hexagon:2610.00613v1

What Does the Refutation of the Exponential Time Hypothesis Mean for Learning Theory?

Xiaoyu Li*

Abstract

Two recent works by OpenAI and Anthropic refuted the Exponential Time Hypothesis (ETH). They count the satisfying assignments of a k-CNF on n variables in time $2^{n/(\log n)^{1/2-o(1)}}$. Many fine-grained lower bounds in learning theory rest on ETH and its relatives. We determine what the refutation changes for learning.

The answer is a *locality principle*: the new algorithms speed up learning when each labeled example constrains few bits of the hypothesis, and under the CNF form of the Strong Exponential Time Hypothesis, which survives, the speed-up ends at $\Theta(\log N)$ bits. Three transitions make the principle precise.

- **Locality.** Call a class with N-bit descriptions κ-*local* if each label depends on at most κ description bits, as for quantized models on sparse data. Then proper PAC learning, exact version-space counting and sampling, and the halving algorithm run in time $\mathrm{poly}(m) \cdot 2^{o(N)}$ on samples of size m for $\kappa = o((\log N)^{1/4}/\sqrt{\log\log N})$, and proper PAC learning at constant accuracy does so for $\kappa = o(\sqrt{\log N})$. At inverse-polynomial accuracy, proper learning of κ-local classes is equivalent to κ-SAT, up to a factor 2 in the locality. Under CNF-SETH, consistency for binarized perceptrons on $O(\log N)$-sparse data needs time $2^{(1-\gamma)N}$ for every $\gamma > 0$.
- **Accuracy.** Exact empirical risk minimization, proper agnostic learning and the exponential mechanism reduce to tensor contractions over a truncated polynomial ring. As a consequence, the Sparse Max-SAT Hypotheses of Alman and Vassilevska Williams are false. At constant accuracy, proper learning is SAT parameterized by formula length. With quasi-linear PCPs this gives a $2^{N/\mathrm{polylog}N}$ lower bound under a length form of ETH, which replaces Gap-ETH for reductions of quasi-linear size. At accuracy $N^{-3}/8$, the Max-3-SAT Hypothesis forces time $2^{(1-o(1))N}$.
- **Treewidth.** Exact inference in graphical models of bounded degree and arity takes time $2^{O(n/\sqrt{\log n})}$. On expanders this beats every elimination order and every structured DNNF. Under #CNF-SETH, time $2^{(1-\gamma)\mathrm{tw}}$ is again necessary once a model may have $\mathrm{tw}^{O(1)}$ variables.

We also take stock of the conditional lower bounds of learning theory: which ones fall, and which ones survive on which hypothesis. The next obstacle is a permutation problem. On difference vectors $e_i - e_j$, empirical risk minimization for real-valued halfspaces is weighted Feedback Arc Set, which the new algorithms do not reach.

*University of New South Wales, xiaoyulics@gmail.com

Contents

1 Introduction **3**
- 1.1 The locality principle . . . 3
- 1.2 Techniques . . . 6
- 1.3 Organization . . . 7

2 Preliminaries **7**
- 2.1 Notation and the learning model . . . 7
- 2.2 Local hypothesis classes . . . 8
- 2.3 The new algorithms . . . 8
- 2.4 The engine: sparse sum-products and error profiles . . . 9
- 2.5 Hypotheses that survive . . . 10

3 The locality transition **11**
- 3.1 Version spaces are κ-CNFs . . . 11
- 3.2 The realizable transfer theorem . . . 12
- 3.3 Learning is equivalent to bounded-width SAT . . . 12
- 3.4 Hardness at logarithmic locality . . . 13
- 3.5 Constant accuracy moves the threshold to $\sqrt{\log N}$ . . . 15

4 The accuracy transition **15**
- 4.1 Exact ERM and proper agnostic learning . . . 15
- 4.2 The exact exponential mechanism . . . 17
- 4.3 Constant accuracy is SAT by length . . . 17
- 4.4 Hardness at inverse-polynomial accuracy . . . 19

5 The treewidth transition **19**
- 5.1 Subexponential inference for bounded-degree models . . . 20
- 5.2 Separation from elimination and compilation . . . 20
- 5.3 Treewidth is necessary again for large models . . . 21

6 Learning theory after ETH: a ledger **22**
- 6.1 Hypotheses . . . 22
- 6.2 How lower bounds transfer . . . 23
- 6.3 The dense side: juntas and DNF . . . 25

7 Related work **25**

8 The permutation obstacle and open problems **26**
- 8.1 The permutation obstacle . . . 26
- 8.2 Open problems . . . 27
- 8.3 Conclusion . . . 28

A Roadmap of the appendix **38**

B Proofs for Section 2: the engine **38**

C Proofs for Section 3: the locality transition **41**
- C.1 Realizable transfer . . . 41
- C.2 The perceptron gadget and the equivalence . . . 42
- C.3 Width reduction . . . 44
- C.4 Hardness at logarithmic locality . . . 45
- C.5 Sample-sensitive realizable learning . . . 47

D Proofs for Section 4: the accuracy transition **48**
- D.1 Exact ERM, agnostic learning and Max-CSP . . . 48
- D.2 The exponential mechanism . . . 50
- D.3 Short PCPs and constant accuracy . . . 51
- D.4 Hardness at inverse-polynomial accuracy . . . 54

E Proofs for Section 5: the treewidth transition **55**

F Proofs for Sections 6 and 8 **58**

G Notation **60**

1 Introduction

The Exponential Time Hypothesis (ETH) of Impagliazzo and Paturi [IP01] states that 3-SAT on n variables needs time $2^{\Omega(n)}$. For a quarter century it was the standard way to turn NP-hardness into quantitative lower bounds. Its relatives did the same for exact constants (SETH), for approximation (Gap-ETH) and for counting (#ETH). Learning theory used all of them. ETH and Gap-ETH give lower bounds for proper learning of decision trees [KST23b, KST23a, KST24] and for computing the VC and Littlestone dimensions [MR17b, Man23]. They also give lower bounds for training neural networks [GKMR21, FHN22], for robust proper learning of halfspaces [DKM19, DKM20], and for computational–statistical tradeoffs [BKST25]. SETH gives quadratic lower bounds for kernel methods, attention and nearest neighbors [BIS17, AS23, ACSS20, Rub18]. In inference, ETH was the basis of the conclusion that bounded treewidth is necessary for efficient inference [KB09, KBvdG10].

In October 2026, two independent works refuted ETH [Ope26a, Cla26]. Both count the satisfying assignments of a k-CNF on n variables in deterministic time $2^{n/(\log n)^{1/2-o(1)}}$, with polynomial dependence on the number of clauses. One works by polynomial interpolation, the other by signed sums of tensor networks contracted in different orders, and both need sparse instances: linearly many constraints of bounded width. SETH, Gap-ETH, #ETH and NSETH each imply ETH, so all of them are false as well. Two forms survive. CNF-SETH states that CNF-SAT with polynomially many clauses of unbounded width needs time $2^{(1-\gamma)n}$, and the Max-3-SAT Hypothesis states the same for the maximum number of satisfiable clauses of a dense 3-CNF (Section 2.5).

This paper determines what the refutation changes in computational learning theory. Which learning problems become subexponential? Which lower bounds survive, and on which hypothesis? We use the new algorithms only through the three statements of Section 2.3.

1.1 The locality principle

A learning problem is a constraint satisfaction problem. Its variables are the bits of the hypothesis description, and its constraints are the labeled examples. For most classical classes, such as DNF, decision trees and halfspaces on dense data, every example constrains every description bit. The resulting formulas have clauses of width $\Theta(N)$, and the new algorithms do not reach this regime. The dividing line is the following notion.

Definition 1.1 (Informal; see Definition 2.2)**.** A class of hypotheses with N-bit descriptions $w \in \{0,1\}^N$ is *κ-local* if for each example x there is a set $R(x) \subseteq [N]$ of at most κ positions such that the label $h_w(x)$ depends only on the bits of w in $R(x)$.

Sparse data make quantized models local. A binarized perceptron $x \mapsto \mathbf{1}[\langle w, x\rangle \geq 0]$ with $w \in \{\pm 1\}^n$ is k-local on k-sparse inputs, because only the weights on the support of x matter. With b-bit weights, and for quantized embedding models and factorization machines on bags of k tokens, the locality is $\kappa = bk$. Two-layer binarized networks with r hidden units are $(rk + r)$-local. A graphical model of bounded degree and arity is local in the same sense: each factor reads a bounded number of variables. Our main finding is a principle.

> *The new algorithms speed up learning when each labeled example constrains few bits of the hypothesis description. Under CNF-SETH the speed-up ends when examples constrain $\Theta(\log N)$ bits.*

Three transitions make the principle precise. Figure 1 summarizes them.

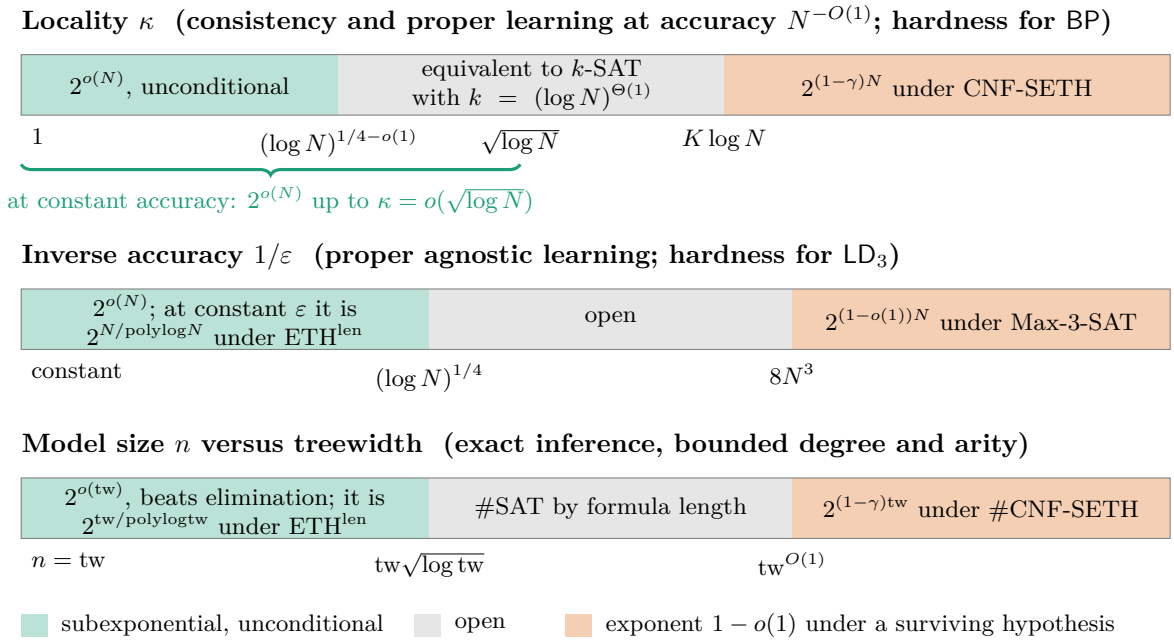

Figure 1: The three transitions. Each row is a parameter that grows to the right; the regimes are not drawn to scale. Green regions are upper bounds that use only the new algorithms [Ope26a, Cla26]; the lower bounds in green regions assume $\mathrm{ETH}^{\mathrm{len}}_C$. Row 1: Theorems 3.2, 3.4, 3.8 and 3.10. Row 2: Theorems 4.3, 4.9 and 4.12. Row 3: Theorems 5.1 and 5.5 and Corollaries 5.2, 5.3 and 5.6.

Transition 1: locality. For local classes, the version space of a sample is the solution set of a κ-CNF, so every computation on it reduces to counting κ-CNFs. Conversely, a padding gadget embeds k-SAT into binarized perceptrons on $2k$-sparse data.

Theorem A (Locality; Theorems 3.2, 3.4, 3.8 and 3.10)**.** *Let $\mathcal{H}$ be a κ-local class with N-bit descriptions.*

(1) Exact counting and uniform sampling of version spaces, the halving algorithm, and proper PAC learning at every accuracy run in time $\mathrm{poly}(m) \cdot 2^{O(\kappa^2 N \log\log N/\sqrt{\log N})}$. At constant accuracy, proper PAC learning runs in time $2^{O(\kappa N/\sqrt{\log N}+\kappa^2)}$.

(2) At inverse-polynomial accuracy, and up to a factor 2 in the locality, proper learning of κ-local classes is equivalent to κ-SAT.

(3) Under CNF-SETH, for every $\gamma > 0$ there is K such that consistency and proper PAC learning to inverse-polynomial accuracy of binarized perceptrons on $(K \log N)$-sparse data need time $2^{(1-\gamma)N}$.

For constant κ, the counting part of (1) is Corollary C.2 of [Ope26a] in learning terms. New are the uniformity in κ, the equivalence (2) and the hardness (3). Consistency and learning to inverse-polynomial accuracy are thus subexponential up to locality $(\log N)^{1/4-o(1)}$, and as hard as CNF-SAT at locality $\Theta(\log N)$. Between the two lies the open window of k-SAT for $k = (\log N)^{\Theta(1)}$. At constant accuracy the subexponential range extends to $(\log N)^{1/2-o(1)}$.

Transition 2: accuracy. For agnostic learning, the cost depends on the size of the sample, not on N. Exact ERM on m examples reads only κm description bits, and it is a sparse weighted constraint problem. We compute its entire error profile, the number of hypotheses with exactly j

errors for every j, by tensor contraction over the truncated polynomial ring $(\mathbb{Z}/2^P\mathbb{Z})[V]/(V^\ell)$. The lower bounds use a length form of ETH. The hypothesis $\mathrm{ETH}^{\mathrm{len}}_C$ states that SAT on CNFs of length L needs time $2^{L/(\log L)^C}$. It is the instance with an explicit rate of the polylog-slack hypotheses of [CRTY23]. The new algorithms refute it for $C < 1/2$, and for $C \geq 1/2$ it states that they are optimal up to polylogarithmic factors in the exponent.

Theorem B (Accuracy; Theorems 4.1, 4.3, 4.6, 4.9 and 4.12 and Corollaries 4.2 and 4.10)**.** *Let $\mathcal{H}$ be a κ-local class with N-bit descriptions and VC dimension d.*

(1) Exact ERM, the error profile and the exponential mechanism on a sample of size $m \geq \max\{\kappa, 2\}$ take time $\mathrm{poly}(N) \cdot 2^{O(\kappa m/\sqrt{\log m})}$. For literal disjunctions on 3-sparse data, consistency on m examples needs time $2^{m/(\log m)^{C+o(1)}}$ under $\mathrm{ETH}^{\mathrm{len}}_C$.

(2) Proper agnostic learning takes time $\mathrm{poly}(N) \cdot 2^{O(\kappa d/(\varepsilon^2\sqrt{\log(2d/\varepsilon)})+\kappa^2)}$ for constant confidence. At a small constant accuracy ε_0, proper learning of literal disjunctions on 3-sparse data, realizable or agnostic, needs time $2^{N/(\log N)^{C+O(1)}}$ under $\mathrm{ETH}^{\mathrm{len}}_C$. So at constant accuracy, proper learning of constant-local classes is SAT parameterized by formula length, up to polylogarithmic factors in the exponent.

(3) Under the Max-3-SAT Hypothesis, proper agnostic learning of literal disjunctions on 3-sparse data to accuracy $1/(8N^3)$ needs time $2^{(1-o(1))N}$.

With the quasi-linear PCPs of Dinur and of Ben-Sasson and Sudan [Din07, BSS08], $\mathrm{ETH}^{\mathrm{len}}_C$ replaces Gap-ETH for gap reductions of quasi-linear size, at a polylogarithmic loss in the exponent (Corollary 4.11). At inverse-polynomial accuracy, more data makes exact ERM harder: it is subexponential on samples of size $o(N\sqrt{\log N})$ and needs time $2^{(1-o(1))N}$ on samples of size $\Theta(N^3)$.

Transition 3: treewidth. Exact inference in graphical models is #SAT parameterized by the length of the model, and the textbook algorithms take time exponential in the treewidth.

Theorem C (Treewidth; Theorems 5.1 and 5.5 and Corollaries 5.2, 5.3 and 5.6)**.** *Consider Boolean graphical models of bounded degree and arity with small nonnegative integer weights.*

(1) On n variables, the partition function, all marginals, and a minimum-energy assignment can be computed exactly in time $2^{O(n/\sqrt{\log n})}$. On bounded-degree expanders, every elimination order and every structured DNNF needs time or size $2^{\Omega(n)}$. Under $\mathrm{ETH}^{\mathrm{len}}_C$, deciding whether the partition function is positive needs time $2^{\mathrm{tw}/(\log \mathrm{tw})^{C+o(1)}}\mathrm{poly}(n)$.

(2) Under #CNF-SETH, for every $\gamma > 0$, no algorithm computes partition functions in time $2^{(1-\gamma)\mathrm{tw}}\mathrm{poly}(n)$, already on models with $n \leq \mathrm{tw}^{O(1)}$ variables.

The treewidth-based algorithms take time $2^{\Theta(\mathrm{tw})}$. So they are not optimal when the model has $n = O(\mathrm{tw})$ variables, but they are optimal again when n may be polynomially larger than tw. The preprint [Cla26] already notes that its algorithm beats every fixed tree of cuts on expanders; the inference statements and both lower bounds are new. The ETH-based bound of [KB09, KBvdG10] has lost its proof. Part (1) puts a bound of the form $2^{\mathrm{tw}/\mathrm{polylog}\,\mathrm{tw}}$ on the surviving hypothesis $\mathrm{ETH}^{\mathrm{len}}_C$, and the qualitative statement that inference is not polynomial in the treewidth rests on NP not having polynomial-size circuits [CSH08, CC16].

By-products for fine-grained complexity. Three results do not concern learning.

- Exact Max-SAT with cn clauses takes time $2^{O((1+c)n/\sqrt{\log n})}$, so the Sparse Max-SAT Hypotheses of Alman and Vassilevska Williams [AV20] are false (Corollary 4.5). This also follows from [Ope26a, Theorem 1.1] with a threshold circuit; the error profile gives in addition the whole distribution of the number of violated clauses.

- Under CNF-SETH, a $\#k$-SAT algorithm with exponent $k^\mu n/(\log n)^\nu$ needs $\mu \geq \nu$ (Corollary 3.7). A better rate must come with a worse dependence on k.
- CNF-SAT with linearly many clauses of arbitrary width is subexponential (Proposition 6.1). So the hard instances of CNF-SETH have superlinearly many clauses, and the hardness of low-dimensional Orthogonal Vectors no longer follows from a hypothesis about SAT.

What survives. Section 6 takes stock of the conditional lower bounds of learning theory. Bounds stated with polynomial slack, such as the hardness of computing the VC dimension [MR17b], survive under Almost-ETH [Man26], and so under $\mathrm{ETH}_C^{\mathrm{len}}$. Bounds from near-linear reductions survive under $\mathrm{ETH}_C^{\mathrm{len}}$ with a polylogarithmic loss in the exponent. Bounds whose exponent is only $\Theta(\log\log N)$, such as those for proper learning of decision trees [KST23b] and for partial MCSP [Ila20], keep almost nothing. Bounds that rest on CNF-SETH or on k-Clique survive verbatim. Among them is the $n^{k-o(1)}$ bound for juntas, the canonical non-local class. The quadratic SETH-based bounds for attention and kernel density estimation are not contradicted, but they now rest on the low-dimensional OV hypothesis alone. The next obstacle is permutations. On difference vectors $e_i - e_j$, ERM for real-valued halfspaces is weighted Feedback Arc Set (Proposition 8.1), and the new algorithms do not reach it.

What is new. The two preprints [Ope26a, Cla26] contain no statement about learning. Lemma 3.1 is the encoding of [Ope26a, Corollary C.2], and the upper bound for counting at constant locality is that corollary. Everything else is new: the locality principle with bounds uniform in κ, the equivalence with κ-SAT and the hardness at logarithmic locality, the error-profile algorithms for ERM, agnostic and private learning, the lower bounds at constant and inverse-polynomial accuracy, the results on inference, the by-products above, and the ledger.

1.2 Techniques

Version spaces as CNFs. Each example excludes at most 2^κ patterns of its κ relevant bits. So the version space is the solution set of a κ-CNF with at most $m2^\kappa$ clauses (Lemma 3.1). The algorithm of [Cla26] counts its solutions in time $2^{O(\kappa^2 N \log\log N/\sqrt{\log N})}$ independently of m, and sampling, halving and learning follow by self-reducibility. For the converse, a padding gadget turns a clause into a sparse example. Its inner product with a sign vector is $2t-2$, where t is the number of true literals (Lemma 3.3).

The error profile as one contraction. Agnostic learning needs the number of hypotheses with exactly j errors for every j. An example contributes the factor 1 when it is fit and V when it is not, and the contraction theorem of [Ope26a] over $(\mathbb{Z}/2^P\mathbb{Z})[V]/(V^\ell)$ computes the product sum. Three choices make it exact. We restrict to the κm description bits that the sample reads. We turn the factor graph into a bipartite Boolean tensor network of degree $\max\{\kappa, 3\}$ (Lemma B.1). And we choose the precision P and the truncation order ℓ so that no coefficient is lost. The cost depends on the total size κm of the sample instead of $\kappa^2 N$, which gives sample-sensitive learning, the exponential mechanism, and exact sparse Max-CSP.

