation*, 75(2):87–106, 1987.

Dana Angluin and Michael Kharitonov. When wont membership queries help? *Journal of Computer and System Sciences*, 50(2):336–355, 1995.

Dana Angluin and Donna K Slonim. Randomly fallible teachers: Learning monotone dnf with an incomplete membership oracle. *Machine Learning*, 14(1):7–26, 1994.

Dana Angluin, Mārtiņš Kriķis, Robert H Sloan, and György Turán. Malicious omissions and errors in answers to membership queries. *Machine Learning*, 28(2-3):211–255, 1997.

Pranjal Awasthi, Vitaly Feldman, and Varun Kanade. Learning using local membership queries. *arXiv preprint arXiv:1211.0996*, 2012.

Eric B Baum. Neural net algorithms that learn in polynomial time from examples and queries. *Neural Networks, IEEE Transactions on*, 2(1):5–19, 1991.

Eric B Baum and Kenneth Lang. Query learning can work poorly when a human oracle is used. In *International Joint Conference on Neural Networks*, volume 8, 1992.

Laurence Bisht, Nader H Bshouty, and Lawrance Khoury. Learning with errors in answers to membership queries. *Journal of Computer and System Sciences*, 74(1):2–15, 2008.

Avrim Blum and Steven Rudich. Fast learning of k-term dnf formulas with queries. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 382–389. ACM, 1992.

Avrim Blum, Prasad Chalasani, Sally A Goldman, and Donna K Slonim. Learning with unreliable boundary queries. In *Proceedings of the eighth annual conference on Computational learning theory*, pages 98–107. ACM, 1995.

Nader H Bshouty. Exact learning boolean functions via the monotone theory. *Information and Computation*, 123(1):146–153, 1995.

Amit Daniely and Shai Shalev-Shwatz. Complexity theoretic limitations on learning dnf's. In *COLT*, 2016.

Vitaly Feldman. On the power of membership queries in agnostic learning. *The Journal of Machine Learning Research*, 10:163–182, 2009.

Yoav Freund. Boosting a weak learning algorithm by majority. *Information and computation*, 121 (2):256–285, 1995.

Jeffrey Jackson. An efficient membership-query algorithm for learning dnf with respect to the uniform distribution. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 42–53. IEEE, 1994.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994.

Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

Robert E Schapire. Learning probabilistic read-once formulas on product distributions. *Machine Learning*, 14(1):47–81, 1994.

Michael Sipser. *Introduction to the Theory of Computation*, volume 2.

Robert H Sloan and György Turán. Learning with queries but incomplete information. In *Proceedings of the seventh annual conference on Computational learning theory*, pages 237–245. ACM, 1994.

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

Appendix A. Classes for which local queries are not useful

We first present a general technique to prove lower bounds on learning with local queries. For $A \subset \mathcal{X}_n$ and $q > 0$ denote

$$B(A,q) = \{\mathbf{x} \in \mathcal{X}_n \mid \exists \mathbf{a} \in A,\ d(\mathbf{x},\mathbf{a}) \le q\}$$

We say that a mapping $\varphi : \{-1,1\}^n \to \{-1,1\}^{n'}$ is a *q-reduction of type A* from a class $\mathcal{H}$ to a class $\mathcal{H}'$ if the following holds:

1. φ is efficiently computable.
2. For every $h \in \mathcal{H}$ there is $h' \in \mathcal{H}'$ such that
   (a) $h = h' \circ \varphi$
   (b) The restriction of h' to $B(\varphi(\mathcal{X}_n),q) \setminus \varphi(\mathcal{X}_n)$ is the constant function 1.

We say that a mapping $\varphi : \{-1,1\}^n \to \{-1,1\}^{n'}$ is a *q-reduction of type B* if the same requirements hold, except that (2b) is replaced by the following condition: For every $\mathbf{z} \in B(\varphi(\mathcal{X}_n),q)$ there is a unique $\mathbf{x} \in \mathcal{X}_n$ satisfying $d(\mathbf{z},\varphi(\mathbf{x})) \le q$, and furthermore, $h'(\mathbf{z}) = h(\mathbf{x})$.