Lower bounds that control variables, length and accuracy. For consistency, a quantitative form of Schuler's width reduction [Sch05, CIP06] shows that, under CNF-SETH, k-SAT with $k = K\log n$ needs time $2^{(1-\gamma)n}$, with solution counts preserved (Lemma 3.5). For constant accuracy, a quasi-linear PCP turns a CNF of length L into an LD_3 sample in dimension $L\mathrm{polylog}L$, with

perfect completeness and a constant gap (Lemma 4.8). For inference, OR-chains with copied variables turn any CNF into a degree-3 factor graph whose pathwidth is at most the number of variables plus 2 (Lemma 5.4).

1.3 Organization

Section 2 fixes the model, imports the new algorithms, builds the engine of all our algorithms, and states the surviving hypotheses. Sections 3 to 5 prove the three transitions. Section 6 takes stock of the conditional lower bounds of learning theory. Section 7 discusses related work, and Section 8 isolates the permutation obstacle and lists open problems. The main text gives a proof sketch for each result. Section A explains the appendix, which contains all full proofs in a structured form.

AI disclosure. This paper was written with Claude Opus 5.5 (Anthropic), working as an agent in Claude Code. The author posed the question and directed the presentation. Claude chose the results, proved them, wrote the text and drew the figures. Separate Claude agents searched the literature, audited the proofs and reviewed the paper as referees, and the paper was revised to address their findings. The gadgets and reductions were tested by brute force on small random instances, and the asymptotic estimates were checked symbolically and in exact arithmetic. The author is responsible for the content.

2 Preliminaries

This section fixes the model, states the new algorithms, builds the computational engine that the rest of the paper uses, and lists the hypotheses that survive.

2.1 Notation and the learning model

We write log for $\log_2$ and $[n]$ for $\{1, \dots, n\}$. For $w \in \{0,1\}^N$ and $R \subseteq [N]$, the string $w|_R$ is the restriction of w to R. A *clause* is a nonempty set of literals, and its *width* is its size. A *CNF* is a finite set of clauses, and a *k-CNF* has only clauses of width at most k. The *length* L of a CNF is its number of literal occurrences, $L = \sum_C |C|$. For a CNF F on declared variables $z_1, \dots, z_n$, the number $\#F$ counts the satisfying assignments in $\{0,1\}^n$, and $\mathsf{SAT}(F)$ is the set of these assignments. We measure time in the word-RAM model, and "polynomial" means polynomial in the input length.

A *sample* is a sequence $S = ((x_1, y_1), \dots, (x_m, y_m))$ of labeled examples. For a class $\mathcal{H} = \{h_w : w \in \{0,1\}^N\}$, the *version space* of S is $\mathrm{VS}(S) = \{w : h_w(x_i) = y_i \text{ for all } i\}$, and the *empirical error* of w is $\mathrm{err}_S(w) = |\{i : h_w(x_i) \neq y_i\}|/m$. The *error profile* of S is the generating polynomial of the error counts,

$$E_S(V) = \sum_{w \in \{0,1\}^N} V^{\,m \cdot \mathrm{err}_S(w)} = \sum_{j=0}^{m} c_j V^j,$$

so the coefficient c_j is the number of descriptions that misclassify exactly j examples. For a distribution $\mathcal{D}$ on labeled examples, $\mathrm{err}_{\mathcal{D}}(w) = \Pr_{(x,y)\sim\mathcal{D}}[h_w(x) \neq y]$ and $\mathrm{OPT} = \min_w \mathrm{err}_{\mathcal{D}}(w)$. A *proper* learner outputs a description w. It is an (ε, δ)-*agnostic learner* if $\mathrm{err}_{\mathcal{D}}(w) \leq \mathrm{OPT} + \varepsilon$ with probability at least $1 - \delta$, and an (ε, δ)-*PAC learner* if this holds whenever $\mathrm{OPT} = 0$. We use three standard sample sizes.

Imported Theorem 2.1 (Sample sizes; see [BEHW87, BEHW89, SSBD14])**.** *Let $\mathcal{H}$ have description length N and VC dimension $d \geq 1$. There are absolute constants $C_0, C_B \geq 1$ such that the following hold with probability at least $1 - \delta$ over a sample of m independent examples from $\mathcal{D}$.*

(a) Occam. *If* $\mathrm{OPT} = 0$ *and* $m \geq m_{\mathrm{Occ}} := (N \ln 2 + \ln(1/\delta))/\varepsilon$, *then every* $w \in \mathrm{VS}(S)$ *has* $\mathrm{err}_{\mathcal{D}}(w) \leq \varepsilon$.

(b) Realizable VC bound. *If* $\mathrm{OPT} = 0$ *and* $m \geq m_{\mathrm{VC}} := C_B(d \log(2/\varepsilon) + \log(2/\delta))/\varepsilon$, *then every* $w \in \mathrm{VS}(S)$ *has* $\mathrm{err}_{\mathcal{D}}(w) \leq \varepsilon$.

(c) Uniform convergence. *If* $m \geq m_{\mathrm{UC}} := C_0(d + \ln(1/\delta))/\varepsilon^2$, *then every* w *has* $|\mathrm{err}_S(w) - \mathrm{err}_{\mathcal{D}}(w)| \leq \varepsilon/2$. *Thus every minimizer of* err_S *is* ε*-optimal.*

2.2 Local hypothesis classes

Definition 2.2 (Local class)**.** Let $N, \kappa \geq 1$. A *κ-local class* $\mathcal{H}$ on a domain $\mathcal{X}$ assigns to every $x \in \mathcal{X}$ a set $R(x) \subseteq [N]$ with $|R(x)| \leq \kappa$ and a predicate $f_x : \{0,1\}^{R(x)} \to \{0,1\}$, both computable in time polynomial in N and the length of x. A *description* $w \in \{0,1\}^N$ defines the hypothesis $h_w(x) = f_x(w|_{R(x)})$, and $\mathcal{H} = \{h_w : w \in \{0,1\}^N\}$. We call N the *description length* and κ the *locality*.

Because $|\mathcal{H}| \leq 2^N$, the VC dimension of a class with description length N is at most N. The following classes are local on sparse data. In each of them, $R(x)$ is determined by the support of x, and a sign vector $\sigma \in \{\pm 1\}^n$ is described by the bits $w_i = (1 + \sigma_i)/2$.

Example 2.3 (Models on sparse data)**.** Let $\mathcal{X}_k = \{x \in \{-1, 0, 1\}^n : |\mathrm{supp}(x)| \leq k\}$.

(a) *Literal disjunctions* LD_k on $\mathcal{X}_k$: $h_\sigma(x) = 1$ if and only if $x_i = \sigma_i$ for some $i \in \mathrm{supp}(x)$. Here $N = n$ and $\kappa = k$. A sample of positive examples is a k-CNF: the example x is the clause $\bigvee_{i \in \mathrm{supp}(x)} (\sigma_i = x_i)$.

(b) *Binarized perceptrons* BP_k on $\mathcal{X}_k$: $h_\sigma(x) = \mathbf{1}[\langle \sigma, x \rangle \geq 0]$. Here $N = n$ and $\kappa = k$.

(c) *b-bit linear threshold models* on k-sparse integer data, with weights in $\{-2^{b-1}, \ldots, 2^{b-1} - 1\}$ and a fixed threshold. Here $N = bn$ and $\kappa = bk$.

(d) *Quantized embedding models.* Each token of a vocabulary $[n]$ has an embedding in $\{0,1\}^b$. An input is a sequence of at most k tokens, and the output is a fixed polynomial-time function of their embeddings. Here $N = bn$ and $\kappa = bk$. Factorization machines with binary factors are a special case.

(e) *Sparse parities*: $h_w(x) = \langle w, x \rangle \bmod 2$ on $\{x \in \{0,1\}^n : |x| \leq k\}$. Here $N = n$ and $\kappa = k$.

(f) *Two-layer binarized networks* with r hidden units on $\mathcal{X}_k$: $h(x) = \mathbf{1}[\sum_{j \leq r} a_j \, \mathrm{sgn}(\langle u_j, x \rangle) \geq 0]$ with $u_j \in \{\pm 1\}^n$, $a \in \{\pm 1\}^r$ and $\mathrm{sgn}(0) = 1$. Here $N = rn + r$ and $\kappa = rk + r$.

2.3 The new algorithms

We use the two refutations of ETH only through the following three statements. The first counts the solutions of k-CNFs for every k. The second does so in polynomial space. The third contracts sparse tensor networks over a truncated polynomial ring.

Imported Theorem 2.4 ([Cla26, Main Theorem and Corollary 11])**.** *There are an absolute constant* C_1 *and a deterministic algorithm with the following property. Given a* k*-CNF, for any* $k \geq 1$, *with* $M \geq 2$ *clauses in which* $n \geq 4$ *variables occur, the algorithm computes the number of satisfying assignments of these* n *variables in time at most*

$$\min\left\{2^{C_1 kM/\sqrt{\log M}},\ 2^{C_1 k^2 n \log\log n/\sqrt{\log n}}\right\}.$$

Imported Theorem 2.5 ([Ope26a, Theorem 1.1])**.** *For each fixed* $k \geq 2$ *there is a deterministic algorithm that, given a* k*-CNF* F *on* $n \geq 20$ *declared variables with* M *clauses, computes* $\#F$ *in* $(n+M)^{O_k(1)} 2^{O_k(n \log\log n/\sqrt{\log n})}$ *bit operations and* $(n+M)^{O_k(1)}$ *bits of space.*

A *Boolean tensor network* is a bipartite multigraph G in which every edge carries one bit. Every vertex v has an ordered list ∂v of incident edges and a *tensor* ψ_v, which assigns a ring element to each Boolean assignment of ∂v. The *contraction* of G is $\sum_{x\in\{0,1\}^{E(G)}} \prod_v \psi_v(x|_{\partial v})$. For integers $P, \ell \geq 1$ let

$$\mathcal{B}_{P,\ell} = (\mathbb{Z}/2^P\mathbb{Z})[V]/(V^\ell).$$

Imported Theorem 2.6 ([Ope26a, Theorem D.1], specialized)**.** *For every polynomial p there is a constant c_p with the following property. Let $s \geq 2$, and let G be a bipartite Boolean tensor network with at most s edges, at most s vertices, maximum degree at most $\lceil s^{9/10} \rceil$, and tensor entries in $\mathcal{B}_{P,\ell}$ with $P, \ell \leq s$. Let $|G|$ be the bit length of the description of G, and suppose that each entry can be computed from it in time $p(s + |G|)$. Then the contraction of G can be computed deterministically in $(s + |G|)^{c_p} \cdot 2^{c_p s/\sqrt{\log s}}$ bit operations and $(s + |G|)^{c_p}$ bits of space.*

Theorem D.1 of [Ope26a] allows $O(s)$ edges, polynomially many vertices and more expensive entries; we use only the special case above. The bound of Imported Theorem 2.4 is checked in Lean [Cla26]. The Lean formalization of [Ope26a] certifies Theorem 1.1 with the weaker rate $(\log\log n)^2/\sqrt{\log n}$ and does not cover Theorem D.1 [Ope26b]. Every result in this paper that uses these statements holds verbatim with any improved rate. If #k-SAT on n variables takes time $2^{\rho(k,n)\,n}$, then every exponent $\kappa^2 N \log\log N/\sqrt{\log N}$ below becomes $\rho(\kappa, N)\,N$. The same holds for the rate $1/\sqrt{\log s}$ of Imported Theorem 2.6.

2.4 The engine: sparse sum-products and error profiles

A *factor graph* Φ over a commutative ring $\mathcal{B}$ consists of n Boolean variables and a finite family $\mathcal{F}$ of *factors*. Each factor f has a *scope* $I_f \subseteq [n]$ and a *table* $\psi_f : \{0,1\}^{I_f} \to \mathcal{B}$. The *partition function* of Φ is

$$Z(\Phi) = \sum_{x\in\{0,1\}^n} \prod_{f\in\mathcal{F}} \psi_f(x|_{I_f}).$$

The *arity* of Φ is $\max_f |I_f|$, its *size* is $\|\Phi\| = \sum_f |I_f|$, and the *degree* of a variable is the number of scopes that contain it. We write $|\Phi|$ for the bit length of the description of Φ. The next theorem is the engine of all our algorithms. It turns Imported Theorem 2.6 into a statement about factor graphs.

Theorem 2.7 (Sparse sum-product)**.** *Let Φ be a factor graph over $\mathcal{B}_{P,\ell}$ of arity at most a, whose table entries can be computed in polynomial time. Let s be an integer with $s \geq \max\{4\|\Phi\|, P, \ell, 2\}$ and $a \leq s^{9/10}$. Then $Z(\Phi)$ can be computed deterministically in time $\mathrm{poly}(s + |\Phi|) \cdot 2^{O(s/\sqrt{\log s})}$ and space $\mathrm{poly}(s + |\Phi|)$. The constants depend only on the polynomial that bounds the time to evaluate one table entry.*

Proof sketch. Factors with empty scope are constants, and we multiply them in at the end. Every other factor becomes a vertex of a tensor network. Every variable becomes a tree of equality tensors of degree at most 3, whose leaves are the occurrences of the variable. A subdivision of every edge makes the network bipartite. The result has at most $4\|\Phi\|$ edges and vertices, maximum degree $\max\{a, 3\}$, and the same contraction up to a factor 2 for each variable in no scope. Imported Theorem 2.6 then applies. Section B gives the details. □

We use two rings. In the *counting ring* $\mathcal{B}_{P,1} = \mathbb{Z}/2^P\mathbb{Z}$, a partition function whose value is a nonnegative integer below 2^P is computed exactly. In the *profile ring* $\mathcal{B}_{P,\ell}$, tables with entries in $\{1, V\}$ make $Z(\Phi)$ a generating function. If ℓ exceeds the number of factors, the coefficient of V^j counts the assignments that pick the entry V in exactly j factors. Theorem D.1 of [Ope26a]

allows truncation orders $\ell > 1$, and its proof uses them. With entries in $\{1, V\}$, a truncation order larger than the number of factors records every value of a sparse objective at once, and this turns counting into optimization. Counting assignments by their weight goes back to Williams' algorithm for Max-2-CSP [Wil05]. The first use is the error profile of a sample.

Lemma 2.8 (Exact error profiles)**.** *Let $\mathcal{H}$ be κ-local with description length N, let S be a sample of size m, and let $s = \max\{4\kappa m,\, m+1,\, \kappa^2\}$. The error profile E_S can be computed exactly and deterministically in time $\mathrm{poly}(N, m) \cdot 2^{O(s/\sqrt{\log s})}$. The same holds for the error profile of the descriptions that extend any given partial assignment.*

Proof sketch. Only the set $R(S) = \bigcup_i R(x_i)$ of description bits matters, and $|R(S)| \le \kappa m$. Build a factor graph on the variables $R(S)$ over $\mathcal{B}_{P,\ell}$ with $P = |R(S)| + 1$ and $\ell = m + 1$. Example i becomes a factor with scope $R(x_i)$ and with table 1 where f_{x_i} agrees with y_i and V where it does not. An assignment with j errors contributes V^j, and $j \le m < \ell$, so nothing is truncated. No coefficient exceeds $2^{|R(S)|} < 2^P$. So the partition function is the profile of the restricted descriptions, and E_S is $2^{N-|R(S)|}$ times it. Theorem 2.7 applies with this s (Section B). □

2.5 Hypotheses that survive

The new algorithms need bounded clause width, or linear total size. The hypotheses below concern the opposite regime, and the new algorithms do not contradict Hypotheses 2.9 to 2.12. Corollary 4.5 refutes Hypothesis 2.13, and Hypothesis 2.14 is false for $C < 1/2$ and open for $C \ge 1/2$. We state all hypotheses for randomized algorithms with error probability at most $1/3$.

Hypothesis 2.9 (CNF-SETH)**.** For every $\gamma > 0$ there is $c > 0$ such that satisfiability of CNF formulas with n variables and at most n^c clauses, of arbitrary width, cannot be decided in time $O(2^{(1-\gamma)n})$.

Hypothesis 2.10 (#CNF-SETH)**.** The same statement for computing the number of satisfying assignments.

Hypothesis 2.11 (CNF-ETH)**.** There are $c, \rho > 0$ such that satisfiability of CNF formulas with n variables and at most n^c clauses cannot be decided in time $O(2^{\rho n})$.

Hypothesis 2.9 implies the form "no algorithm runs in time $2^{(1-\gamma)n}\mathrm{poly}(M)$" considered in [CDL+16] for instances with M clauses. Indeed, an algorithm of that form runs in time $O(2^{(1-\gamma/2)n})$ whenever $M \le n^c$. The converse implication is open, because the hard instances of the latter form might need superpolynomially many clauses. We need the polynomial bound, because the width reduction of Lemma 3.5 branches on clauses. Cygan et al. [CDL+16] already noted that SETH could fail while CNF-SAT stays hard; the refutation of ETH realizes the first half of this scenario. All three hypotheses are monotone in c, so we may assume $c \ge 1$. Hypothesis 2.9 implies Hypothesis 2.10, because counting decides satisfiability, and it implies Hypothesis 2.11 with $\rho = 1/2$.

Hypothesis 2.12 (Max-3-SAT Hypothesis; randomized form of [LVW18, Hypothesis 2.2])**.** For every $\gamma > 0$, the maximum number of simultaneously satisfiable clauses of a 3-CNF on n variables cannot be computed in time $O(2^{(1-\gamma)n})$.

Hypothesis 2.13 (Sparse Max-SAT Hypotheses [AV20, Hypotheses 3 and 4])**.** For every $\gamma > 0$ there is $c > 0$ such that Max-3-SAT on n variables and cn clauses cannot be solved in time $O(2^{(1-\gamma)n})$. For every $\gamma > 0$ there is $c > 0$ such that Max-2-SAT on n variables and cn clauses cannot be solved in time $O(2^{(\omega/3-\gamma)n})$, where ω is the exponent of matrix multiplication.

A 3-CNF on n variables has fewer than $8n^3$ clauses, so Hypothesis 2.12 concerns instances of polynomial size. Bringmann, Cassis, Fischer and Künnemann [BCFK22] base their fine-grained approximability classification on the Max-3-SAT hypotheses. The last hypothesis measures satisfiability by formula length, at the rate that the new algorithms make natural.

Hypothesis 2.14 (Length form of ETH at rate C)**.** For $C > 0$, the hypothesis $\mathrm{ETH}^{\mathrm{len}}_C$ states that no randomized algorithm with error probability at most $1/3$ decides satisfiability of every CNF formula of length L in time $2^{L/(\log L)^C}$, for all sufficiently large L.

A CNF of length L becomes an equisatisfiable 3-CNF with at most $2L$ variables and at most L clauses when we split long clauses with fresh variables. Imported Theorem 2.4 decides it in time $2^{3C_1 L/\sqrt{\log L}}$. So $\mathrm{ETH}^{\mathrm{len}}_C$ is false for $C < 1/2$ and open for $C \geq 1/2$. It states that the new algorithms are optimal up to polylogarithmic factors in the exponent. Before 2026, the fastest algorithms for SAT parameterized by length ran in time 2^{cL} with $c \approx 0.089$ [Hir00, Wah05, CL09, PX23], and ETH made $2^{\Omega(L)}$ the truth. The form of the hypothesis is not new. Stearns and Hunt [SH90] measured SAT by the length of the instance. Chen, Rothblum, Tell and Yogev [CRTY23] studied the hypothesis that 3-SAT has no randomized algorithm that runs in time $2^{n/\mathrm{polylog} n}$ in the input length n. The hypothesis $\mathrm{ETH}^{\mathrm{len}}_C$ is the instance of this form with an explicit rate, calibrated so that the new algorithms refute it exactly for $C < 1/2$.

3 The locality transition

This section proves the first transition. Consistency for κ-local classes is κ-SAT. So it is subexponential for slowly growing κ, and as hard as CNF-SAT at $\kappa = \Theta(\log N)$. Throughout, $\mathcal{H}$ is a κ-local class with description length N. Let $g(n) = n \log\log n/\sqrt{\log n}$ for $n \geq 16$ and $g(n) = 1$ for $n < 16$, which is nondecreasing, and let

$$T_\kappa(N) = \min\{2^N,\ 2^{C_1 \kappa^2 g(N)}\}$$

with C_1 from Imported Theorem 2.4.

3.1 Version spaces are κ-CNFs

Lemma 3.1 (Version spaces are κ-CNFs)**.** *Given a sample S of size m, one can compute in time $\mathrm{poly}(N, m) \cdot 2^\kappa$ a κ-CNF Γ_S on the variables $w_1, \ldots, w_N$ with at most $m2^\kappa$ clauses such that $\mathrm{VS}(S) = \mathsf{SAT}(\Gamma_S)$.*

Proof. An example (x, y) rules out exactly the patterns $\alpha \in \{0,1\}^{R(x)}$ with $f_x(\alpha) \neq y$. For each such pattern, let $C_{x,\alpha}$ be the clause on $\{w_i : i \in R(x)\}$ that is false exactly when $w|_{R(x)} = \alpha$. Let Γ_S be the set of all clauses $C_{x_i,\alpha}$. Then w satisfies Γ_S if and only if $f_{x_i}(w|_{R(x_i)}) = y_i$ for every i. Each clause has width at most κ, and Γ_S takes $m2^\kappa$ evaluations of predicates. The only exception is an example with $R(x) = \emptyset$ whose constant predicate disagrees with its label. Then $\mathrm{VS}(S) = \emptyset$, and we output the unsatisfiable CNF $\{(w_1), (\neg w_1)\}$. □

The sample defines the map $w \mapsto (h_w(x_i))_{i \leq m}$. Each output bit of this map reads at most κ input bits, and the version space is the preimage of the label vector. Lemma 3.1 is the encoding that [Ope26a, Corollary C.2] uses to count, find and sample preimages of maps with constant output locality. For constant κ, parts (a)–(c) of the next theorem are these statements in learning terms.