Lemma 9 *Suppose that there is a q-reduction φ from $\mathcal{H}$ to $\mathcal{H}'$. Then, learning $\mathcal{H}'$ with q-local queries is as hard as PAC learning $\mathcal{H}$.*

Proof (sketch) Suppose that $\mathcal{A}'$ learns $\mathcal{H}'$ with q-local queries. We need to show that there is an algorithm that learns $\mathcal{H}$. Indeed, by an argument similar to the one in theorem 8, it is not hard to verify that the following algorithm learns $\mathcal{H}$. Given a sample $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m,y_m)\} \subset \{-1,1\}^n \times \{0,1\}$, run $\mathcal{A}'$ on the sample $\{(\varphi(\mathbf{x}_1),y_1),\ldots,(\varphi(\mathbf{x}_m),y_m)\} \subset \{-1,1\}^{n'} \times \{0,1\}$. Whenever $\mathcal{A}'$ makes a q-local query for $\mathbf{z} \in \{-1,1\}^{n'}$, respond 1 if φ is of type A. If φ is of type B, respond y_i, where $\mathbf{x}_i$ is the unique training sample satisfying $d(\mathbf{z},\varphi(\mathbf{x}_i)) \le q$. Finally, if $\mathcal{A}'$ returned the hypothesis h', return $h' \circ \varphi$. $\square$

We next use Lemma 9 to prove that for several classes, local queries are not useful. Namely, if the class is learnable with local queries then it is also learnable without queries. We will use the following terminology. We say that q-local queries cannot help to learn a class $\mathcal{H}$ if $\mathcal{H}$ is learnable if and only if it is learnable with q-local queries.

Corollary 10 *For every $\epsilon_0 > 0$, $\left(n^{1-\epsilon_0}\right)$-local queries cannot help to learn poly-sized DNFs, intersection of halfspaces, decision lists, depth-d circuits for any $d = d(n) \ge 2$, and depth-d threshold circuits for any $d = d(n) \ge 2$.*

Proof We will only prove the corollary for DNFs. The proofs for the remaining classes are similar. Also, for simplicity, we will assume that $\epsilon_0 = \frac{1}{3}$. Consider the mapping $\varphi : \{-1,1\}^n \to \{-1,1\}^{n^3}$ that replicates each coordinate n^2 times. To establish the corollary, we show that φ is an $\left((n')^{\frac{2}{3}} - 1\right)$-reduction of type A from poly-sized DNF to poly-sized DNF.

Indeed, let $F = T_1 \vee \ldots \vee T_d$ be a DNF formula on n variables, consider the following formula on the n^3 variables $\{x_{i,j}\}_{1\le i\le n, 1\le j\le n^2}$. Let,

$$\begin{aligned}
T'_t(\{x_{i,j}\}_{i,j}) &= T_t(x_{1,1},\ldots,x_{n,1}) \\
G'(\{x_{i,j}\}_{i,j}) &= \vee_{i=1}^{n} \vee_{j=1}^{n^2-1} (x_{i,j} \wedge \neg x_{i,j+1}) \vee (x_{i,j+1} \wedge \neg x_{i,j}) \\
F' &= \left(T'_1 \vee \ldots \vee T'_d\right) \vee G'
\end{aligned}$$

It is not hard to verify that $h_F = h_{F'} \circ \varphi$. Moreover, if $\mathbf{x} = \{x_{i,j}\}_{i,j} \in B(\varphi(\mathcal{X}_n), n^2 - 1) \setminus \varphi(\mathcal{X}_n)$ then $x_{i,j} \neq x_{i,j+1}$ for some i, j, meaning that $h_G(x) = 1$ and therefore also $h_F(x) = 1$. □

Corollary 11 *For every $\epsilon_0 > 0$, $\left(n^{1-\epsilon_0}\right)$-local queries cannot help to learn poly-sized automata.*

Proof Again, for simplicity, we assume that $\epsilon_0 = \frac{1}{3}$, and consider the mapping $\varphi : \{-1,1\}^n \to \{-1,1\}^{n^3}$ that replicates each coordinate n^2 times. To establish the corollary, we show that φ is an $\left((n')^{\frac{2}{3}} - 1\right)$-reduction of type A from poly-sized automata to poly-sized automata.

Indeed, let A be an automaton on n variables. It is enough to show that there is an automaton A' on the n^3 variables $\{x_{i,j}\}_{1 \le i \le n, 1 \le j \le n^2}$ such that (i) the size of A' is polynomial, (ii) A' accepts any string in $B(\varphi(\mathcal{X}_n), n^2 - 1) \setminus \varphi(\mathcal{X}_n)$, and (iii) A' accepts $\varphi(x)$ if and only if A accepts x. Now, by the product construction of automatons Sipser, if A'_1, A'_2 are automata that induce the functions $h_{A'_1}, h_{A'_2} : \{-1,1\}^{n^3} \to \{0,1\}$, then the function $h_{A'_1} \vee h_{A'_2}$ can be induced by an automaton of size $|A'_1| \cdot |A'_2|$. Hence, it is enough to show that there are poly-sized automata A'_1, A'_2 that satisfies (ii) and (iii) respectively.