3.2 The realizable transfer theorem

Theorem 3.2 (Realizable transfer)**.** *Let $\mathcal{H}$ be κ-local with description length N, and let S be a sample of size m.*

(a) Counting. *The number $|\mathrm{VS}(S)|$ can be computed deterministically in time $\mathrm{poly}(N, m) \cdot 2^{\kappa} \cdot T_{\kappa}(N)$.*

(b) Sampling. *If $\mathrm{VS}(S) \neq \emptyset$, a uniformly random element of $\mathrm{VS}(S)$ can be generated exactly in the same time, given uniform random integers in $\{0, \ldots, M-1\}$ at cost $\mathrm{poly}(\log M)$. With fair coins only, the same bound holds for the expected time of a Las Vegas algorithm.*

(c) Proper PAC learning. *For all $\varepsilon, \delta \in (0,1)$, the class $\mathcal{H}$ is properly (ε, δ)-PAC learnable from $\lceil m_{\mathrm{Occ}} \rceil$ examples by an algorithm that is deterministic after sampling and runs in the same time.*

(d) Online learning. *In the realizable mistake-bound model, the halving algorithm makes at most N mistakes, and round t takes time $\mathrm{poly}(N, t) \cdot 2^{\kappa} \cdot T_{\kappa}(N)$.*

All four run in time $\mathrm{poly}(N, m) \cdot 2^{o(N)}$ when $\kappa = o((\log N)^{1/4}/\sqrt{\log\log N})$.

Proof sketch. Lemma 3.1 turns the version space into a κ-CNF. Imported Theorem 2.4 counts its solutions in time $2^{C_1 \kappa^2 g(N)}$ independently of the number of clauses, and brute force gives the bound 2^N. Sampling follows from counting by self-reducibility [JVV86]: fix the bits one at a time, each with probability proportional to the number of solutions that extend the choice. Learning follows from the Occam bound, and halving from counting the two possible version spaces. Section C gives the details. □

The threshold $(\log N)^{1/4}/\sqrt{\log\log N}$ is the range in which the bound of [Cla26] is subexponential [Cla26, Appendix D]. The bound is uniform in κ because its constant does not depend on k.

Counting the version space is the computational core of several classical learners. These include the halving algorithm [BF72, Lit88] and query-by-committee [SOS92, FSST97], which needs samples from the version space. They also include the Gibbs algorithm, which predicts with a uniformly random consistent hypothesis. Part (a) also gives the Bayes-optimal prediction under the uniform prior, which is the halving prediction. For local classes all of these become subexponential. For constant $\kappa \geq 2$, Imported Theorem 2.5 gives the same conclusions in polynomial space, with $2^{O_\kappa(N \log\log N/\sqrt{\log N})}$ in place of $T_\kappa(N)$.

Approximate counting suffices for halving [GRS93], and hit-and-run sampling makes query-by-committee efficient on convex version spaces [GBNT05]. For local classes, approximation does not help: any approximate count decides whether the version space is empty, which is κ-SAT by Theorem 3.4; see also [HMP04]. So Theorem 3.2 is optimal up to the rate of SAT. Gibbs learning of binary perceptrons is a classical model of statistical physics [Gyö90, SST92], in which the data are random and dense. When there are few examples per weight, efficient algorithms exist there [ALS22]. SAT and #SAT encodings of binarized networks also serve to verify them, by counting inputs rather than weights [NKR$^+$18, BSS$^+$19], and to train them exactly [TIC$^+$19].

3.3 Learning is equivalent to bounded-width SAT

The transfer theorem is tight in a strong sense: proper learning of local classes is fine-grained equivalent to bounded-width SAT. Lemma 3.1 gives one direction. A padding gadget gives the other.

Lemma 3.3 (Perceptron gadget)**.** *Let $k \geq 1$ and $p = \max\{1, k-2\}$. Given a k-CNF F on $z_1, \ldots, z_N$ with M clauses, none of them tautological, one can compute in polynomial time a sample S_F of $M + p$ examples in $\{-1, 0, 1\}^{N+p}$, all labeled 1 and each with support of size at most $\max\{2k-2, 2\}$, with the following property. A sign vector σ is consistent with S_F as a binarized perceptron if and only if $\sigma_{N+1} = \cdots = \sigma_{N+p} = 1$ and the assignment $z_i = \mathbf{1}[\sigma_i = 1]$ satisfies F. In particular, $|\mathrm{VS}_{\mathsf{BP}}(S_F)| = \#F$.*

The smallest case shows how the gadget works. Take $k = 3$ and the clause $C = z_1 \vee \neg z_2 \vee z_3$. There is one padding coordinate, and the example of C is $x_C = e_1 - e_2 + e_3 + e_{N+1}$. A pinning example e_{N+1} forces $\sigma_{N+1} = 1$. Then $\langle \sigma, x_C \rangle = 2t - 2$, where t is the number of literals of C that the assignment makes true. So x_C is classified positive exactly when C is satisfied. In general, a clause of width $r \geq 2$ uses $r - 2$ padding coordinates with entry $+1$, a unit clause uses one padding coordinate with entry -1, and the same identity holds (Section C).

Theorem 3.4 (Locality equivalence)**.** *Let $\kappa, k \geq 1$, and let T and T' be nondecreasing time bounds.*

(i) If #κ-SAT on n variables with M clauses can be solved in time $T(n, M)$, then for every κ-local class the version space of a sample of size m can be counted in time $T(N, m2^\kappa) + \mathrm{poly}(N, m) \cdot 2^\kappa$.

(ii) If version spaces of BP_{2k} samples of size m in dimension n can be counted (respectively, decided nonempty) in time $T'(n, m)$, then #k-SAT (respectively, k-SAT) on N variables with M clauses can be solved in time $T'(N + k, M + k) + \mathrm{poly}(N, M)$.

(iii) If BP_{2k} in dimension n has a proper $(\varepsilon, 1/3)$-PAC learner that runs in time $T'(n, 1/\varepsilon)$, then k-SAT on N variables with M clauses has a randomized algorithm with one-sided error that runs in time $T'(N + k, 2(M + k)) \cdot \mathrm{poly}(N, M)$.

Proof sketch. Part (i) is Lemma 3.1. For (ii), delete tautological clauses and apply Lemma 3.3. For (iii), run the learner on the uniform distribution over the $M + p$ examples of S_F with accuracy $\varepsilon = 1/(2(M + k))$. All errors are multiples of $1/(M + p) > \varepsilon$, so an ε-accurate output is consistent with every example and encodes a satisfying assignment, which we check (Section C). □

Up to a factor 2 in the locality and an additive k in the dimension, the exponential-time complexity of consistency and of proper learning to inverse-polynomial accuracy for κ-local classes *is* that of κ-SAT. So any improvement for k-SAT, in the rate or in the dependence on k, transfers verbatim to Theorem 3.2, and any improvement for learning transfers back to k-SAT.

3.4 Hardness at logarithmic locality

Bounded-width SAT is subexponential, but SAT with clauses of width $\Theta(\log n)$ is as hard as CNF-SAT with polynomially many clauses. The tool is the width reduction of Schuler [Sch05], in the form of Calabro, Impagliazzo and Paturi [CIP06] that preserves solution counts. We need it with explicit constants.

Lemma 3.5 (Width-reduction transfer)**.** *Let $\rho \in (0, 1]$, $c \geq 1$, $K_0 = 2(c + 1)/\rho$ and $K' = 2K_0$. Suppose that k-SAT (respectively, #k-SAT) on n variables with $k \leq K' \log n$ and at most n^{2c} clauses can be solved in time $O(2^{\rho n})$ with error probability at most $1/3$. Then CNF-SAT (respectively, #CNF-SAT) on n variables with at most n^c clauses can be solved in time $O(2^{\rho n}\mathrm{poly}(n))$ with error probability at most $1/3$.*

Proof sketch. Let $k = \lfloor K_0 \log n \rfloor$. Branch on a clause wider than k: either replace it by its first k literals, or set these k literals to false. The leaves partition the solutions [CIP06]. Let $M \leq n^c$ be the number of clauses. A leaf reached with r branches of the second kind has at most $n - kr$

free variables, and at most $\binom{M+r}{r}$ leaves have this form. Solving each leaf in time $2^{\rho(n-kr)}$ costs $\sum_r \binom{M+r}{r} 2^{\rho(n-kr)} \le 2^{\rho n}(1+2^{-\rho k})^{M+n/k}$, which is $O(2^{\rho n})$ for this choice of k. Leaves with few variables are solved by brute force (Section C.3). □

Corollary 3.6. *(a) Under Hypothesis 2.9, for every $\gamma \in (0,1)$ there are K', c' such that k-SAT with $k \le K' \log n$ cannot be solved in time $O(2^{(1-\gamma)n})$, even on instances with at most $n^{c'}$ clauses. The same holds for counting under Hypothesis 2.10.*

(b) Under Hypothesis 2.11, there are $K', c', \rho > 0$ such that k-SAT with $k \le K' \log n$ and at most $n^{c'}$ clauses cannot be solved in time $O(2^{\rho n})$.

The same corollary limits how a future k-SAT algorithm can trade the dependence on k against the rate. No such limit was known for $2^{o(n)}$ algorithms; earlier results concern k-dependence of $2^{(1-\Omega(1/k))n}$ algorithms [IP01, VW21].

Corollary 3.7 (The k-dependence of k-SAT algorithms)**.** *Assume Hypothesis 2.9. Let $\mu, \nu \ge 0$ and $C' > 0$ be constants. Suppose that a single algorithm, given k and a k-CNF on n variables with at most n^c clauses, runs in time $2^{C' k^\mu n (\log\log n)^{O(1)}/(\log n)^\nu}$ for every constant c. Then $\mu \ge \nu$.*

Proof. Take K' and c' from Corollary 3.6(a) for $\gamma = 1/2$. For $k = \lfloor K' \log n \rfloor$ and instances with at most $n^{c'}$ clauses, the time is at most $2^{C' K'^\mu n (\log n)^{\mu-\nu} (\log\log n)^{O(1)}}$. If $\mu < \nu$, this is $O(2^{n/2})$, which contradicts Corollary 3.6(a). □

The bound of [Cla26] has $\mu = 2$ and $\nu = 1/2$. So a k-SAT algorithm with rate $1/\log n$, that is $\nu = 1$, must have exponent at least linear in k. In terms of learning, an algorithm with exponent $k^\mu n/(\log n)^\nu$ makes learning subexponential for $\kappa \ll (\log N)^{\nu/\mu}$, and CNF-SETH caps this threshold at $\log N$.

Theorem 3.8 (Hardness at logarithmic locality)**.** *Assume Hypothesis 2.9. For every $\gamma \in (0,1)$ there are constants K and c_γ with the following properties. No randomized algorithm decides in time $O(2^{(1-\gamma)n})$ whether a sample of size at most n^{c_γ} in dimension n is consistent with some binarized perceptron on $\lceil K \log n \rceil$-sparse data. No proper $(n^{-c_\gamma}, 1/3)$-PAC learner for $\mathsf{BP}_{\lceil K \log n \rceil}$ runs in time $O(2^{(1-\gamma)n})$. Under Hypothesis 2.10, the same holds for counting version spaces. Under Hypothesis 2.11, there are $K, \rho > 0$ with the same statements for time $O(2^{\rho n})$.*

Proof sketch. Take a k-CNF with $k = \lfloor K' \log N \rfloor$ from Corollary 3.6, and apply Lemma 3.3. The sample has dimension $n \le N + k$ and sparsity $2k \le K \log n$. An algorithm in time $O(2^{(1-\gamma)n})$ would solve the formula in time $O(2^{(1-\gamma/2)N})$ (Section C). □

Corollary 3.9 (The data-sparsity transition)**.** *Consider binarized perceptrons on k-sparse data in dimension n, with samples of polynomial size.*

(a) If $k = o((\log n)^{1/4}/\sqrt{\log\log n})$, then consistency, version-space counting, uniform sampling, and proper PAC learning at any accuracy take time $\mathrm{poly}(m) \cdot 2^{o(n)}$.

(b) Under Hypothesis 2.9, for every $\gamma \in (0,1)$ there are K and c_γ such that for $k \ge K \log n$, consistency and proper PAC learning to accuracy n^{-c_γ} cannot be done in time $O(2^{(1-\gamma)n})$. Under Hypothesis 2.10, the same holds for version-space counting.

The same dichotomy holds for literal disjunctions LD_k. There, a sample of positive examples is literally a k-CNF, and (b) follows from Corollary 3.6 without a gadget.

Part (a) holds for every class of Example 2.3, with k replaced by the locality. Part (b) does not. Sparse parities are k-local, but their consistency problem is linear algebra over $\mathbb{F}_2$ and takes polynomial time. Locality makes subexponential algorithms possible. Hardness at logarithmic locality needs predicates f_x that can encode clauses, as those of BP and LD can.

3.5 Constant accuracy moves the threshold to $\sqrt{\log N}$

Theorem 3.2 pays $\kappa^2 \log\log N$ in the exponent because it handles every sample, and the version space of an arbitrary sample is an arbitrary κ-CNF. A PAC learner needs only m_{VC} examples. The engine of Section 2.4 charges for the total size κm of the sample instead of the number of variables. At constant accuracy this trades the factor $\kappa \log\log N$ for a constant.

Theorem 3.10 (Sample-sensitive realizable learning)**.** *Let $\mathcal{H}$ be κ-local with description length N and VC dimension $d \geq 1$, and let $\varepsilon, \delta \in (0,1)$. With $m = \lceil m_{\mathrm{VC}} \rceil$ examples, $\mathcal{H}$ is properly (ε,δ)-PAC learnable by an algorithm that is deterministic after sampling and runs in time*

$$\mathrm{poly}(N,m) \cdot \exp\Big(O\Big(\frac{\kappa\,(d\log(2/\varepsilon)+\log(2/\delta))}{\varepsilon\sqrt{\log(2d/\varepsilon)}} + \kappa^2\Big)\Big).$$

For constant ε and δ the time is $2^{O(\kappa N/\sqrt{\log N}+\kappa^2)}$, which is $2^{o(N)}$ whenever $\kappa = o(\sqrt{\log N})$.

Proof sketch. Draw the sample. With probability one it is consistent with a target, so the coefficient of V^0 in its error profile is positive. Fix the bits that the sample reads one at a time, each to a value that keeps this coefficient positive (Lemma 2.8). The result is consistent, and Imported Theorem 2.1(b) bounds its error. The engine runs with $s = O(\kappa m + \kappa^2)$ and $\log s \geq \log(2d/\varepsilon)$ (Section C). □

So proper PAC learning takes time $2^{o(N)}$ whenever $\kappa\log(2/\varepsilon)/\varepsilon = o(\sqrt{\log N})$ and δ is constant. At inverse-polynomial accuracy, the window $(\log N)^{1/4} \lesssim \kappa \lesssim \log N$ is the window of k-SAT itself (Theorem 3.4). Its lower end is where the k^2 in the exponent of [Cla26] stops giving subexponential time. Its upper end is where width reduction makes CNF-SETH apply. At constant accuracy the subexponential range reaches $\kappa = o(\sqrt{\log N})$, and Section 4.3 shows that there, for constant κ, the exponent $N/\mathrm{polylog}N$ is the truth.

4 The accuracy transition

Counting is not enough for agnostic learning. An agnostic learner must find a hypothesis with the fewest errors, and a private learner must weigh every hypothesis by its error. Both tasks need the error profile, which the engine computes exactly (Lemma 2.8). This section turns the profile into learners. It then shows that the accuracy decides which form of SAT governs learning: SAT by formula length at constant accuracy, and Max-3-SAT at inverse-polynomial accuracy. Throughout, $\mathcal{H}$ is κ-local with description length N.

4.1 Exact ERM and proper agnostic learning

Theorem 4.1 (Exact ERM)**.** *For a κ-local class and any sample S of size m, a minimizer of err_S, the number of minimizers and the entire error profile can be computed in time $\mathrm{poly}(N,m)\cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\kappa m, m+1, \kappa^2\}$. In particular, for constant κ and $m = O(N)$, exact ERM takes time $2^{O(N/\sqrt{\log N})}$.*

Proof sketch. Lemma 2.8 gives the profile, and its lowest nonzero coefficient gives the minimum number j^* of errors. To find a minimizer, fix the bits that the sample reads one at a time, each to a value for which the profile of the restricted descriptions still has a nonzero coefficient at V^{j^*}. Such a value exists, because the profile is the sum of the profiles of the two restrictions (Section D). □

For 3-local classes the engine is optimal up to polylogarithmic factors in the exponent, already for consistency. The reason is that a CNF is a sample of positive examples of literal disjunctions.

Corollary 4.2 (Consistency is SAT by length)**.** *Assume* $\mathrm{ETH}_C^{\mathrm{len}}$*, and let* $\theta > 0$*. No randomized algorithm with error probability at most* $1/3$ *decides in time* $2^{m/(\log m)^{C+\theta}}$ *whether a sample of* m *positive examples of* LD_3 *is consistent. So for* 3*-local classes, Theorem 4.1 is optimal up to a factor* $(\log m)^{C-1/2+\theta}$ *in the exponent.*

Proof sketch. Split the clauses of a CNF of length L into clauses of width at most 3 with fresh variables. The result has at most L clauses, and each clause is a positive example of LD_3 (Example 2.3(a)). So $m \le L$, and the sample is consistent if and only if the CNF is satisfiable (Section D). □

Theorem 4.3 (Agnostic transfer)**.** *Let* $\mathcal{H}$ *be* κ*-local with description length* N *and VC dimension* d*, and let* $\varepsilon, \delta \in (0,1)$*. Exact ERM on* $m = \lceil m_{\mathrm{UC}} \rceil$ *examples is a proper* (ε,δ)*-agnostic learner, and it runs deterministically after sampling in time*

$$\mathrm{poly}(N,m) \cdot \exp\Big(O\Big(\frac{\kappa(d+\log(1/\delta))}{\varepsilon^2\sqrt{\log(2d/\varepsilon)}} + \kappa^2\Big)\Big).$$

If $\kappa/\varepsilon^2 = o(\sqrt{\log N})$ *and* δ *is constant, the time is* $2^{o(N)}$*.*

Proof sketch. By Imported Theorem 2.1(c), every minimizer of err_S is ε-optimal with probability at least $1-\delta$. Theorem 4.1 finds one with $s = O(\kappa m + \kappa^2)$, and $s \ge 4\kappa m \ge 2d/\varepsilon$ gives $\log s \ge \log(2d/\varepsilon)$ (Section D). □

This bound depends on the VC dimension through d/ε^2, not on N. For local classes whose descriptions are much longer than their VC dimension, the learner is polynomial in N and subexponential in d/ε^2.

The proof of Theorem 4.1 never uses that the constraints come from a hypothesis class. It solves sparse Max-CSP exactly, and this refutes a hypothesis from fine-grained approximability.

Corollary 4.4 (Sparse Max-CSP)**.** *Let* Φ *be a Boolean constraint system on* n *variables with* M *constraints of arity at most* k*. For every* j*, the number of assignments that violate exactly* j *constraints can be computed in time* $\mathrm{poly}(n,M)\cdot 2^{O(s/\sqrt{\log s})}$*, where* $s = \max\{4kM, n+1, M+1, k^2\}$*. In particular, for every constant* c*, Max-*3*-SAT with* n *variables and at most* cn *clauses can be solved in time* $2^{O((1+c)n/\sqrt{\log n})}$*. With positive integer clause weights at most* W*, the same holds with* $s = \max\{4kM, n+1, WM+1, k^2\}$*, and Max-*3*-SAT takes time* $2^{O((1+cW)n/\sqrt{\log n})}$*.*

Corollary 4.5. *Both Sparse Max-SAT Hypotheses of Alman and Vassilevska Williams (Hypothesis 2.13) are false.*

Proof. Let $c > 0$. By Corollary 4.4, Max-3-SAT and Max-2-SAT with cn clauses can be solved deterministically in time $2^{O((1+c)n/\sqrt{\log n})}$. This is $O(2^{n/2})$, and $1/2 < 1-\gamma$ and $1/2 < \omega/3 - \gamma$ for $\gamma = 1/10$, because $\omega \ge 2$. So for $\gamma = 1/10$ no c satisfies either requirement of Hypothesis 2.13. □

Before 2026, the best algorithms for Max-SAT with cn clauses saved a factor $2^{n/O(c^{1/3}\log^{7/3} c)}$ over brute force [ACW16], and Max-2-SAT took time $2^{\omega n/3}$ [Wil05]. Alman and Vassilevska Williams [AV20] posed Hypothesis 2.13 as a basis for lower bounds on OV graphs. Bringmann, Cassis, Fischer and Künnemann [BCFK22] showed that the sparse Max-3-SAT hypothesis implies all hardness results of their fine-grained approximability classification. Most of those results also

follow from hypotheses about clique detection in hypergraphs [BCFK22], which the new algorithms do not affect. Corollary 4.5 also follows from Theorem 1.1 of [Ope26a]. One applies it to a CNF encoding of a linear-size threshold circuit for the number of satisfied clauses, as in the proof of its Corollary 1.2. The error profile gives it directly, together with the number of optimal assignments and the whole distribution of the number of violated clauses.