A construction of a size $O(n^2)$ automaton that satisfies (ii) is a simple exercise. We next explain how to construct a poly-sized automaton A'_2 satisfying (iii). The states of A'_2 will be the $S(A) \times [n^2]$ (here, $S(A)$ denotes the set of states of A). The start state of A'_2 will be $(\alpha_0, 1)$, where α_0 is the start state of A, and the accept states of A'_2 will be the cartesian product of the accept states of A with $[n^2]$. Finally if $\delta : S(A) \times \{-1,1\} \to S(A)$ is the transition function of A, then the transition function of A'_2 is defined by

$$\delta'((\alpha, i), b) = \begin{cases} (\alpha, i+1) & 1 \le i < n^2 \\ (\delta(\alpha, b), 1) & i = n^2 \end{cases}$$

It is not hard to verify that A'_2 satisfies (iii). □

In the next corollary, a *Junta* of size t is a function $h : \{-1,1\}^n \to \{0,1\}$ that depends on $\le \log(t)$ variables. The rational behind this definition of size, is the fact that in order to describe a general function that depends on K variables, at least 2^K bits are required. Likewise, the sample complexity of learning Juntas is proportional to t rather than $\log(t)$

Corollary 12 *For every constant $q_0 > 0$, q_0-local queries cannot help to learn poly-sized Juntas.*

Proof Consider the mapping $\varphi : \{-1,1\}^n \to \{-1,1\}^{(2q_0+1)n}$ that replicates each coordinate $2q_0 + 1$ times. To establish the corollary, we show that φ is a q_0-reduction of type B from poly-sized Juntas to poly-sized Juntas. Indeed, let $h : \{-1,1\}^n \to \{0,1\}$ be a function that depends on K variables. It is enough to show that there is a function h' on the variables $\{z_{i,j}\}_{i \in [n], j \in [2q_0+1]}$ that satisfies (i) $h = h' \circ \varphi$, (ii) for every $\mathbf{x} \in \mathcal{X}$, if $\mathbf{z} \in \{-1,1\}^{(2q_0+1)n}$ is obtained from $\varphi(\mathbf{x})$ by modifying $\le q_0$ coordinates, then $h'(\mathbf{z}) = h'(\varphi(\mathbf{x}))$ and (iii) h' depends on $(2q_0 + 1)K$ variables. It is not hard to check that the following function satisfies these requirements:

$$h'(\mathbf{z}) = h\left(\mathrm{MAJ}(z_{1,1}, \ldots, z_{1,2q_0+1}), \ldots, \mathrm{MAJ}(z_{n,1}, \ldots, z_{n,2q_0+1})\right)$$

□

We remark that by taking $q_0 = \log^{1-\epsilon_0}(n)$ for some $\epsilon_0 > 0$, and using a similar argument, it can be shown that an efficient algorithm for learning poly-sized Juntas with $\left(\log^{1-\epsilon_0}(n)\right)$-local queries would imply a PAC algorithm for poly-sized Juntas that runs in time $n^{O\left(\log^{1-\epsilon_0}(n)\right)}$.

Corollary 13 *For every constant $q_0 > 0$, q_0-local queries cannot help to learn poly-sized decision tress.*

Proof As with Juntas, consider the mapping $\varphi : \{-1,1\}^n \to \{-1,1\}^{(2q_0+1)n}$ that replicates each coordinate $2q_0 + 1$ times. To establish the corollary, we show that φ is a q_0-reduction of type B from poly-sized decision trees to poly-sized decision trees. Indeed, let $h : \{-1,1\}^n \to \{0,1\}$ be a function that is realized by a decision tree T. It is enough to show that there is a function h' on the variables $\{z_{i,j}\}_{i\in[n],j\in[2q_0+1]}$ that satisfies (i) $h = h' \circ \varphi$, (ii) for every $\mathbf{x} \in \mathcal{X}$, if $\mathbf{z} \in \{-1,1\}^{(2q_0+1)n}$ is obtained from $\varphi(\mathbf{x})$ by modifying $\le q_0$ coordinates, then $h'(\mathbf{z}) = h'(\varphi(\mathbf{x}))$ and (iii) h' can be realized by a tree of size $|T|^{2q_0+1}$. As we explain, this will hold for the following function:

$$h'(\mathbf{z}) = \text{MAJ}\left(h(z_{1,1}, \ldots, z_{n,1}), \ldots, h(z_{1,2q_0+1}, \ldots, z_{n,2q_0+1})\right)$$

It is not hard to check that h' satisfies (i) and (ii). As for (iii), consider the following tree T'. First replicate T on the variables $z_{1,1}, \ldots, z_{n,1}$. Then, on the obtained tree, replace each leaf by a replica of T on the variables $z_{1,2}, \ldots, z_{n,2}$. Then, again, replace each leaf by a replica of T on the variables $z_{1,3}, \ldots, z_{n,3}$. Continues doing so, until the leafs are replaced by a replica on the variables $z_{1,2q_0+1}, \ldots, z_{n,2q_0+1}$. Now, each leaf in the resulted tree corresponds to $(2q_0+1)$ root-to-leaf paths in the original tree T. The label of such leaf will be the majority of the labels of these paths. □

As with Juntas, we remark that by taking $q_0 = \log^{1-\epsilon_0}(n)$ for some $\epsilon_0 > 0$, and using a similar argument, it can be shown that an efficient algorithm for learning poly-sized trees with $\left(\log^{1-\epsilon_0}(n)\right)$-local queries would imply a PAC algorithm for poly-sized trees that runs in time $n^{O\left(\log^{1-\epsilon_0}(n)\right)}$.

In the sequel, we say that a function $h : \{-1,1\}^n \to \{0,1\}$ is realized by a poly-sized polynomial, if it is the function induced by a polynomial with polynomially many non-zero coefficient and degree $O(\log(n))$. A similar convention applies to the term poly-sized polynomial threshold function. We remark that the following corollary and its proof remain correct also if in our definition, we replace the number of coefficients with the ℓ^1 norm of the coefficients vector.

Corollary 14 *For every constant $q_0 > 0$, q_0-local queries cannot help to learn poly-sized polynomials, as well as poly-sized polynomial threshold functions.*

Proof We will prove the corollary for polynomials. The proof for polynomial threshold functions is similar. Consider the mapping $\varphi : \{-1,1\}^n \to \{-1,1\}^{(2q_0+1)n}$ that replicates each coordinate $2q_0 + 1$ times. To establish the corollary, we show that φ is a q_0-reduction of type B from poly-sized polynomials to poly-sized polynomials. Indeed, let $h : \{-1,1\}^n \to \{0,1\}$ be a poly-sized polynomial. It is enough to show that there is a poly-sized polynomial h' on the variables $\{z_{i,j}\}_{i\in[n],j\in[2q_0+1]}$ that satisfies (i) $h = h' \circ \varphi$, and (ii) for every $\mathbf{x} \in \mathcal{X}$, if $\mathbf{z} \in \{-1,1\}^{(2q_0+1)n}$ is obtained from $\varphi(\mathbf{x})$ by modifying $\le q_0$ coordinates, then $h'(\mathbf{z}) = h'(\varphi(\mathbf{x}))$. As we explain next, this will hold for the following polynomial:

$$h'(\mathbf{z}) = h\left(\text{MAJ}(z_{1,1}, \ldots, z_{1,2q_0+1}), \ldots, \text{MAJ}(z_{n,1}, \ldots, z_{n,2q_0+1})\right)$$

It is not hard to check that h' satisfies (i) and (ii). It remains to show that h' is poly-sized. Indeed, the majority function on $2q_0+1$ coordinates is a polynomial of degree $\le 2q_0+1$ with at most 2^{2q_0+1} non zero coefficients (this is true for any function on $2q_0 + 1$ coordinates). Hence, if we replace each variable in h by a polynomial of the form $\text{MAJ}(z_{i,1}, \ldots, z_{i,2q_0+1})$, the degree is multiplied by at most $(2q_0 + 1)$, while the number of non zero coefficients is multiplied by at most 2^{2q_0+1}. □

As with Juntas and decision trees, by taking $q_0 = \log^{1-\epsilon_0}(n)$ for some $\epsilon_0 > 0$, and using a similar argument, it can be shown that an efficient algorithm for learning poly-sized polynomials or polynomial threshold functions with $\left(\log^{1-\epsilon_0}(n)\right)$-local queries would imply a PAC algorithm for the same problem that runs in time $n^{O\left(\log^{1-\epsilon_0}(n)\right)}$.