4.2 The exact exponential mechanism

The profile also gives the exponential mechanism of McSherry and Talwar [MT07] exactly, and with it the generic private learner of Kasiviswanathan et al. [KLN+11].

Theorem 4.6 (Exact private learning)**.** *Let $\mathcal{H}$ be κ-local and $\varepsilon_{\mathrm{dp}} > 0$. On a sample S of size m, one can sample w with probability proportional to $\exp(-\varepsilon_{\mathrm{dp}}\, m\, \mathrm{err}_S(w)/2)$ in time $\mathrm{poly}(N, m) \cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\kappa m, m+1, \kappa^2\}$, given exact samples from explicitly given distributions on $\{0, \dots, m\}$. This is the exponential mechanism, and it is $\varepsilon_{\mathrm{dp}}$-differentially private. Without exact sampling, the output distribution is within statistical distance η of the mechanism for any $\eta \geq 2^{-\mathrm{poly}(N,m)}$, and the output is $(\varepsilon_{\mathrm{dp}}, (1 + e^{\varepsilon_{\mathrm{dp}}})\eta)$-differentially private. With $m = O((N + \log(1/\delta)) \max\{1/(\varepsilon\varepsilon_{\mathrm{dp}}), 1/\varepsilon^2\})$, the output is ε-optimal with probability at least $1 - \delta$ [KLN+11], and the time is, up to polynomial factors,*

$$\exp\Big(O\Big(\frac{\kappa\,(N + \log(1/\delta))}{\varepsilon \min\{\varepsilon, \varepsilon_{\mathrm{dp}}\}\sqrt{\log N}}\Big)\Big).$$

Proof sketch. Descriptions that agree on the bits read by the sample have the same score. So it suffices to sample an error count j with probability proportional to $c_j e^{-\varepsilon_{\mathrm{dp}} j/2}$, then a uniform description with exactly j errors, by self-reduction on the restricted profiles (Section D). □

So for local classes the exponential mechanism over the whole class, the canonical but inefficient generic private learner, runs in time $2^{o(N)}$ when $\kappa/(\varepsilon \min\{\varepsilon, \varepsilon_{\mathrm{dp}}\}) = o(\sqrt{\log N})$ and $\log(1/\delta) = O(N)$. Its output distribution is the Gibbs posterior at inverse temperature $\varepsilon_{\mathrm{dp}}/2$. So Theorem 4.6 is also exact posterior sampling over discrete weights, and one posterior sample is differentially private [WFS15]. Implementations of the exponential mechanism without the loss η are studied in [Ilv20, CKS20]. The time grows with the sample size, and private sample complexity can be much smaller than N [BNS13]. The polynomial-time barriers for private learning [DNR+09, Ull16, BZ16] are not affected by subexponential algorithms.

4.3 Constant accuracy is SAT by length

At constant accuracy the parameter that governs learning changes. Exact consistency is SAT parameterized by the number of variables (Theorem 3.4). Constant-accuracy learning is SAT parameterized by the *length* of the formula. In one direction the bridge is the Occam bound. In the other it is a probabilistically checkable proof of quasi-linear size.

Imported Theorem 4.7 (Short PCPs [Din07, BSS08])**.** *There are absolute constants q, β and c_V and a deterministic polynomial-time verifier with the following behavior on a CNF F of length $L \geq 2$. The verifier picks a seed τ uniformly from a set Ω_F of size at most $c_V L(\log L)^\beta$. From F and τ it computes a set Q_τ of at most q positions of a proof π of length at most $q|\Omega_F|$, and a predicate D_τ on these positions. It accepts if and only if $D_\tau(\pi|_{Q_\tau}) = 1$. If F is satisfiable, some proof is accepted for every seed. If F is unsatisfiable, every proof is rejected for at least half of the seeds.*

Dinur [Din07] proves SAT $\in \mathrm{PCP}_{1/2,1}[\log(n\,\mathrm{polylog}n), O(1)]$ by gap amplification of the PCPs of Ben-Sasson and Sudan [BSS08], where n is the input bit length. For a CNF of length L we have $n = O(L \log L)$, and β absorbs the difference. Her verifier reads $O(1)$ symbols over a constant alphabet. Reading their binary encodings gives the non-adaptive form above, and deleting unqueried positions bounds the proof length.

Lemma 4.8 (Short PCPs as LD_3 samples)**.** *Let $\varepsilon_0 = 1/(q2^{q+2})$ and $c_q = 2q2^q c_V$. There is a deterministic polynomial-time reduction that maps a CNF F of length $L \geq 2$ to a list S_F^{pcp} of positive examples of LD_3 in dimension $N \leq c_q L(\log L)^\beta$, with the following property. Let $\mathcal{D}_F$ be the uniform distribution over the entries of S_F^{pcp}. If F is satisfiable, then $\mathrm{OPT}(\mathcal{D}_F) = 0$. If F is unsatisfiable, then $\mathrm{OPT}(\mathcal{D}_F) \geq 2\varepsilon_0$.*

Proof sketch. Every rejected pattern of every seed becomes a clause of width at most q on the proof bits. A chain of fresh variables splits this clause into clauses of width 3, and each 3-clause becomes a positive example of LD_3 (Example 2.3(a)). A proof that the verifier always accepts extends to an assignment that satisfies every chain. If F is unsatisfiable, every assignment falsifies, for at least half of the seeds, some clause in the chain of that seed. Each seed contributes at most $q2^q$ entries, which gives the gap $2\varepsilon_0$ (Section D). □

Theorem 4.9 (Hardness at constant accuracy)**.** *If LD_3 has a proper realizable $(\varepsilon_0, 1/3)$-PAC learner that runs in time $T(N)$, with T nondecreasing, then satisfiability of CNF formulas of length L can be decided with one-sided error $1/3$ in time $T(c_q L(\log L)^\beta) \cdot \mathrm{polylog}(L) + \mathrm{poly}(L)$. Consequently, under $\mathrm{ETH}_C^{\mathrm{len}}$, no proper realizable $(\varepsilon_0, 1/3)$-PAC learner for LD_3, and so no proper $(\varepsilon_0, 1/3)$-agnostic learner, runs in time $2^{N/(\log N)^{C+\beta+1}}$.*

Proof sketch. Run the learner on $\mathcal{D}_F$ with a clock, and accept if and only if it outputs, in time, a hypothesis with error below $2\varepsilon_0$ on S_F^{pcp}. Lemma 4.8 makes this correct. A time bound $2^{N/(\log N)^{C+\beta+1}}$ at $N \leq c_q L(\log L)^\beta$ gives $2^{L/(\log L)^C}$ for SAT (Section D). □

For realizable learning the Occam bound gives the converse, so the two problems have the same complexity up to polylogarithmic factors in the exponent.

Corollary 4.10 (Constant accuracy is SAT by length)**.** *Fix a constant $\kappa \geq 3$, and let T and T' be nondecreasing.*

(a) If satisfiability of CNF formulas of length L can be decided in time $T(L)$ with error at most $1/3$, then every κ-local class is properly (ε, δ)-PAC learnable in time $O(N \log(N/\delta)) \cdot T(\kappa 2^\kappa m + N) + \mathrm{poly}(N, m)$, where $m = \lceil (N \ln 2 + \ln(2/\delta))/\varepsilon \rceil$.

(b) Conversely, a proper realizable $(\varepsilon_0, 1/3)$-PAC learner for LD_3 that runs in time $T'(N)$ decides satisfiability of CNF formulas of length L in time $T'(c_q L(\log L)^\beta) \cdot \mathrm{polylog}(L) + \mathrm{poly}(L)$.

With the new algorithms, (a) gives time $2^{O_{\kappa,\varepsilon,\delta}(N/\sqrt{\log N})}$. Under $\mathrm{ETH}_C^{\mathrm{len}}$, (b) gives the lower bound $2^{N/(\log N)^{C+\beta+1}}$ for every $\varepsilon \leq \varepsilon_0$. A SAT algorithm with exponent $L/f(L)$ yields learners with exponent $O(N/f(\Theta(N)))$, and a learner with exponent $N/f(N)$ yields SAT algorithms with exponent $O(L(\log L)^\beta / f(\Theta(L(\log L)^\beta)))$.

So at a constant accuracy $\varepsilon \leq \varepsilon_0$, the exponent of proper learning of LD_3, realizable or agnostic, lies between $N/(\log N)^{C+\beta+1}$ under $\mathrm{ETH}_C^{\mathrm{len}}$ (Theorem 4.9) and $O(N/\sqrt{\log N})$ (Theorem 4.3). Both bounds are $N/\mathrm{polylog}N$. The same reduction replaces Gap-ETH.

Corollary 4.11 (Gap-3-SAT under $\mathrm{ETH}_C^{\mathrm{len}}$)**.** *Assume $\mathrm{ETH}_C^{\mathrm{len}}$. No randomized algorithm with error probability at most $1/3$ runs in time $2^{n/(\log n)^{C+\beta+1}}$ and distinguishes satisfiable 3-CNF clause*

multisets from those in which every assignment violates at least a $2\varepsilon_0$ fraction of the clauses. Here n is the number of variables plus the number of clauses, and no bound on the number of occurrences of a variable is assumed.

Consider a lower bound proved from Gap-ETH by a reduction of size $O(n)$ that works for every constant gap and accepts such instances. Under $\mathrm{ETH}_C^{\mathrm{len}}$ it holds with exponent $n/(\log n)^{C+\beta+1}$ in place of $\Omega(n)$. A reduction with larger blowup loses correspondingly more. The route from ETH to gap hardness through nearly-linear PCPs is standard [DKM19, GKMR21, Man23, BMVY25]. Since the refutation, Manurangsi takes this route from Almost-ETH [Man26], which gives only bounds of the form $2^{n^{1-\gamma}}$. The point of Corollary 4.11 is that $\mathrm{ETH}_C^{\mathrm{len}}$ gives $2^{n/\mathrm{polylog}n}$, which matches the learners of Theorem 4.3.

4.4 Hardness at inverse-polynomial accuracy

At inverse-polynomial accuracy, proper agnostic learning is as hard as exact ERM on a sample of polynomial size. In that regime the Max-3-SAT Hypothesis applies, and the exponent jumps to $(1-o(1))N$.

Theorem 4.12 (Hardness at inverse-polynomial accuracy)**.** *Assume Hypothesis 2.12. For every $\gamma > 0$, no randomized proper $(\varepsilon, 1/3)$-agnostic learner for LD_3 in dimension n with $\varepsilon = 1/(8n^3)$ runs in time $O(2^{(1-\gamma)n})$.*

Proof sketch. Remove tautological clauses from a 3-CNF. Each of the $M < 8n^3$ remaining clauses becomes a positive example of LD_3, and the error of σ under the uniform distribution is the fraction of clauses that σ violates. All errors are multiples of $1/M > \varepsilon$, so an ε-optimal output is an optimal assignment (Section D). □

For 3-local classes, Theorem 4.1 solves exact ERM on samples of size $m = o(N\sqrt{\log N})$ in time $2^{o(N)}$. Under Hypothesis 2.12, exact ERM on samples of size $\Theta(N^3)$ needs time $2^{(1-o(1))N}$. Approximate ERM is unaffected, because one can subsample to size $O(N/\varepsilon^2)$. Exact optimization, however, becomes harder as the sample grows. This complements the phenomenon of Daniely, Linial and Shalev-Shwartz [DLSS13], where more data makes *improper* learning easier.

5 The treewidth transition

Exact inference in graphical models is the computation of learning closest to #SAT. It computes partition functions for maximum-likelihood estimation, marginals for prediction, and minimum-energy assignments for decoding. The textbook algorithms all take time exponential in the treewidth: variable elimination, junction trees, recursive conditioning, and compilation into decision diagrams or d-DNNF; for #SAT see [SS10].

Kwisthout, Bodlaender and van der Gaag [KB09, KBvdG10] derived from ETH, through the treewidth lower bounds of Marx [Mar10], that bounded treewidth is necessary for efficient inference. ETH is false, so their quantitative bound has lost its proof. The qualitative statement survives. Chandrasekaran, Srebro and Harsha [CSH08] show that no family of graphs of unbounded treewidth admits inference in time polynomial in the treewidth unless NP has polynomial-size circuits, and the grid-minor bounds that they assume are now theorems [CC16]. This section settles the quantitative question for Boolean models of bounded degree and arity. Treewidth-based algorithms take time $2^{\Theta(\mathrm{tw})}$. They are not optimal for models whose size is close to their treewidth, where the truth is $2^{\mathrm{tw}/\mathrm{polylogtw}}$ under $\mathrm{ETH}_C^{\mathrm{len}}$. They are optimal again for models that may be polynomially larger.

5.1 Subexponential inference for bounded-degree models

Theorem 5.1 (Inference)**.** *Let Φ be a factor graph on n Boolean variables with nonempty scopes, arity at most a, size $\|\Phi\|$, and $|\mathcal{F}|$ factors.*

(a) If all tables have nonnegative integer entries below 2^b, then $Z(\Phi)$ and, if $Z(\Phi) > 0$, the marginal $\Pr[x_i = 1]$ for every i can be computed exactly in time $\mathrm{poly}(|\Phi|) \cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\|\Phi\|,\ n + b|\mathcal{F}| + 1,\ a^2\}$.

(b) If the factors carry integer energies in $\{0, \dots, b'\}$, then the minimum total energy, a minimizer, and the number of minimizers can be computed in time $\mathrm{poly}(|\Phi|) \cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\|\Phi\|,\ n + 1,\ b'|\mathcal{F}| + 1,\ a^2\}$.

For constant degree, arity, b and b', all of these take time $2^{O(n/\sqrt{\log n})}$.

Proof sketch. For (a), use the counting ring with $P = n + b|\mathcal{F}| + 1$, which exceeds the bit length of $Z(\Phi)$; a marginal is a ratio of two partition functions. For (b), use the profile ring with table V^{E_f} for the factor f with energy E_f, and recover a minimizer by self-reduction as in Theorem 4.1 (Section E). □

5.2 Separation from elimination and compilation

Every bounded-degree expander has treewidth $\Theta(n)$ [GM09]. On such graphs the classical exact methods are exponentially slower than Theorem 5.1.

Corollary 5.2 (Separation)**.** *Let (G_n) be a family of n-vertex graphs of maximum degree $\Delta = O(1)$ with $\mathrm{tw}(G_n) = \Omega(n)$, for example bounded-degree expanders. Let Φ_n be the hardcore model on G_n: one variable per vertex and the factor $\mathbf{1}[\neg(x_u \wedge x_v)]$ on every edge, so that $Z(\Phi_n)$ counts independent sets.*

(a) The partition function $Z(\Phi_n)$ and all marginals can be computed in time $2^{O(n/\sqrt{\log n})}$.

(b) Variable elimination along any order, and junction-tree propagation over any tree decomposition, with explicitly stored tables, take time $2^{\Omega(n)}$.

(c) Every structured DNNF that represents the CNF $\bigwedge_{uv \in E(G_n)} (\neg x_u \vee \neg x_v)$ has size $2^{\Omega(n)}$ [ACMS20]. This includes every structured d-DNNF and every OBDD.

For suitable expanders of degree 3, Bova, Capelli, Mengel and Slivovsky [BCMS14] prove that *every* DNNF for this CNF has size $2^{\Omega(n)}$. This covers decision-DNNF, which captures the traces of DPLL-based exact model counters [BLRS13].

Contraction along one tree decomposition is the standard way to evaluate tensor networks and weighted model counts [MS08, KCMR19, DDOV19], and it costs $2^{\Theta(\mathrm{tw})}$. The algorithm of Theorem 5.1 is a signed sum of tensor networks contracted in different orders [Cla26], or a polynomial interpolation [Ope26a]. In either form it lies outside every model of computation that fixes one elimination order or one vtree. The preprint [Cla26] already identifies its sums with partition functions of pairwise Markov random fields with nonnegative weights. It also notes that expanders defeat every fixed tree of cuts, and it places its algorithm outside the contraction-tree model of Austrin, Kaski and Kubjas [AKK22]. The statements for marginals and minimum energy, the separation from structured DNNF, and the lower bounds below are new.

On models whose size is close to their treewidth, Theorem 5.1 is optimal up to polylogarithmic factors in the exponent. The reason is that OR-chains (Lemma 5.4 below) turn a CNF of length L into a model with at most $4L$ variables and treewidth at most $L + 2$.

Corollary 5.3 (Inference is SAT by length)**.** *Assume $\mathrm{ETH}^{\mathrm{len}}_C$, and let $\theta > 0$. No randomized algorithm with error probability at most $1/3$ decides whether the partition function of a $0/1$ factor*

graph of arity and degree at most 3 *is positive in time* $2^{\mathrm{tw}/(\log \mathrm{tw})^{C+\theta}}\mathrm{poly}(n)$*, or in time* $2^{n/(\log n)^{C+\theta}}$*. For* $C < 1$*, no algorithm runs in time* $2^{O(\mathrm{tw}/\log \mathrm{tw})}\mathrm{poly}(n)$*.*

So under $\mathrm{ETH}^{\mathrm{len}}_C$ the exponent of exact inference on such models lies between $\mathrm{tw}/(\log \mathrm{tw})^{C+o(1)}$ and $O(n/\sqrt{\log n})$. The second statement answers, for Boolean models and under this hypothesis, whether the $\mathrm{tw}/\log \mathrm{tw}$ of the ETH-based bound can be beaten: it cannot.

5.3 Treewidth is necessary again for large models

Treewidth algorithms are optimal again once the model may have many more variables than its treewidth. Lokshtanov, Marx and Saurabh [LMS18a] proved such bounds for graph problems from CNF-SAT without sparsification. For Maximum Independent Set, which is MAP inference in the hardcore model, their bound is $(2-\gamma)^{\mathrm{tw}}$. Theorem 5.5 adds the counting version under #CNF-SETH, for models of bounded arity and degree given with a path decomposition. The reduction copies each variable along a path. Every clause is then checked by a chain of local OR gates, while only the current copies stay in the bag.

Lemma 5.4 (OR-chains with copying)**.** *Given a CNF* F *on* N *variables with* M *clauses of total length* L*, one can compute in polynomial time a factor graph* Φ_F *with* 0/1 *tables, arity and maximum degree at most* 3*, and at most* $N + 3L$ *variables, together with a path decomposition of width at most* $N + 2$*, such that* $Z(\Phi_F) = \#F$*.*

Theorem 5.5 (Inference lower bound)**.** *Assume Hypothesis 2.10. For every* $\gamma > 0$ *there is* c *such that no algorithm that is given a* 0/1 *factor graph of arity at most* 3 *and degree at most* 4 *with* n *variables, together with a path decomposition of width* t*, computes its partition function in time* $O(2^{(1-\gamma)t})$*, even when* $n \le 5t^{c+1}$ *and the treewidth is at least* $\sqrt{t}/2$*. Under Hypothesis 2.9, the same holds for deciding whether the partition function is positive, and so for MAP inference.*

Proof sketch. Apply Lemma 5.4 to a CNF with N variables and at most N^c clauses, after deleting tautological clauses. Then add a disjoint $\lceil\sqrt{N}\rceil \times \lceil\sqrt{N}\rceil$ grid with the all-ones factor on each edge. The grid multiplies the partition function by a known power of 2 and makes the treewidth at least $\sqrt{N}$, while the width stays $t = N + 2$ and $n \le 5N^{c+1}$ (Section E). □

Corollary 5.6 (The treewidth transition)**.** *Consider Boolean models of bounded degree and arity, with small nonnegative integer weights.*

- *(a) If* $n \le \mathrm{tw} \cdot (\log \mathrm{tw})^{1/2-\theta}$ *for some* $\theta > 0$*, exact inference takes time* $2^{o(\mathrm{tw})}$*, and treewidth-based algorithms are not optimal.*
- *(b) Under Hypothesis 2.10, for every* $\gamma > 0$ *no algorithm runs in time* $2^{(1-\gamma)\mathrm{tw}}\mathrm{poly}(n)$ *on all such models. This already holds for models with* $n \le \mathrm{tw}^{O(1)}$ *variables. So treewidth-based algorithms are optimal up to* γ *in the exponent once polynomially many variables are allowed.*

Between the two regimes, when n ranges from $\mathrm{tw}\sqrt{\log \mathrm{tw}}$ to $\mathrm{tw}^{O(1)}$, the complexity is open. It is the complexity of #SAT on formulas whose length lies between $N\sqrt{\log N}$ and $N^{O(1)}$ in the number N of variables. The correspondence is tight in both directions. A 0/1 factor graph of arity a with $|\mathcal{F}|$ factors is a CNF with at most $2^a|\mathcal{F}|$ clauses of width at most a: it has one clause for each zero entry of each table. Conversely, Lemma 5.4 turns a CNF of length L, in which every variable occurs, into a 0/1 factor graph of arity and degree 3 with at most $4L$ variables. So exact inference in bounded-degree 0/1 models with n variables has the complexity of #SAT for formulas of length $\Theta(n)$. Any improvement of the rate $1/\sqrt{\log n}$ in Theorem 5.1 for such models is an improvement of the rate of #SAT as a function of formula length.

6 Learning theory after ETH: a ledger

Two kinds of statements need a new basis: the hypotheses, and the lower bounds proved from them. This section takes stock of both. Table 1 records the hypotheses, Section 6.2 gives the rule by which ETH-based bounds transfer, and Table 2 applies it to the conditional lower bounds of learning theory.

6.1 Hypotheses

Hypothesis	Status	Reason
ETH and SETH (k-SAT form) [IP01], randomized ETH, #ETH [DHM+14], #SETH, NSETH [CGI+16], Gap-ETH [Din16, MR17a]	false	each implies ETH [Ope26a, Cla26]
Sparse Max-3-SAT and Max-2-SAT [AV20]	false	Corollary 4.5
$\mathrm{ETH}_C^{\mathrm{len}}$ (Hypothesis 2.14)	false for $C < \frac{1}{2}$, open for $C \geq \frac{1}{2}$	SAT of length L takes time $2^{O(L/\sqrt{\log L})}$
Almost-ETH [Man26], Weak ETH [Aar26]	open	implied by $\mathrm{ETH}_C^{\mathrm{len}}$ for every C
CNF-SETH, #CNF-SETH, CNF-ETH	open	hard instances need superlinearly many clauses (Proposition 6.1); width $O(\log n)$ suffices (Lemma 3.5)
Max-3-SAT (Hypothesis 2.12)	open	superlinear size; Max-SAT does not sparsify
OV in dimension polylog$n$	open	implied by CNF-SETH [Wil05]
OV in dimension $c \log n$ [GIKW19]	open	no longer known to follow from a hypothesis about SAT (Proposition 6.1)
k-Clique, 3SUM, APSP, Set Cover, Hitting Set [AVW16], planted clique, $k \times k$ permutation clique [LMS18b]	open	not statements about bounded-width SAT with linearly many clauses

Table 1: Fine-grained hypotheses after the refutation of ETH.

The hard instances of CNF-SETH must have superlinearly many clauses. This follows from the duality between clause width and clause density of Calabro, Impagliazzo and Paturi [CIP06] together with the new algorithms.

Proposition 6.1 (Sparse CNF-SAT is subexponential)**.** *For all constants $c, \varepsilon' > 0$, the number of satisfying assignments of a CNF formula with n variables and at most cn clauses of arbitrary width can be computed in time $O(2^{\varepsilon' n})$.*

Proof sketch. Branch with the procedure of Lemma 3.5 for a large constant k. There are at most $(n+1)(e(ck+1))^{n/k} = 2^{O(n \log(ck)/k)}$ leaves, each a k-CNF, and Imported Theorem 2.4 counts each leaf in time $2^{o(n)}$ (Section F). □

Williams' reduction [Wil05] turns CNF-SAT with n variables and M clauses into Orthogonal Vectors with $2^{n/2}$ vectors in dimension M. Through sparsification, SETH thus implied that OV needs time $n^{2-o(1)}$ already in dimension $c \log n$. By Proposition 6.1, this route now starts from a false statement. With polynomially many clauses, the same reduction shows that CNF-SETH implies the hardness of OV in dimension polylog$n$.

Two weaker replacements of ETH appeared within a day of the refutation. Weak ETH asks for time $2^{n^{1-o(1)}}$ for 3-SAT [Aar26], and Almost-ETH excludes time $2^{n^{1-\gamma}}$ for every constant $\gamma > 0$ [Man26]. Both are implied by $\mathrm{ETH}_C^{\mathrm{len}}$, and both suffice for bounds with polynomial slack, such as the hardness of approximating Densest k-Subgraph in [Man26]. Composed with a quasi-linear PCP, they give only bounds of the form $2^{N^{1-\gamma}}$. The hypothesis $\mathrm{ETH}_C^{\mathrm{len}}$ is the fine-grained version. It measures the instance by its length for two reasons. Tseitin transformations and quasi-linear PCPs change the length only by polylogarithmic factors, and the new algorithms run in time $2^{O(L/\sqrt{\log L})}$ in the length.

6.2 How lower bounds transfer

Consider a lower bound proved from ETH by a reduction that maps a 3-CNF of length L, in time polynomial in its output, to an equivalent instance of size N with parameter k. A CNF of length L becomes a 3-CNF of length at most $3L$ by splitting clauses, so the reduction applies to every CNF. Assume that $k \log N \le L(\log L)^{\beta}$ and $L \le (k \log N)^{O(1)}$. An algorithm that solves the target in time $N^{k/g}$ then decides SAT of length L in time $2^{O(L(\log L)^{\beta}/g)} + 2^{O(L(\log L)^{\beta}/k)}$, where the second term pays for the reduction. Under $\mathrm{ETH}_C^{\mathrm{len}}$, for every $\theta > 0$, no algorithm therefore solves the target in time $N^{k/(\log(k \log N))^{C+\beta+\theta}}$. The bound is meaningful when $k \ge (\log(k \log N))^{C+\beta+\theta}$; for smaller k its exponent is below 1. In the same way, a bound $2^{\Omega(n)}$ proved by a reduction of size $n \le L(\log L)^{\beta}$ becomes $2^{n/(\log n)^{C+\beta+\theta}}$. The lower bounds of learning theory then fall into six groups. Table 2 lists examples of each.

(1) *Bounds with built-in slack survive under Almost-ETH.* Bounds of the form $n^{\log^{1-o(1)} n}$ or $2^{n^{1-o(1)}}$ need only Almost-ETH, and so they survive under $\mathrm{ETH}_C^{\mathrm{len}}$. Examples are the hardness of computing the VC and Littlestone dimensions [MR17b] and of robust proper learning of halfspaces with a margin [DKM19]. The hardness of proper learning of decision trees in [ABF+08] also has a hypothesis of this form.

(2) *Bounds from near-linear reductions lose a polylogarithmic factor in the exponent.* This covers the exact training of ReLU networks and the training of two or more ReLUs [GKMR21], proper learning of decision trees with queries [KST23a], whose $\tilde{O}(n^2)$ reduction gives $2^{\sqrt{N}/\mathrm{polylog}N}$, the computational–statistical tradeoffs of [BKST25], and the ETH-based inapproximability of the VC dimension [Man23]. Bounds proved from Gap-ETH through a near-linear PCP join this group through Corollary 4.11. The route from ETH through nearly-linear PCPs is the one used in [DKM19, GKMR21, Man23]; the new point is that its hypothesis survives.

(3) *Bounds whose parameter is $O(\log \log N)$ keep almost nothing.* Examples are the bounds $n^{\tilde{\Omega}(\log \log s)}$ for proper learning of decision trees of size s [KST23b] and $N^{\Omega(\log \log N)}$ for partial MCSP and MFSP [Ila20, Ila21]. Under $\mathrm{ETH}_C^{\mathrm{len}}$ they keep only a $(\log \log N)^{-C-O(1)}$ fraction of their exponent, so they remain superpolynomial only for small C.

(4) *Bounds from other hypotheses survive verbatim.* Consistency and proper learning of k-juntas need time $n^{k-o(1)}$ under CNF-SETH, because k-Dominating Set, which needs this time under CNF-SETH [PW10], is a special case of k-junta consistency. Label the indicator of each closed neighborhood 1 and the all-zero example 0. A junta on a set J separates each neighborhood from the zero example only if J meets the neighborhood, so J dominates the graph; conversely, the disjunction of a dominating set is consistent. ReLU training is W[1]-hard in the dimension d and needs time $n^{\Omega(d)}$ under the hypothesis that k-Clique has no $n^{o(k)}$ algorithm [FHN22]. The inapproximability of the VC dimension also follows from planted-clique hypotheses [Man23]. The bounds for partial MCSP survive if one takes the hardness of $k \times k$ Permutation Clique, $2^{\Omega(k \log k)}$, as a hypothesis of its own [LMS18b].

Lower bound	Original hypothesis	Group	Status and surviving hypothesis
VC and Littlestone dimension, $n^{\log^{1-o(1)} n}$ [MR17b]	randomized ETH	1	unchanged under Almost-ETH
Robust proper halfspaces with margin [DKM19]	randomized ETH	1	unchanged under Almost-ETH
Training $k \geq 2$ ReLUs, $2^{o(1/\varepsilon)}$ [GKMR21]	Gap-ETH	2	$2^{(1/\varepsilon)/\mathrm{polylog}(1/\varepsilon)}$ under $\mathrm{ETH}_C^{\mathrm{len}}$
Decision trees with queries [KST23a]	ETH	2	$2^{\sqrt{N}/\mathrm{polylog}N}$ under $\mathrm{ETH}_C^{\mathrm{len}}$
Sample–time tradeoffs [BKST25]	randomized ETH	2	$2^{n/\mathrm{polylog}n}$ under $\mathrm{ETH}_C^{\mathrm{len}}$
Proper decision trees, $n^{\tilde{\Omega}(\log\log s)}$ [KST23b]	randomized ETH	3	superpolynomial under $\mathrm{ETH}_C^{\mathrm{len}}$ only for small C
Partial MCSP and MFSP [Ila20, Ila21]	ETH	3, 4	as above; unchanged under permutation hardness [LMS18b]
Proper k-juntas, $n^{k-o(1)}$ [KST23b]	randomized SETH	4	unchanged under CNF-SETH [PW10]
ReLU training, $n^{\Omega(d)}$ [FHN22]	ETH	4	unchanged under the k-Clique hypothesis
Training one ReLU, $2^{o(1/\varepsilon^2)}$ [GKMR21]	bounded-degree Densest k-Subgraph	4	unaffected
Adversarially robust halfspaces [DKM20]	Gap-ETH	5	no replacement known
Kernel ERM, network gradients, $n^{2-o(1)}$ [BIS17]	SETH	6	unchanged under CNF-SETH for $d = \mathrm{polylog}n$
Attention with entries $\Theta(\sqrt{\log n})$ [AS23, AS24a]	SETH	6	unchanged under low-dimensional OV
VC dimension, $2^{\Omega(\lvert V\rvert)}$ [FGMT25]	ETH, via 3-coloring	–	basis lost; the source problem is subexponential
Inference needs bounded treewidth [KB09, KBvdG10]	ETH	–	$2^{\mathrm{tw}/\mathrm{polylogtw}}$ under $\mathrm{ETH}_C^{\mathrm{len}}$ (Corollary 5.3); $2^{(1-\gamma)\mathrm{tw}}$ under #CNF-SETH once $n = \mathrm{tw}^{O(1)}$ (Theorem 5.5)
Permutation problems, $2^{\Omega(k\log k)}$ [LMS18b, CFG+17]	ETH	–	basis lost; a hypothesis of its own (Section 8.1)
Proper k-term DNF	CNF-SETH	–	$2^{(1/2-\gamma)n}$ (Proposition F.1)

Table 2: Conditional lower bounds of learning theory and their status. The groups refer to Section 6.2.

(5) *Bounds from parameterized gap hypotheses have no replacement yet.* Gap-ETH gave $n^{o(k)}$ lower bounds for parameterized label cover, which underlie the bounds of [DKM20, KST24]. Corollary 4.11 gives gap hardness without a parameter, and re-deriving these bounds is open.

(6) *Polynomial-time bounds through Orthogonal Vectors keep their statement but change their basis.* These are $n^{2-o(1)}$ bounds stated under SETH, and the new algorithms give nothing subquadratic for OV. Bounds whose proofs need sparsification, typically through OV in dimension $\Theta(\log n)$, now rest on the low-dimensional OV hypothesis alone. They include the thresholds for fast attention and its gradients at entries of size $\sqrt{\log n}$ [AS23, AS24a] and its tensor version [AS24b], the transition for Gaussian kernel density estimation [AA22], the dichotomy for linear algebra on geometric graphs [ACSS20], and the hardness of approximate

nearest neighbors and maximum inner product [Rub18, ARW17]. Bounds proved for every $d = \omega(\log n)$ by reductions polynomial in d, such as those for kernel methods and neural-network ERM [BIS17], self-attention [KWH23] and hierarchical clustering [ACAH19], hold under CNF-SETH for $d = \text{polylog} n$, through the reduction of [Wil05] with polynomially many clauses. Bounds stated under #SETH, such as the tight treewidth bounds for counting list homomorphisms [FMR22], survive under #CNF-SETH exactly when their reductions start from CNF-SAT without sparsification. Bounds based on APSP or k-Clique, such as the one for the Viterbi algorithm [BT17], are unaffected.

The single-ReLU bound of [GKMR21] rests on a hypothesis about Densest k-Subgraph on bounded-degree graphs, a sparse problem. Our engine does not refute it. Tracking the size of a vertex set and the number of edges inside it needs a second formal variable. Its Kronecker substitution needs truncation order $\Theta(n^2)$, beyond the range of Imported Theorem 2.6.

6.3 The dense side: juntas and DNF

Juntas are the canonical non-local class: whether a description selects a variable affects the label of every example. The lower bound for juntas in group (4) therefore fits the locality principle. Proper DNF learning shows the same contrast inside one reduction. Pitt and Valiant [PV88] reduce graph k-coloring to k-term DNF consistency in dimension n, with one coordinate for each vertex. Graph k-coloring is now subexponential [Ope26a, Corollary 1.3], so the instances of this reduction are easy. The same reduction applied to *hypergraphs* with edges of unbounded size survives: under CNF-SETH, consistency for k-term DNF in dimension n needs time $2^{(1/2-\gamma)n}$ (Proposition F.1 in Section F).

7 Related work

The refutation. The two refutations [Ope26a, Cla26] appeared within days of each other. Within a day, Aaronson [Aar26] and Manurangsi [Man26] proposed the replacements Weak ETH and Almost-ETH, which have polynomial slack. Chen [Che26] applied the contraction theorem of [Ope26a] to the smallest grammar problem.

Fine-grained hypotheses. ETH and SETH are due to Impagliazzo and Paturi [IP01], and the sparsification lemma to Impagliazzo, Paturi and Zane [IPZ01]. The duality between clause width and clause density is due to Calabro, Impagliazzo and Paturi [CIP06]. The CNF form of SETH and the problems equivalent to it are studied in [CDL+16, PW10], the counting versions in [DHM+14], NSETH in [CGI+16], and Gap-ETH in [Din16, MR17a, CCK+20]. The Max-SAT hypotheses are from [LVW18, AV20], and the classification of [BCFK22] builds on them. Hypotheses with polylogarithmic slack in the input length appear in [CRTY23], and the measurement of SAT by instance length goes back to [SH90]. Before 2026, the best algorithms for SAT by length [Hir00, Wah05, CL09, PX23], by number of clauses [CXZ21], and for #SAT [IMP12, CW21] saved constant or $1/O(k)$ factors in the exponent. The Super-Strong ETH [VW21] conjectured that the saving $1/O(k)$ for k-SAT was optimal. Quasi-linear PCPs are due to Ben-Sasson and Sudan and to Dinur [BSS08, Din07]; Bafna, Minzer, Vyas and Yun obtain small soundness [BMVY25].

Hardness of learning. NP-hardness of proper learning goes back to Pitt and Valiant [PV88] and Blum and Rivest [BR92]; see also [ABF+08]. Recent conditional bounds concern decision

trees [KST23b, KST23a, KST24], MCSP and learning programs [Ila20, Ila21, Hir22], and neural networks [GKMR21, FHN22]. Others concern the VC dimension [MR17b, Man23, FGMT25], robust learning of halfspaces [DKM19, DKM20], parameterized learning [AKL07, BGS23], and computational–statistical tradeoffs [BKST25]. Decision trees can be learned properly with queries in time $n^{O(\log\log n)}$ [BLQT21], and training neural networks is $\exists\mathbb{R}$-complete [AKM21, BHJ+23]. ETH-based bounds also exist for unsupervised learning, for example for nonnegative matrix factorization [AGKM12] and for k-means and k-median clustering in low dimension [CAdMRR18, CACSS26]; the transfer rule of Section 6.2 applies to them. Daniely, Linial and Shalev-Shwartz [DLSS13] studied halfspaces on sparse data and showed that more data can speed up improper learning; we study the complementary exponential-time regime of proper learning. SETH-based quadratic bounds in machine learning include [BIS17, KWH23, AS23, AS24a, AS24b, AY25, ACSS20, AA22, Rub18, ARW17, ACAH19]. Section 6 classifies these results.

Counting, inference and permutations. The halving algorithm goes back to Barzdin and Freivald and to Littlestone [BF72, Lit88], and query-by-committee to Seung, Opper and Sompolinsky and to Freund et al. [SOS92, FSST97]. Sampling by self-reducibility is due to Jerrum, Valiant and Vazirani [JVV86]. Approximate counting gives efficient halving for total orders [GRS93] and efficient query-by-committee for convex version spaces [GBNT05]. Counting assignments by their weight goes back to Williams' algorithm for Max-2-CSP [Wil05]. The exponential mechanism is due to McSherry and Talwar [MT07], and its use as a generic private learner to Kasiviswanathan et al. [KLN+11]. Treewidth lower bounds for inference are in [CSH08, KB09, KBvdG10], building on Marx [Mar10]. CNF-SAT-based treewidth lower bounds for graph problems are in [LMS18a], and tight treewidth bounds for counting list homomorphisms are stated under #SETH in [FMR22]. Knowledge compilation needs size $2^{\Omega(\mathrm{tw})}$ for structured DNNF [ACMS20] and $2^{\Omega(n)}$ for every DNNF on suitable expanders [BCMS14, BCMS16]. Tree-decomposition contraction is the standard way to evaluate tensor networks [MS08, KCMR19, DDOV19]; the contraction-tree lower bounds of [AKK22] are discussed in [Cla26]. Learning to order is Feedback Arc Set [CSS99], which has a 2^n dynamic program [BFK+12], and permutation problems have $2^{\Omega(k\log k)}$ lower bounds from ETH [LMS18b, CFG+17].

8 The permutation obstacle and open problems

8.1 The permutation obstacle

Everything above concerns classes with discrete descriptions. With real-valued parameters, a local class on sparse data can encode a different kind of problem.

Proposition 8.1 (ERM for halfspaces on difference vectors)**.** *Let $G = ([n], A)$ be a digraph with nonnegative arc weights* wt*. Consider the sample $S_G = \{(e_i - e_j, 1) : (i,j) \in A\}$, in which the example of the arc (i,j) has weight* $\mathrm{wt}(i,j)$*, and homogeneous halfspaces $h_w(x) = \mathbf{1}[\langle w, x\rangle > 0]$ with $w \in \mathbb{R}^n$. Then the minimum over w of the weighted error $\sum_{(i,j)\in A} \mathrm{wt}(i,j)\,\mathbf{1}[h_w(e_i - e_j) = 0]$ is the minimum weight of a feedback arc set of G.*

This is the classical connection between learning to order and Feedback Arc Set [CSS99], in the language of halfspaces. The correctly classified arcs satisfy $w_i > w_j$, so they form an acyclic set. Conversely, a topological order of the arcs outside a feedback arc set gives a w that classifies all of them correctly (Section F). With negative examples as well, ERM becomes a weak-order variant of Feedback Arc Set, and the discussion below applies to it unchanged.

Feedback Arc Set has a $2^n\mathrm{poly}(n)$ dynamic program [BFK+12]. An algorithm in time $(2-\gamma)^n$ is a long-standing open problem, while $(1+\varepsilon)$-approximation is possible in time $(2-\gamma_\varepsilon)^n$ [BFIS25]. The classical ETH-based lower bound for Feedback Arc Set came from linear reductions from bounded-width problems, so it now has no basis. Yet no $2^{o(n)}$ algorithm follows from the new algorithms. The ETH-based $2^{\Omega(k \log k)}$ lower bounds for permutation problems [LMS18b, CFG+17] have lost their basis as well, and they now stand as a hypothesis of their own. Bounds for Hamiltonicity derived from CNF-SAT survive [CKN18], while sparse and undirected instances of Hamiltonicity and the Traveling Salesperson Problem beat 2^n [BHKK12, Bjö14]. For rank aggregation, Kemeny's rule and Feedback Arc Set on tournaments take time $2^{O(\sqrt{\mathrm{OPT}})}$ [KS10].

The natural Boolean encodings of Feedback Arc Set use either $\Theta(n \log n)$ bits or $\Theta(n^2)$ bits. The first records the positions in an order, which gives locality $\Theta(\log n)$. The second records the pairwise orders, with transitivity constraints. Hamiltonian Cycle and the Traveling Salesperson Problem behave in the same way. In the language of this paper, real-valued parameters turn ERM on sparse data into a *permutation problem*, whose natural encodings have logarithmic locality. That is the open window of the first transition, where neither the new algorithms nor CNF-SETH decide the complexity.

8.2 Open problems

Open Problem 1 (The locality window)**.** Consistency for κ-local classes is subexponential for $\kappa \ll (\log N)^{1/4}$ and CNF-SETH-hard for $\kappa \gg \log N$ (Theorems 3.2 and 3.8). By Theorem 3.4, closing this window is the same as determining the complexity of k-SAT for $k = (\log n)^{\Theta(1)}$. Can $\#k$-SAT with $k = \sqrt{\log n}$ be solved in time $2^{o(n)}$?

Open Problem 2 (Constant accuracy beyond $\sqrt{\log N}$)**.** At a constant accuracy $\varepsilon \le \varepsilon_0$ and locality $\kappa \le (\log N)^{1/2-\theta}$, the exponent of proper learning is $O(N/(\log N)^\theta)$ (Theorem 3.10), and under $\mathrm{ETH}_C^{\mathrm{len}}$ it is at least $N/(\log N)^{C+\beta+1}$ for LD_3 (Theorem 4.9). For $\kappa \ge \sqrt{\log N}$, is it still $N/\mathrm{polylog}N$, or does it reach $(1-o(1))N$ at $\kappa = K \log N$? A $(1-o(1))N$ lower bound at constant accuracy needs gap hardness on instances with $(1+o(1))N$ variables.

Open Problem 3 (The agnostic window)**.** For constant κ, the exponent of proper agnostic learning is $O(N/\sqrt{\log N})$ at constant accuracy (Theorem 4.3), at least $N/\mathrm{polylog}N$ there under $\mathrm{ETH}_C^{\mathrm{len}}$ (Theorem 4.9), and $(1-o(1))N$ at accuracy $N^{-3}/8$ under the Max-3-SAT Hypothesis (Theorem 4.12). How does it grow for $\varepsilon = N^{-c}$ with $0 < c < 3$?

Open Problem 4 (The treewidth window)**.** Determine the complexity of exact inference for bounded-degree models with $\mathrm{tw}\sqrt{\log \mathrm{tw}} \le n \le \mathrm{tw}^{O(1)}$ (Corollary 5.6). By Section 5.3, this is the complexity of #SAT on formulas with N variables and length between $N\sqrt{\log N}$ and $N^{O(1)}$.

Open Problem 5 (Permutations)**.** Can Feedback Arc Set, that is, ERM for real-valued homogeneous halfspaces on positive examples of the form $e_i - e_j$, be solved in time $2^{o(n)}$? Even time $(2-\gamma)^n$ is open [BFIS25]. Does a hypothesis about permutation problems, such as the one of [LMS18b], follow from a hypothesis about SAT?

Open Problem 6 (Improper learning)**.** The new algorithms evaluate signed sums of tensor networks and low-degree interpolants. Do such representations, used as *hypothesis classes*, give new improper learners for dense classes, as polynomial threshold functions did for DNF [KS04]?

8.3 Conclusion

The refutation of ETH is a statement about sparse, bounded-width structure, and learning inherits it through locality. Accuracy decides which form of SAT governs learning: consistency is SAT by the number of variables, and constant-accuracy learning is SAT by the length of the formula. Where examples are dense, where agnostic learning must reach inverse-polynomial accuracy, and where a model may be polynomially larger than its treewidth, the surviving hypotheses keep the old exponents. The boundary lies at logarithmic locality, between fixed-width and unbounded-width SAT, and the permutation problems of learning to order lie inside it.

References

[AA22] Amol Aggarwal and Josh Alman. Optimal-degree polynomial approximations for exponentials and Gaussian kernel density estimation. In *37th Computational Complexity Conference (CCC)*, volume 234 of *LIPIcs*, pages 22:1–22:23, 2022.

[Aar26] Scott Aaronson. Podcasts about the topic: Part II. Shtetl-Optimized, October 26, 2026, comment #31 of October 27, 2026. https://scottaaronson.blog/?p=10239, 2026.

[ABF+08] Misha Alekhnovich, Mark Braverman, Vitaly Feldman, Adam R. Klivans, and Toniann Pitassi. The complexity of properly learning simple concept classes. *Journal of Computer and System Sciences*, 74(1):16–34, 2008.

[ACAH19] Amir Abboud, Vincent Cohen-Addad, and Hussein Houdrouge. Subquadratic high-dimensional hierarchical clustering. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, 2019.

[ACMS20] Antoine Amarilli, Florent Capelli, Mikaël Monet, and Pierre Senellart. Connecting knowledge compilation classes and width parameters. *Theory of Computing Systems*, 64(5):861–914, 2020.

[ACSS20] Josh Alman, Timothy Chu, Aaron Schild, and Zhao Song. Algorithms and hardness for linear algebra on geometric graphs. In *61st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 541–552, 2020.

[ACW16] Josh Alman, Timothy M. Chan, and Ryan Williams. Polynomial representations of threshold functions and algorithmic applications. In *57th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 467–476, 2016.

[AGKM12] Sanjeev Arora, Rong Ge, Ravi Kannan, and Ankur Moitra. Computing a nonnegative matrix factorization – provably. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, 2012.

[AKK22] Per Austrin, Petteri Kaski, and Kaie Kubjas. Tensor network complexity of multilinear maps. *Theory of Computing*, 18(16):1–54, 2022.

[AKL07] Vikraman Arvind, Johannes Köbler, and Wolfgang Lindner. Parameterized learnability of k-juntas and related problems. In *Algorithmic Learning Theory (ALT)*, Lecture Notes in Computer Science, pages 120–134, 2007.

[AKM21] Mikkel Abrahamsen, Linda Kleist, and Tillmann Miltzow. Training neural networks is ER-complete. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 18293–18306, 2021.

[ALS22] Emmanuel Abbe, Shuangping Li, and Allan Sly. Binary perceptron: efficient algorithms can find solutions in a rare well-connected cluster. In *Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC)*, pages 860–873, 2022.

[ARW17] Amir Abboud, Aviad Rubinstein, and Ryan Williams. Distributed PCP theorems for hardness of approximation in P. In *58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 25–36, 2017.

[AS23] Josh Alman and Zhao Song. Fast attention requires bounded entries. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, pages 63117–63135, 2023.

[AS24a] Josh Alman and Zhao Song. The fine-grained complexity of gradient computation for training large language models. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, pages 61419–61454, 2024.

[AS24b] Josh Alman and Zhao Song. How to capture higher-order correlations? Generalizing matrix softmax attention to Kronecker computation. In *International Conference on Learning Representations (ICLR)*, pages 4306–4333, 2024.

[AV20] Josh Alman and Virginia Vassilevska Williams. OV graphs are (probably) hard instances. In *11th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 151 of *LIPIcs*, pages 83:1–83:18, 2020.

[AVW16] Amir Abboud, Virginia Vassilevska Williams, and Joshua Wang. Approximation and fixed parameter subquadratic algorithms for radius and diameter in sparse graphs. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 377–391, 2016.

[AY25] Josh Alman and Hantao Yu. Fundamental limitations on subquadratic alternatives to transformers. In *International Conference on Learning Representations (ICLR)*, pages 47475–47498, 2025.

[BCFK22] Karl Bringmann, Alejandro Cassis, Nick Fischer, and Marvin Künnemann. A structural investigation of the approximability of polynomial-time problems. In *49th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 229 of *LIPIcs*, pages 30:1–30:20, 2022.

[BCMS14] Simone Bova, Florent Capelli, Stefan Mengel, and Friedrich Slivovsky. A strongly exponential separation of DNNFs from CNF formulas. arXiv:1411.1995, 2014.

[BCMS16] Simone Bova, Florent Capelli, Stefan Mengel, and Friedrich Slivovsky. Knowledge compilation meets communication complexity. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1008–1013, 2016.

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

[BEHW89] 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.

[BF72] J. M. Barzdin and R. V. Freivald. On the prediction of general recursive functions. *Doklady Akademii Nauk SSSR*, 206(3):521–524, 1972. English translation in Soviet Mathematics Doklady 13:1224–1228.

[BFIS25] Matthias Bentert, Fedor V. Fomin, Tanmay Inamdar, and Saket Saurabh. Exponential-time approximation (schemes) for vertex-ordering problems. In *16th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 325 of *LIPIcs*, pages 15:1–15:18, 2025.

[BFK+12] Hans L. Bodlaender, Fedor V. Fomin, Arie M. C. A. Koster, Dieter Kratsch, and Dimitrios M. Thilikos. A note on exact algorithms for vertex ordering problems on graphs. *Theory of Computing Systems*, 50(3):420–432, 2012.

[BGS23] Cornelius Brand, Robert Ganian, and Kirill Simonov. A parameterized theory of PAC learning. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, volume 37, pages 6834–6841, 2023.

[BHJ+23] Daniel Bertschinger, Christoph Hertrich, Paul Jungeblut, Tillmann Miltzow, and Simon Weber. Training fully connected neural networks is $\exists\mathbb{R}$-complete. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, pages 36222–36237, 2023.

[BHKK12] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. The traveling salesman problem in bounded degree graphs. *ACM Transactions on Algorithms*, 8(2):18:1–18:13, 2012.

[BIS17] Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. On the fine-grained complexity of empirical risk minimization: Kernel methods and neural networks. In *Advances in Neural Information Processing Systems 30 (NIPS)*, pages 4308–4318, 2017.

[Bjö14] Andreas Björklund. Determinant sums for undirected Hamiltonicity. *SIAM Journal on Computing*, 43(1):280–299, 2014.

[BKST25] Guy Blanc, Caleb Koch, Carmen Strassle, and Li-Yang Tan. Computational-statistical tradeoffs from NP-hardness. In *66th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 512–530, 2025.

[BLQT21] Guy Blanc, Jane Lange, Mingda Qiao, and Li-Yang Tan. Properly learning decision trees in almost polynomial time. In *62nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 920–929, 2021.

[BLRS13] Paul Beame, Jerry Li, Sudeepa Roy, and Dan Suciu. Lower bounds for exact model counting and applications in probabilistic databases. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 52–61, 2013.

[BMVY25] Mitali Bafna, Dor Minzer, Nikhil Vyas, and Zhiwei Yun. Quasi-linear size PCPs with small soundness from HDX. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing (STOC)*, pages 45–53, 2025.

[BNS13] Amos Beimel, Kobbi Nissim, and Uri Stemmer. Characterizing the sample complexity of private learners. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 97–110, 2013.

[BR92] Avrim Blum and Ronald L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 5(1):117–127, 1992.

[BSS08] Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM Journal on Computing*, 38(2):551–607, 2008.

[BSS+19] Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S. Meel, and Prateek Saxena. Quantitative verification of neural networks and its security applications. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1249–1264, 2019.

[BT17] Arturs Backurs and Christos Tzamos. Improving Viterbi is hard: better runtimes imply faster clique algorithms. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *PMLR*, pages 311–321, 2017.

[BZ16] Mark Bun and Mark Zhandry. Order-revealing encryption and the hardness of private learning. In *Theory of Cryptography (TCC)*, Lecture Notes in Computer Science, pages 176–206, 2016.

[CACSS26] Vincent Cohen-Addad, Karthik C. S., David Saulpic, and Chris Schwiegelshohn. Almost-optimal upper and lower bounds for clustering in low dimensional Euclidean spaces. In *42nd International Symposium on Computational Geometry (SoCG)*, volume 367 of *LIPIcs*, pages 34:1–34:16, 2026.

[CAdMRR18] Vincent Cohen-Addad, Arnaud de Mesmay, Eva Rotenberg, and Alan Roytman. The bane of low-dimensionality clustering. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2018.

[CC16] Chandra Chekuri and Julia Chuzhoy. Polynomial bounds for the grid-minor theorem. *Journal of the ACM*, 63(5):40:1–40:65, 2016.

[CCK+20] Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-Exponential Time Hypothesis to fixed parameter tractable inapproximability: Clique, Dominating Set, and more. *SIAM Journal on Computing*, 49(4):772–810, 2020.

[CDL+16] Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On problems as hard as CNF-SAT. *ACM Transactions on Algorithms*, 12(3):41:1–41:24, 2016.

[CFG+17] Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socała. Tight lower bounds on graph embedding problems. *Journal of the ACM*, 64(3):18:1–18:22, 2017.

[CGI+16] Marco L. Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mihajlin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic extensions of the strong exponential time hypothesis and consequences for non-reducibility. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 261–270, 2016.

[Che26] Xiangle Chen. Exact algorithms and complexity of the smallest grammar problem. arXiv:2610.32298v2, October 27, 2026, 2026.

[CIP06] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. A duality between clause width and clause density for SAT. In *21st Annual IEEE Conference on Computational Complexity (CCC)*, pages 252–260, 2006.

[CKN18] Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Fast Hamiltonicity checking via bases of perfect matchings. *Journal of the ACM*, 65(3):12:1–12:46, 2018.

[CKS20] Clément L. Canonne, Gautam Kamath, and Thomas Steinke. The discrete Gaussian for differential privacy. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, pages 15676–15688, 2020.

[CL09] Jianer Chen and Yang Liu. An improved SAT algorithm in terms of formula length. In *Algorithms and Data Structures (WADS)*, volume 5664 of *Lecture Notes in Computer Science*, pages 144–155, 2009.

[Cla26] Claude (Anthropic). A $2^{o(n)}$ time algorithm for 3-SAT. Hexagon preprint hexagon:2610.00577v1, dated October 26, 2026. https://hexagonmath.org/2610.00577v1. Lean development at https://github.com/anthropics/formal-math/tree/main/eth, 2026.

[CRTY23] Lijie Chen, Ron D. Rothblum, Roei Tell, and Eylon Yogev. On exponential-time hypotheses, derandomization, and circuit lower bounds. *Journal of the ACM*, 70(4):1–62, 2023.

[CSH08] Venkat Chandrasekaran, Nathan Srebro, and Prahladh Harsha. Complexity of inference in graphical models. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 70–78, 2008.

[CSS99] William W. Cohen, Robert E. Schapire, and Yoram Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.

[CW21] Timothy M. Chan and R. Ryan Williams. Deterministic APSP, orthogonal vectors, and more: quickly derandomizing Razborov–Smolensky. *ACM Transactions on Algorithms*, 17(1):2:1–2:14, 2021.

[CXZ21] Huairui Chu, Mingyu Xiao, and Zhe Zhang. An improved upper bound for SAT. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, volume 35, pages 3707–3714, 2021.

[DDOV19] Jeffrey M. Dudek, Leonardo Dueñas-Osorio, and Moshe Y. Vardi. Efficient contraction of large tensor networks for weighted model counting through graph decompositions. arXiv:1908.04381, 2019.

[DHM+14] Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlén. Exponential time complexity of the permanent and the Tutte polynomial. *ACM Transactions on Algorithms*, 10(4):21:1–21:32, 2014.

[Din07] Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM*, 54(3):12:1–12:44, 2007.

[Din16] Irit Dinur. Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover. Electronic Colloquium on Computational Complexity (ECCC), TR16-128, 2016.

[DKM19] Ilias Diakonikolas, Daniel M. Kane, and Pasin Manurangsi. Nearly tight bounds for robust proper learning of halfspaces with a margin. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, 2019.

[DKM20] Ilias Diakonikolas, Daniel M. Kane, and Pasin Manurangsi. The complexity of adversarially robust proper learning of halfspaces with agnostic noise. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, pages 20449–20461, 2020.

[DLSS13] Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. More data speeds up training time in learning halfspaces over sparse vectors. In *Advances in Neural Information Processing Systems 26 (NIPS)*, 2013.

[DNR+09] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 381–390, 2009.

[FGMT25] Florent Foucaud, Harmender Gahlawat, Fionn Mc Inerney, and Prafullkumar Tale. The parameterized complexity of computing the VC-dimension. Advances in Neural Information Processing Systems 38 (NeurIPS); arXiv:2510.17451, 2025.

[FHN22] Vincent Froese, Christoph Hertrich, and Rolf Niedermeier. The computational complexity of ReLU network training parameterized by data dimensionality. *Journal of Artificial Intelligence Research*, 74:1775–1790, 2022.

[FMR22] Jacob Focke, Dániel Marx, and Paweł Rzążewski. Counting list homomorphisms from graphs of bounded treewidth: tight complexity bounds. In *Proceedings of the 33rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2022.

[FSST97] Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2–3):133–168, 1997.

[GBNT05] Ran Gilad-Bachrach, Amir Navot, and Naftali Tishby. Query by committee made real. In *Advances in Neural Information Processing Systems 18 (NIPS)*, 2005.

[GIKW19] Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. *ACM Transactions on Algorithms*, 15(2), 2019.

[GKMR21] Surbhi Goel, Adam Klivans, Pasin Manurangsi, and Daniel Reichman. Tight hardness results for training depth-2 ReLU networks. In *12th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 185 of *LIPIcs*, pages 22:1–22:14, 2021.

[GM09] Martin Grohe and Dániel Marx. On tree width, bramble size, and expansion. *Journal of Combinatorial Theory, Series B*, 99(1):218–228, 2009.

[GRS93] Sally A. Goldman, Ronald L. Rivest, and Robert E. Schapire. Learning binary relations and total orders. *SIAM Journal on Computing*, 22(5):1006–1034, 1993.

[Gyö90] Géza Györgyi. First-order transition to perfect generalization in a neural network with binary synapses. *Physical Review A*, 41(12):7097–7100, 1990.

[Hir00] Edward A. Hirsch. New worst-case upper bounds for SAT. *Journal of Automated Reasoning*, 24(4):397–420, 2000.

[Hir22] Shuichi Hirahara. NP-hardness of learning programs and partial MCSP. In *63rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 968–979, 2022.

[HMP04] Haym Hirsh, Nina Mishra, and Leonard Pitt. Version spaces and the consistency problem. *Artificial Intelligence*, 156(2):115–138, 2004.

[Ila20] Rahul Ilango. Constant depth formula and partial function versions of MCSP are hard. In *61st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 424–433, 2020.

[Ila21] Rahul Ilango. The minimum formula size problem is (ETH) hard. In *62nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 427–432, 2021.

[Ilv20] Christina Ilvento. Implementing the exponential mechanism with base-2 differential privacy. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 717–742, 2020.

[IMP12] Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 961–972, 2012.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[JVV86] Mark R. Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.

[KB09] Johan Kwisthout and Hans L. Bodlaender. Conditional lower bounds on the complexity of probabilistic inference. Technical Report UU-CS-2009-018, Department of Information and Computing Sciences, Utrecht University, 2009.

[KBvdG10] Johan Kwisthout, Hans L. Bodlaender, and Linda C. van der Gaag. The necessity of bounded treewidth for efficient inference in Bayesian networks. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI)*, pages 237–242, 2010.

[KCMR19] Stefanos Kourtis, Claudio Chamon, Eduardo Mucciolo, and Andrei E. Ruckenstein. Fast counting with tensor networks. *SciPost Physics*, 7(5):060, 2019.

[KLN+11] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[KS04] Adam R. Klivans and Rocco A. Servedio. Learning DNF in time $2^{\tilde{O}(n^{1/3})}$. *Journal of Computer and System Sciences*, 68(2):303–318, 2004.

[KS10] Marek Karpinski and Warren Schudy. Faster algorithms for feedback arc set tournament, Kemeny rank aggregation and betweenness tournament. In *Algorithms and Computation (ISAAC)*, Lecture Notes in Computer Science, pages 3–14, 2010.

[KST23a] Caleb Koch, Carmen Strassle, and Li-Yang Tan. Properly learning decision trees with queries is NP-hard. In *64th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 2383–2407, 2023.

[KST23b] Caleb Koch, Carmen Strassle, and Li-Yang Tan. Superpolynomial lower bounds for decision tree learning and testing. In *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1962–1994, 2023.

[KST24] Caleb Koch, Carmen Strassle, and Li-Yang Tan. Fast decision tree learning solves hard coding-theoretic problems. In *65th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1893–1910, 2024.

[KWH23] Feyza Duman Keles, Pruthuvi Mahesakya Wijewardena, and Chinmay Hegde. On the computational complexity of self-attention. In *International Conference on Algorithmic Learning Theory (ALT)*, volume 201 of *PMLR*, pages 597–619, 2023.

[Lam95] Leslie Lamport. How to write a proof. *The American Mathematical Monthly*, 102(7):600–608, 1995.

[Lam12] Leslie Lamport. How to write a 21st century proof. *Journal of Fixed Point Theory and Applications*, 11(1):43–63, 2012.

[Lit88] Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

[LMS18a] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Known algorithms on graphs of bounded treewidth are probably optimal. *ACM Transactions on Algorithms*, 14(2):13:1–13:30, 2018.

[LMS18b] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Slightly superexponential parameterized problems. *SIAM Journal on Computing*, 47(3):675–702, 2018.

[LVW18] Andrea Lincoln, Virginia Vassilevska Williams, and Ryan Williams. Tight hardness for shortest cycles and paths in sparse graphs. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1236–1252, 2018.

[Man23] Pasin Manurangsi. Improved inapproximability of VC dimension and Littlestone's dimension via (unbalanced) biclique. In *14th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 251 of *LIPIcs*, pages 85:1–85:18, 2023.

[Man26] Pasin Manurangsi. Almost-polynomial ratio Almost-ETH-hardness of approximating densest k-subgraph. arXiv:1611.05991v3, version of October 27, 2026, 2026.

[Mar10] Dániel Marx. Can you beat treewidth? *Theory of Computing*, 6:85–112, 2010.

[MR17a] Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense CSPs. In *44th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 80 of *LIPIcs*, pages 78:1–78:15, 2017.

[MR17b] Pasin Manurangsi and Aviad Rubinstein. Inapproximability of VC dimension and Littlestone's dimension. In *Proceedings of the 30th Conference on Learning Theory (COLT)*, volume 65 of *PMLR*, pages 1432–1460, 2017.

[MS08] Igor L. Markov and Yaoyun Shi. Simulating quantum computation by contracting tensor networks. *SIAM Journal on Computing*, 38(3):963–981, 2008.

[MT07] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 94–103, 2007.

[NKR+18] Nina Narodytska, Shiva Kasiviswanathan, Leonid Ryzhyk, Mooly Sagiv, and Toby Walsh. Verifying properties of binarized deep neural networks. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

[Ope26a] OpenAI. The Exponential Time Hypothesis is false. OpenAI Math Release preprint, dated October 17, 2026, revised October 25, 2026, released October 26, 2026. https://github.com/openai/math/blob/main/preprints/The-Exponential-Time-Hypothesis-is-False-October-25-2026/paper.pdf, 2026.

[Ope26b] OpenAI. Scope of the Lean formalization for "The Exponential Time Hypothesis is false". https://github.com/openai/math/blob/main/lean/docs/378.md, 2026.

[PV88] Leonard Pitt and Leslie G. Valiant. Computational limitations on learning from examples. *Journal of the ACM*, 35(4):965–984, 1988.

[PW10] Mihai Pătraşcu and Ryan Williams. On the possibility of faster SAT algorithms. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1065–1075, 2010.

[PX23] Junqiang Peng and Mingyu Xiao. Further improvements for SAT in terms of formula length. *Information and Computation*, 294:105085, 2023.

[Rub18] Aviad Rubinstein. Hardness of approximate nearest neighbor search. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1260–1268, 2018.

[Sch05] Rainer Schuler. An algorithm for the satisfiability problem of formulas in conjunctive normal form. *Journal of Algorithms*, 54(1):40–44, 2005.

[SH90] Richard E. Stearns and Harry B. Hunt III. Power indices and easier hard problems. *Mathematical Systems Theory*, 23:209–225, 1990.

[SOS92] H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory (COLT)*, pages 287–294, 1992.

[SS10] Marko Samer and Stefan Szeider. Algorithms for propositional model counting. *Journal of Discrete Algorithms*, 8(1):50–64, 2010.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[SST92] H. Sebastian Seung, Haim Sompolinsky, and Naftali Tishby. Statistical mechanics of learning from examples. *Physical Review A*, 45(8):6056–6091, 1992.

[TIC+19] Rodrigo Toro Icarte, León Illanes, Margarita P. Castro, Andre A. Cire, Sheila A. McIlraith, and J. Christopher Beck. Training binarized neural networks using MIP and CP. In *Principles and Practice of Constraint Programming (CP)*, Lecture Notes in Computer Science, pages 401–417, 2019.

[Ull16] Jonathan Ullman. Answering $n^{2+o(1)}$ counting queries with differential privacy is hard. *SIAM Journal on Computing*, 45(2):473–496, 2016.

[VW21] Nikhil Vyas and Ryan Williams. On super strong ETH. *Journal of Artificial Intelligence Research*, 70:473–495, 2021.

[Wah05] Magnus Wahlström. An algorithm for the SAT problem for formulae of linear length. In *Algorithms – ESA 2005*, volume 3669 of *Lecture Notes in Computer Science*, pages 107–118, 2005.

[WFS15] Yu-Xiang Wang, Stephen Fienberg, and Alex Smola. Privacy for free: posterior sampling and stochastic gradient Monte Carlo. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37 of *PMLR*, pages 2493–2502, 2015.

[Wil05] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2–3):357–365, 2005.

A Roadmap of the appendix

The appendix contains the full proofs of all results of the main text, in the order of the sections, and ends with a table of notation. Figure 2 shows how the results depend on each other.

Figure 2: Dependencies between the results. An arrow from X to Y means that the proof of Y cites X, after transitive reduction. Dashed frames mark imported theorems, thick frames mark the main theorems, and hypotheses are not drawn. Theorem 4.12 and Proposition 8.1 use no other result.

How to read the proofs. We write the proofs in the structured style of Lamport [Lam95, Lam12]. A proof is a list of numbered steps, and $\langle k \rangle j$ is the j-th step at level k. Each step states a claim, and a proof of the claim follows it. This proof is a short paragraph or a list of steps at the next level. The last step of each list, Q.E.D., proves the goal of the list from the earlier steps. ASSUME and PROVE state the hypothesis and the goal of a step, LET introduces a symbol, and CASE marks one case of a case split. A reader who wants only the main idea can read the level-1 steps and skip their proofs. Each restated result keeps its number from the main text.

B Proofs for Section 2: the engine

Lemma B.1 (Factor graphs as tensor networks)**.** *Let Φ be a factor graph of arity at most a whose scopes are all nonempty, and let u be the number of variables in no scope. There is a bipartite Boolean tensor network G_Φ with at most $4\|\Phi\|$ edges, at most $4\|\Phi\|$ vertices and maximum degree at*

most $\max\{a, 3\}$*, such that* $Z(\Phi) = 2^u \cdot \text{contraction}(G_\Phi)$*. Every entry of* G_Φ *is either an entry of some table* ψ_f *or an element of* $\{0, 1\}$*. The network can be built in time polynomial in the size of* Φ*.*

Proof of Lemma B.1. ⟨1⟩1. LET: G' be the following tensor network. Each factor f is a vertex with one edge (a *port*) for each variable of I_f and with tensor ψ_f. Each variable v of degree $\deg(v) \geq 1$ gets a *gadget* attached to its $\deg(v)$ ports:

- if $\deg(v) = 1$, one vertex with the unary tensor $(1, 1)$;
- if $\deg(v) = 2$, one vertex with the binary equality tensor;
- if $\deg(v) \geq 3$, a tree with $\deg(v) - 2$ vertices of degree 3, each with the ternary equality tensor, whose $\deg(v)$ leaf edges are the ports of v and which has $\deg(v) - 3$ internal edges.

Then, for every variable v and every assignment of its ports, the sum over the internal edges of its gadget of the product of the gadget tensors is 1 if all ports of v carry the same bit, and 0 otherwise.

PROOF: For $\deg(v) \leq 2$ this is the definition of the tensor. For $\deg(v) \geq 3$, a product of ternary equality tensors is nonzero only if every gadget vertex sees one bit on all three of its edges. The tree is connected, so then all its edges carry one bit. Conversely, the constant assignment gives the product 1. So the sum is 1 if the ports agree and 0 otherwise.

⟨1⟩2. $Z(\Phi) = 2^u \cdot \text{contraction}(G')$.

PROOF: By ⟨1⟩1, an assignment of the ports contributes to the contraction only if, for every variable, all ports of the variable carry the same bit, and then it contributes exactly once. These assignments correspond bijectively to the assignments x of the $n - u$ variables that lie in some scope, and their product is $\prod_f \psi_f(x|_{I_f})$. Each variable in no scope contributes the factor 2 to $Z(\Phi)$.

⟨1⟩3. Subdividing every edge of G' by a new vertex with the binary equality tensor gives a bipartite network G_Φ with the same contraction.

PROOF: A new vertex forces its two edges to carry the same bit, so the assignments with nonzero product correspond bijectively to those of G', with the same product. Old vertices form one side and new vertices the other.

⟨1⟩4. The network G_Φ has at most $4\|\Phi\|$ edges, at most $4\|\Phi\|$ vertices, and maximum degree at most $\max\{a, 3\}$.

PROOF: A variable v has $\deg(v)$ ports and at most $\deg(v)$ internal edges, so G' has at most $2\|\Phi\|$ edges, and G_Φ has at most $4\|\Phi\|$. The network G' has $|\mathcal{F}| \leq \|\Phi\|$ factor vertices, because every scope is nonempty, and at most $\sum_v \deg(v) = \|\Phi\|$ gadget vertices. The subdivision adds at most $2\|\Phi\|$ vertices. Factor vertices have degree $|I_f| \leq a$, gadget vertices have degree at most 3, and subdivision vertices have degree 2.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩2, ⟨1⟩3 and ⟨1⟩4. All entries are table entries or bits, and the construction takes polynomial time.

□

Theorem 2.7 (Sparse sum-product)**.** *Let* Φ *be a factor graph over* $\mathcal{B}_{P,\ell}$ *of arity at most* a*, whose table entries can be computed in polynomial time. Let* s *be an integer with* $s \geq \max\{4\|\Phi\|, P, \ell, 2\}$ *and* $a \leq s^{9/10}$*. Then* $Z(\Phi)$ *can be computed deterministically in time* $\text{poly}(s + |\Phi|) \cdot 2^{O(s/\sqrt{\log s})}$ *and space* $\text{poly}(s + |\Phi|)$*. The constants depend only on the polynomial that bounds the time to evaluate one table entry.*

Proof of Theorem 2.7. ⟨1⟩1. LET: $\mathcal{F}_0$ be the factors with empty scope, and let Φ' be Φ without

them. Then $Z(\Phi) = \left(\prod_{f \in \mathcal{F}_0} \psi_f\right) \cdot Z(\Phi')$.

PROOF: A factor with empty scope contributes the same constant to every term.

⟨1⟩2. CASE: $\|\Phi'\| = 0$. Then $Z(\Phi') = 2^n$, and the theorem holds.

PROOF: Without factors every assignment contributes 1.

⟨1⟩3. CASE: $\|\Phi'\| \geq 1$. Then the network $G_{\Phi'}$ of Lemma B.1 satisfies the hypotheses of Imported Theorem 2.6 with the given s.

PROOF: It has at most $4\|\Phi'\| \leq s$ edges and vertices. Here $s \geq 4$, so $\lceil s^{9/10} \rceil \geq 4$, and the degree bound $\max\{a, 3\} \leq \lceil s^{9/10} \rceil$ follows from $a \leq s^{9/10}$. The ring is $\mathcal{B}_{P,\ell}$ with $P, \ell \leq s$. Each entry is a table entry, computable in polynomial time, or a bit. The description of $G_{\Phi'}$ has polynomial size.

⟨1⟩4. Q.E.D.

PROOF: In the first case use ⟨1⟩2. In the second, Imported Theorem 2.6 computes the contraction of $G_{\Phi'}$ in time $\text{poly}(s + |\Phi|) \cdot 2^{O(s/\sqrt{\log s})}$ and polynomial space, by ⟨1⟩3. Multiply by 2^u (Lemma B.1) and by the constants of ⟨1⟩1 in $\mathcal{B}_{P,\ell}$.

□

Lemma 2.8 (Exact error profiles)**.** *Let $\mathcal{H}$ be κ-local with description length N, let S be a sample of size m, and let $s = \max\{4\kappa m,\, m+1,\, \kappa^2\}$. The error profile E_S can be computed exactly and deterministically in time $\text{poly}(N, m) \cdot 2^{O(s/\sqrt{\log s})}$. The same holds for the error profile of the descriptions that extend any given partial assignment.*

Proof of Lemma 2.8. ⟨1⟩1. LET: $R(S) = \bigcup_i R(x_i)$ and $n_S = |R(S)| \leq \kappa m$. For $u \in \{0,1\}^{R(S)}$, LET: $\text{mis}(u)$ be the number of examples that a description extending u misclassifies. Then $\text{mis}(u)$ is well defined, and $E_S(V) = 2^{N-n_S} \sum_u V^{\text{mis}(u)}$.

PROOF: The label $h_w(x_i)$ depends only on $w|_{R(x_i)}$, and $R(x_i) \subseteq R(S)$. Each u has 2^{N-n_S} extensions.

⟨1⟩2. LET: Φ_S be the factor graph on the variables $R(S)$ over $\mathcal{B}_{P,\ell}$ with $P = n_S + 1$ and $\ell = m + 1$ and with one factor for each example i: scope $R(x_i)$, and table 1 at the patterns α with $f_{x_i}(\alpha) = y_i$ and V elsewhere. Then $Z(\Phi_S)$ determines the integer polynomial $\sum_u V^{\text{mis}(u)}$.

PROOF: Each u contributes $V^{\text{mis}(u)}$. Because $\text{mis}(u) \leq m < \ell$, no term is truncated. Each coefficient is at most $2^{n_S} < 2^P$, so its residue modulo 2^P determines it.

⟨1⟩3. The factor graph Φ_S satisfies the hypotheses of Theorem 2.7 with $s = \max\{4\kappa m, m+1, \kappa^2\}$.

PROOF: The size is $\|\Phi_S\| \leq \kappa m$, so $4\|\Phi_S\| \leq s$. Also $P \leq \kappa m + 1 \leq s$, $\ell = m + 1 \leq s$, and $s \geq 2$. The arity is at most $\kappa \leq s^{1/2} \leq s^{9/10}$. The entries take polynomial time.

⟨1⟩4. Fixing a set of description bits to given values preserves ⟨1⟩1–⟨1⟩3.

PROOF: Fixed bits in $R(S)$ restrict scopes and tables, and factors whose scope becomes empty are constants (Theorem 2.7). Each fixed bit outside $R(S)$ halves the factor 2^{N-n_S}. Sizes, P and ℓ do not grow.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩2–⟨1⟩4, Theorem 2.7 computes the restricted profile in time $\text{poly}(N, m) \cdot 2^{O(s/\sqrt{\log s})}$. Multiply by the power of 2 from ⟨1⟩1.

□

C Proofs for Section 3: the locality transition

C.1 Realizable transfer

Theorem 3.2 (Realizable transfer)**.** *Let $\mathcal{H}$ be κ-local with description length N, and let S be a sample of size m.*

(a) Counting. *The number $|\mathrm{VS}(S)|$ can be computed deterministically in time $\mathrm{poly}(N,m)\cdot 2^{\kappa}\cdot T_{\kappa}(N)$.*

(b) Sampling. *If $\mathrm{VS}(S)\neq\emptyset$, a uniformly random element of $\mathrm{VS}(S)$ can be generated exactly in the same time, given uniform random integers in $\{0,\dots,M-1\}$ at cost $\mathrm{poly}(\log M)$. With fair coins only, the same bound holds for the expected time of a Las Vegas algorithm.*

(c) Proper PAC learning. *For all $\varepsilon,\delta\in(0,1)$, the class $\mathcal{H}$ is properly (ε,δ)-PAC learnable from $\lceil m_{\mathrm{Occ}}\rceil$ examples by an algorithm that is deterministic after sampling and runs in the same time.*

(d) Online learning. *In the realizable mistake-bound model, the halving algorithm makes at most N mistakes, and round t takes time $\mathrm{poly}(N,t)\cdot 2^{\kappa}\cdot T_{\kappa}(N)$.*

All four run in time $\mathrm{poly}(N,m)\cdot 2^{o(N)}$ when $\kappa=o((\log N)^{1/4}/\sqrt{\log\log N})$.

Proof of Theorem 3.2. ⟨1⟩1. For every κ-CNF Γ on $w_1,\dots,w_N$ with M' clauses, the number $\#\Gamma$ can be computed deterministically in time $\mathrm{poly}(N,M')\cdot 2^{\kappa}\cdot T_{\kappa}(N)$.

⟨2⟩1. Let: N' be the number of variables that occur in Γ. Then $\#\Gamma=2^{N-N'}\cdot\#'\Gamma$, where $\#'\Gamma$ counts the satisfying assignments of the occurring variables.

Proof: Each variable that does not occur doubles the count.

⟨2⟩2. Case: $N'<16$ or $M'\leq 1$. Brute force over $2^{N'}$ assignments computes $\#'\Gamma$ in time $\mathrm{poly}(N,M')\cdot 2^{\max\{16,\kappa\}}$.

Proof: If $M'\leq 1$, all occurring variables lie in one clause, so $N'\leq\kappa$.

⟨2⟩3. Case: $N'\geq 16$ and $M'\geq 2$. Then $\#'\Gamma$ can be computed in time $\mathrm{poly}(N,M')\cdot T_{\kappa}(N)$.

Proof: Imported Theorem 2.4 with $k=\kappa$ and $n=N'\geq 4$ takes time at most $2^{C_1\kappa^2 g(N')}\leq 2^{C_1\kappa^2 g(N)}$, because g is nondecreasing. Brute force takes time $2^{N'}\mathrm{poly}(M')\leq 2^{N}\mathrm{poly}(M')$. Run the faster of the two.

⟨2⟩4. Q.E.D.

Proof: The two cases are exhaustive. Combine them with ⟨2⟩1.

⟨1⟩2. Part (a) holds.

Proof: Build Γ_S by Lemma 3.1, with $M'\leq m2^{\kappa}$ clauses, and apply ⟨1⟩1.

⟨1⟩3. Part (b) holds.

⟨2⟩1. Let: the algorithm fix $w_1,\dots,w_N$ in order. If $w_1,\dots,w_{i-1}$ are fixed to $b_1,\dots,b_{i-1}$, it computes, for $b\in\{0,1\}$, the number c_b of elements of $\mathrm{VS}(S)$ that extend $(b_1,\dots,b_{i-1},b)$, and it sets $w_i=b$ with probability $c_b/(c_0+c_1)$.

⟨2⟩2. Each count c_b takes time $\mathrm{poly}(N,m)\cdot 2^{\kappa}\cdot T_{\kappa}(N)$.

Proof: The count c_b is $\#\Gamma$ for Γ_S plus at most N unit clauses, which is a κ-CNF because $\kappa\geq 1$. Apply ⟨1⟩1.

⟨2⟩3. Every $w\in\mathrm{VS}(S)$ is output with probability $1/|\mathrm{VS}(S)|$.

Proof: The probability is a telescoping product of the ratios $c_{w_i}/(c_0+c_1)$, whose first denominator is $|\mathrm{VS}(S)|$ and whose last numerator is 1.

⟨2⟩4. Q.E.D.

PROOF: By ⟨2⟩2 and ⟨2⟩3, with $2N$ counts. The choice of w_i takes one uniform integer U below $c_0 + c_1 \le 2^N$, and sets $w_i = 0$ if and only if $U < c_0$. With fair coins, rejection sampling draws such an integer with an expected number of $O(N)$ coins.

⟨1⟩4. Part (c) holds.

PROOF: Draw $m = \lceil m_{\mathrm{Occ}} \rceil$ examples. In the realizable case $\mathrm{VS}(S) \neq \emptyset$. Run the procedure of ⟨1⟩3, but set w_i to any b with $c_b > 0$; the result lies in $\mathrm{VS}(S)$. By Imported Theorem 2.1(a), its error is at most ε with probability at least $1 - \delta$.

⟨1⟩5. Part (d) holds.

PROOF: In round t, with the examples S_{t-1} of earlier rounds, compute $c_b = |\mathrm{VS}(S_{t-1} \cup \{(x_t, b)\})|$ for $b \in \{0, 1\}$ by part (a), and predict a label b with the larger count. After a mistake the version space keeps at most half of its elements. It starts with at most 2^N elements, and in the realizable model it never becomes empty. So there are at most N mistakes.

⟨1⟩6. If $\kappa = o((\log N)^{1/4}/\sqrt{\log\log N})$, then $2^\kappa \cdot T_\kappa(N) = 2^{o(N)}$.

PROOF: Here $\kappa^2 g(N)/N = \kappa^2 \log\log N/\sqrt{\log N} = o(1)$, and $\kappa = o(N)$.

⟨1⟩7. Q.E.D.

PROOF: By ⟨1⟩2–⟨1⟩6.

□

C.2 The perceptron gadget and the equivalence

Lemma 3.3 (Perceptron gadget)**.** *Let $k \ge 1$ and $p = \max\{1, k-2\}$. Given a k-CNF F on $z_1, \dots, z_N$ with M clauses, none of them tautological, one can compute in polynomial time a sample S_F of $M + p$ examples in $\{-1, 0, 1\}^{N+p}$, all labeled 1 and each with support of size at most $\max\{2k-2, 2\}$, with the following property. A sign vector σ is consistent with S_F as a binarized perceptron if and only if $\sigma_{N+1} = \cdots = \sigma_{N+p} = 1$ and the assignment $z_i = \mathbf{1}[\sigma_i = 1]$ satisfies F. In particular, $|\mathrm{VS}_{\mathsf{BP}}(S_F)| = \#F$.*

Proof of Lemma 3.3. ⟨1⟩1. LET: the coordinates $1, \dots, N$ be *variable coordinates* and $N+1, \dots, N+p$ *padding coordinates*. LET: S_F consist of the *pinning examples* e_{N+j} for $j \le p$ and, for each clause C with $r = |C|$ literals, the example x_C defined as follows. On the variable coordinates, x_C has entry $+1$ at i if $z_i \in C$, entry -1 if $\neg z_i \in C$, and 0 elsewhere. On the padding coordinates, x_C has entry $+1$ at $N+1, \dots, N+r-2$ if $r \ge 2$, entry -1 at $N+1$ if $r = 1$, and 0 elsewhere. All labels are 1.

PROOF: The clause is not tautological, so its r literals are on r distinct variables, and x_C is well defined. It uses at most $\max\{1, k-2\} = p$ padding coordinates.

⟨1⟩2. Every example has support of size at most $\max\{2k-2, 2\}$.

PROOF: A clause of width $r \ge 2$ gives support $r + (r-2) \le 2k-2$, a clause of width 1 gives support 2, and a pinning example has support 1.

⟨1⟩3. A pinning example e_{N+j} is consistent with σ if and only if $\sigma_{N+j} = 1$.

PROOF: Its label is 1 and $\langle \sigma, e_{N+j} \rangle = \sigma_{N+j}$.

⟨1⟩4. ASSUME: $\sigma_{N+1} = \cdots = \sigma_{N+p} = 1$. LET: $z_i = \mathbf{1}[\sigma_i = 1]$, and LET: t be the number of literals of C that z makes true. PROVE: $\langle \sigma, x_C \rangle = 2t - 2$.

PROOF: On a variable coordinate of C, the product of the entry of x_C and σ_i is $+1$ for a true literal and -1 for a false one, so the variable part is $t - (r - t) = 2t - r$. The padding part is $r - 2$: it is a sum of $r - 2$ ones if $r \ge 2$, and it is $-1 = r - 2$ if $r = 1$.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩3 and ⟨1⟩4, σ is consistent with S_F if and only if its padding is pinned and $2t - 2 \geq 0$, that is $t \geq 1$, for every clause. This means that z satisfies F. The map from pinned σ to z is a bijection onto $\{0,1\}^N$, so $|\mathrm{VS}_{\mathsf{BP}}(S_F)| = \#F$. The construction takes polynomial time, and ⟨1⟩2 bounds the supports.

□

Theorem 3.4 (Locality equivalence)**.** *Let $\kappa, k \geq 1$, and let T and T' be nondecreasing time bounds.*

(i) If #κ-SAT on n variables with M clauses can be solved in time $T(n, M)$, then for every κ-local class the version space of a sample of size m can be counted in time $T(N, m2^\kappa) + \mathrm{poly}(N, m) \cdot 2^\kappa$.

(ii) If version spaces of BP_{2k} samples of size m in dimension n can be counted (respectively, decided nonempty) in time $T'(n, m)$, then #k-SAT (respectively, k-SAT) on N variables with M clauses can be solved in time $T'(N + k, M + k) + \mathrm{poly}(N, M)$.

(iii) If BP_{2k} in dimension n has a proper $(\varepsilon, 1/3)$-PAC learner that runs in time $T'(n, 1/\varepsilon)$, then k-SAT on N variables with M clauses has a randomized algorithm with one-sided error that runs in time $T'(N + k, 2(M + k)) \cdot \mathrm{poly}(N, M)$.

Proof of Theorem 3.4. ⟨1⟩1. Part (i) holds.

PROOF: Lemma 3.1 builds the κ-CNF Γ_S on N variables with at most $m2^\kappa$ clauses in time $\mathrm{poly}(N, m) \cdot 2^\kappa$, and $|\mathrm{VS}(S)| = \#\Gamma_S$. The assumed algorithm counts it in time $T(N, m2^\kappa)$, because T is nondecreasing.

⟨1⟩2. Part (ii) holds.

PROOF: Let F be a k-CNF on N variables with M clauses. Deleting tautological clauses changes neither $\#F$ nor satisfiability. Lemma 3.3 gives a sample of at most $M + k$ examples in dimension $N + p \leq N + k$, with supports of size at most $\max\{2k - 2, 2\} \leq 2k$, whose version space has size $\#F$. Count it, or decide whether it is nonempty, in time $T'(N + k, M + k)$.

⟨1⟩3. Part (iii) holds.

⟨2⟩1. LET: F be a k-CNF on N variables with M clauses, without tautological clauses, and LET: S_F be the sample of Lemma 3.3, with $M + p \leq M + k$ examples. LET: $\mathcal{D}$ be the uniform distribution on S_F. Run the learner on $\mathcal{D}$ with $\varepsilon = 1/(2(M + k))$, simulating each example draw in polynomial time, for at most $T'(N + k, 2(M + k))$ steps. If it does not halt in time, reject; otherwise check whether the output encodes a satisfying assignment.

⟨2⟩2. If F is satisfiable, the algorithm accepts with probability at least $2/3$.

PROOF: By Lemma 3.3, $\mathcal{D}$ is realizable. With probability at least $2/3$, the output σ has $\mathrm{err}_{\mathcal{D}}(\sigma) \leq \varepsilon < 1/(M + p)$. Errors under $\mathcal{D}$ are multiples of $1/(M + p)$, so σ is consistent with every example, and by Lemma 3.3 it encodes a satisfying assignment.

⟨2⟩3. If F is unsatisfiable, the algorithm rejects.

PROOF: No assignment passes the check.

⟨2⟩4. Q.E.D.

PROOF: By ⟨2⟩2 and ⟨2⟩3, the error is one-sided. The learner runs in dimension $N + p \leq N + k$ with $1/\varepsilon = 2(M + k)$, and each step costs at most $\mathrm{poly}(N, M)$ for the simulation.

⟨1⟩4. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩3.

□

C.3 Width reduction

We use the branching procedure of Schuler [Sch05] in the form of [CIP06, Section 3.2]. For $k \geq 1$, the procedure $\textsc{Reduce}_k(F)$ works as follows on a CNF F without tautological clauses. After every assignment it deletes satisfied clauses and removes false literals; no step creates a tautological clause.

- If F contains the empty clause, stop; this leaf contributes nothing. If F has no clause of width larger than k, output F as a leaf.
- Otherwise let C be the first clause of width larger than k, and let $C' \subset C$ consist of its first k literals. Recurse on the *left branch* $F_{\mathrm{L}} = (F \setminus \{C\}) \cup \{C'\}$ and on the *right branch* F_{R}, which is F with the k literals of C' set to false.

Claim C.1. *The satisfying assignments of F are the disjoint union of those of F_{L} and of those of F_{R} extended by the assignment that falsifies C'.*

Proof of Claim C.1. ⟨1⟩1. Every satisfying assignment x of F lies in exactly one of the two sets.
PROOF: If x satisfies C', then x satisfies F_{L}, and x does not falsify C'. If x falsifies C', then x does not satisfy F_{L}, and its restriction satisfies F_{R}.

⟨1⟩2. Every element of the two sets satisfies F.
PROOF: F_{L} implies F because $C' \subseteq C$. An assignment that satisfies F_{R} and falsifies C' satisfies every clause of F.

⟨1⟩3. Q.E.D.
PROOF: By ⟨1⟩1 and ⟨1⟩2.

□

By induction, $\#F$ is the sum over the leaves of the number of satisfying assignments of the leaf formula over the variables that are still free at the leaf, and F is satisfiable if and only if some leaf formula is.

Claim C.2. *Let F have n variables and m clauses. Every leaf is reached along a path with at most m left branches and $r \leq \lfloor n/k \rfloor$ right branches, and its formula is a k-CNF on at most $n - kr$ variables. At most $\binom{m+r}{r}$ leaves have exactly r right branches.*

Proof of Claim C.2. ⟨1⟩1. Every path has at most m left branches.
PROOF: A left branch replaces a clause of width larger than k by one of width k. Right branches and simplifications never widen or create a clause. So the number of clauses of width larger than k drops at every left branch and never grows.

⟨1⟩2. A path with r right branches has $r \leq n/k$, and its leaf has at most $n - kr$ free variables.
PROOF: Each right branch fixes k free variables.

⟨1⟩3. At most $\binom{m+r}{r}$ leaf paths have exactly r right branches.
PROOF: Pad each such path with trailing left branches to length $m + r$. No leaf path is a prefix of another, so two distinct paths with the same padding would be a prefix of one another. So the padding is injective into the words of length $m + r$ with r right branches.

⟨1⟩4. Q.E.D.
PROOF: By ⟨1⟩1–⟨1⟩3; a leaf formula has no clause of width larger than k.

□

Lemma 3.5 (Width-reduction transfer)**.** *Let $\rho \in (0, 1]$, $c \geq 1$, $K_0 = 2(c + 1)/\rho$ and $K' = 2K_0$. Suppose that k-SAT (respectively, #k-SAT) on n variables with $k \leq K' \log n$ and at most n^{2c} clauses can be solved in time $O(2^{\rho n})$ with error probability at most $1/3$. Then CNF-SAT (respectively,*

#CNF-SAT) on n variables with at most n^c clauses can be solved in time $O(2^{\rho n}\mathrm{poly}(n))$ with error probability at most $1/3$.

Proof of Lemma 3.5. ⟨1⟩1. LET: F have n variables and $m \le n^c$ clauses, none of them tautological, and LET: $k = \lfloor K_0 \log n \rfloor$. Run $\textsc{Reduce}_k(F)$, and solve each leaf as follows. A leaf with $n' \ge \sqrt{n}$ free variables is solved by the assumed algorithm, repeated $O(n)$ times with a majority vote, or the most frequent answer for counting. A leaf with $n' < \sqrt{n}$ free variables is solved by brute force. Combine the leaf answers by Claim C.1.

PROOF: Deleting tautological clauses changes neither satisfiability nor the count. Small n is handled by brute force.

⟨1⟩2. The assumed algorithm applies to every leaf with $n' \ge \sqrt{n}$, and each repeated run has error probability at most 2^{-2n}.

PROOF: Here $\log n' \ge \frac{1}{2}\log n$, so the width is $k \le K_0 \log n \le K' \log n'$. The number of clauses is at most $m \le n^c \le n'^{2c}$. Repetition with a majority vote reduces the error to 2^{-2n}.

⟨1⟩3. The large leaves cost $O(n) \cdot \sum_r \binom{m+r}{r} O(2^{\rho(n-kr)}) \le O(n) \cdot e \cdot 2^{\rho n}$ in total.

PROOF: By Claim C.2, $\binom{m+r}{r} \le \binom{m+\lfloor n/k \rfloor}{r}$, and the binomial theorem gives

$$\sum_{r=0}^{\lfloor n/k \rfloor} \binom{m+r}{r} 2^{\rho(n-kr)} \le 2^{\rho n}\left(1+2^{-\rho k}\right)^{m+n/k} \le 2^{\rho n} \exp\left(\left(m+n/k\right) 2^{-\rho k}\right) \le e \cdot 2^{\rho n}.$$

The last step uses $k \ge K_0 \log n - 1$, so $2^{-\rho k} \le 2n^{-\rho K_0} = 2n^{-2(c+1)}$, and $2(m+n/k)n^{-2(c+1)} \le 2(n^c+n)n^{-2c-2} \le 1$ for $n \ge 2$.

⟨1⟩4. The tree has at most $2^{(\rho/2+o(1))n}$ leaves, and the small leaves cost $O(2^{\rho n})$ in total.

PROOF: By Claim C.2, the number of leaves is at most

$$\sum_{r \le n/k} \binom{m+r}{r} \le (n+1)\binom{m+\lfloor n/k \rfloor}{\lfloor n/k \rfloor} \le (n+1)\left(\frac{e(m+n/k)}{n/k}\right)^{n/k} \le (n+1)\left(2ekn^c\right)^{n/k}.$$

Its logarithm is at most $\frac{n}{K_0 \log n - 1}(c \log n + \log(2ek)) + \log(n+1) = (c/K_0 + o(1))\, n \le (\rho/2 + o(1))\, n$, because $c/K_0 = c\rho/(2(c+1)) \le \rho/2$. Each small leaf costs $2^{\sqrt{n}}\mathrm{poly}(n)$.

⟨1⟩5. All leaf answers are correct with probability at least $1 - 2^{-n}$.

PROOF: By ⟨1⟩2, ⟨1⟩4 and a union bound over at most 2^n leaves.

⟨1⟩6. Q.E.D.

PROOF: The tree has at most twice as many nodes as leaves, each with $\mathrm{poly}(n,m)$ work. With ⟨1⟩3 and ⟨1⟩4, the total time is $O(2^{\rho n}\mathrm{poly}(n))$. By ⟨1⟩5 and Claim C.1, the combined answer is correct with probability at least $1 - 2^{-n} \ge 2/3$.

□

C.4 Hardness at logarithmic locality

Corollary 3.6. *(a) Under Hypothesis 2.9, for every $\gamma \in (0,1)$ there are K', c' such that k-SAT with $k \le K' \log n$ cannot be solved in time $O(2^{(1-\gamma)n})$, even on instances with at most $n^{c'}$ clauses. The same holds for counting under Hypothesis 2.10.*

(b) Under Hypothesis 2.11, there are $K', c', \rho > 0$ such that k-SAT with $k \le K' \log n$ and at most $n^{c'}$ clauses cannot be solved in time $O(2^{\rho n})$.

Proof of Corollary 3.6. ⟨1⟩1. Part (a) holds.

PROOF: Let c be the constant that Hypothesis 2.9 gives for $\gamma/2$, and put $c' = 2c$ and $\rho = 1-\gamma$; let K' be as in Lemma 3.5. An $O(2^{(1-\gamma)n})$ algorithm for $k \le K' \log n$ and at most $n^{c'}$ clauses would, by Lemma 3.5, solve CNF-SAT with at most n^c clauses in time $O(2^{(1-\gamma)n}\text{poly}(n)) \subseteq O(2^{(1-\gamma/2)n})$, which contradicts Hypothesis 2.9. The counting statement follows in the same way from Hypothesis 2.10.

⟨1⟩2. Part (b) holds.

PROOF: Let c, ρ_0 be the constants of Hypothesis 2.11, and put $c' = 2c$ and $\rho = \rho_0/2$. An $O(2^{\rho n})$ algorithm would solve CNF-SAT with at most n^c clauses in time $O(2^{\rho n}\text{poly}(n)) \subseteq O(2^{\rho_0 n})$.

⟨1⟩3. Q.E.D.

PROOF: By ⟨1⟩1 and ⟨1⟩2.

□

Theorem 3.8 (Hardness at logarithmic locality)**.** *Assume Hypothesis 2.9. For every $\gamma \in (0,1)$ there are constants K and c_γ with the following properties. No randomized algorithm decides in time $O(2^{(1-\gamma)n})$ whether a sample of size at most n^{c_γ} in dimension n is consistent with some binarized perceptron on $\lceil K \log n \rceil$-sparse data. No proper $(n^{-c_\gamma}, 1/3)$-PAC learner for $\mathsf{BP}_{\lceil K \log n \rceil}$ runs in time $O(2^{(1-\gamma)n})$. Under Hypothesis 2.10, the same holds for counting version spaces. Under Hypothesis 2.11, there are $K, \rho > 0$ with the same statements for time $O(2^{\rho n})$.*

Proof of Theorem 3.8. ⟨1⟩1. LET: K', c' be given by Corollary 3.6(a) for $\gamma/2$, and LET: $K = 2K'+2$ and $c_\gamma = c'+2$. LET: F be a k-CNF on N variables with $k = \lfloor K' \log N \rfloor$ and at most $N^{c'}$ clauses, and LET: S_F be the sample of Lemma 3.3, in dimension $n = N + p$.

⟨1⟩2. The sample S_F is a $\mathsf{BP}_{\lceil K \log n \rceil}$ sample of size at most n^{c_γ}, and $n \le N + K' \log N$.

PROOF: The sparsity is at most $2k \le K \log n$, the size is at most $N^{c'} + k \le n^{c'} + n \le n^{c'+2}$, and $p \le k$.

⟨1⟩3. A consistency test in time $O(2^{(1-\gamma)n})$ contradicts Corollary 3.6(a).

PROOF: It decides F in time $O(2^{(1-\gamma)(N+K' \log N)}) + \text{poly}(N) \subseteq O(2^{(1-\gamma/2)N})$.

⟨1⟩4. A proper $(n^{-c_\gamma}, 1/3)$-PAC learner in time $O(2^{(1-\gamma)n})$ contradicts Corollary 3.6(a).

PROOF: The argument of Theorem 3.4(iii) needs an accuracy below $1/(M+p)$, and $n^{-c_\gamma} \le 1/(2(n^{c'}+n)) \le 1/(2(M+k))$. It gives a one-sided randomized algorithm for F with the time bound of ⟨1⟩3.

⟨1⟩5. The counting and CNF-ETH statements hold.

PROOF: Use Theorem 3.4(ii) with the counting part of Corollary 3.6(a), and Corollary 3.6(b), in the same way.

⟨1⟩6. Q.E.D.

PROOF: By ⟨1⟩3–⟨1⟩5.

□

Corollary 3.9 (The data-sparsity transition)**.** *Consider binarized perceptrons on k-sparse data in dimension n, with samples of polynomial size.*

(a) If $k = o((\log n)^{1/4}/\sqrt{\log\log n})$, then consistency, version-space counting, uniform sampling, and proper PAC learning at any accuracy take time $\text{poly}(m) \cdot 2^{o(n)}$.

(b) Under Hypothesis 2.9, for every $\gamma \in (0,1)$ there are K and c_γ such that for $k \ge K \log n$, consistency and proper PAC learning to accuracy n^{-c_γ} cannot be done in time $O(2^{(1-\gamma)n})$. Under Hypothesis 2.10, the same holds for version-space counting.

The same dichotomy holds for literal disjunctions LD_k. There, a sample of positive examples is literally a k-CNF, and (b) follows from Corollary 3.6 without a gadget.

Proof of Corollary 3.9. ⟨1⟩1. Part (a) holds.

PROOF: The class BP_k is k-local with $N = n$, and consistency is the question whether $|\mathrm{VS}(S)| > 0$. Apply Theorem 3.2 with $\kappa = k$.

⟨1⟩2. Part (b) holds.

PROOF: By Theorem 3.8. A $\lceil K \log n \rceil$-sparse sample is also k-sparse for every integer $k \geq K \log n$, so the hardness extends to all such k.

⟨1⟩3. The statements hold for LD_k.

PROOF: Part (a) holds as in ⟨1⟩1. For (b), a k-CNF without tautological clauses is a sample of positive LD_k examples (Example 2.3(a)), so Corollary 3.6 applies directly, and the argument of Theorem 3.4(iii) covers learners.

⟨1⟩4. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩3.

□

C.5 Sample-sensitive realizable learning

Theorem 3.10 (Sample-sensitive realizable learning)**.** *Let $\mathcal{H}$ be κ-local with description length N and VC dimension $d \geq 1$, and let $\varepsilon, \delta \in (0,1)$. With $m = \lceil m_{\mathrm{VC}} \rceil$ examples, $\mathcal{H}$ is properly (ε, δ)-PAC learnable by an algorithm that is deterministic after sampling and runs in time*

$$\mathrm{poly}(N, m) \cdot \exp\Big(O\Big(\frac{\kappa\,(d \log(2/\varepsilon) + \log(2/\delta))}{\varepsilon \sqrt{\log(2d/\varepsilon)}} + \kappa^2\Big)\Big).$$

For constant ε and δ the time is $2^{O(\kappa N/\sqrt{\log N} + \kappa^2)}$, which is $2^{o(N)}$ whenever $\kappa = o(\sqrt{\log N})$.

Proof of Theorem 3.10. ⟨1⟩1. LET: S be a sample of $m = \lceil m_{\mathrm{VC}} \rceil$ examples. With probability at least $1 - \delta$, every $w \in \mathrm{VS}(S)$ has $\mathrm{err}_{\mathcal{D}}(w) \leq \varepsilon$.

PROOF: Imported Theorem 2.1(b).

⟨1⟩2. With probability 1, $\mathrm{VS}(S) \neq \emptyset$.

PROOF: Because $\mathrm{OPT} = 0$, some w^* has $\mathrm{err}_{\mathcal{D}}(w^*) = 0$, so each example is consistent with w^* with probability 1.

⟨1⟩3. If $\mathrm{VS}(S) \neq \emptyset$, the following procedure outputs an element of $\mathrm{VS}(S)$ after at most $\kappa m + 1$ profile computations. Process the bits of $R(S)$ in order, and fix each to a value for which the coefficient of V^0 in the profile of the descriptions that extend the current partial assignment stays positive. Set all other bits to 0.

PROOF: The coefficient is positive at the start, and at each step it is the sum of the coefficients for the two values, so one value keeps it positive. At the end all bits of $R(S)$ are fixed, and a positive coefficient of V^0 means that the fixed bits make no error. The other bits do not affect any label.

⟨1⟩4. Each profile computation takes time $\mathrm{poly}(N, m) \cdot 2^{O(s/\sqrt{\log s})}$ with $s = \max\{4\kappa m, m + 1, \kappa^2\} \leq 6\kappa m + \kappa^2$, and $\log s \geq \log(2d/\varepsilon) \geq 1$.

PROOF: The time is Lemma 2.8. For the bound on $\log s$: because $C_B \geq 1$ and $\log(2/\varepsilon) \geq 1$, we have $s \geq 4m \geq 4d/\varepsilon \geq 2d/\varepsilon \geq 2$.

⟨1⟩5. The total time satisfies the displayed bound.

PROOF: By ⟨1⟩4, $s/\sqrt{\log s} \le (6\kappa m + \kappa^2)/\sqrt{\log(2d/\varepsilon)}$, and $m = O((d\log(2/\varepsilon) + \log(2/\delta))/\varepsilon)$. The factor $\kappa m + 1$ of ⟨1⟩3 is polynomial.

⟨1⟩6. For constant ε and δ the time is $2^{O(\kappa N/\sqrt{\log N} + \kappa^2)}$.

PROOF: Here $m = O(d)$ and the exponent is $O(\kappa d/\sqrt{\log(2d/\varepsilon)} + \kappa^2)$. The function $x \mapsto x/\sqrt{\log x}$ is increasing for $x \ge 2$, and $d \le N$, so $d/\sqrt{\log(2d/\varepsilon)} = O(N/\sqrt{\log N})$.

⟨1⟩7. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩6. The exponent is $o(N)$ when $\kappa = o(\sqrt{\log N})$, because then also $\kappa^2 = o(\log N)$.

□

D Proofs for Section 4: the accuracy transition

D.1 Exact ERM, agnostic learning and Max-CSP

Theorem 4.1 (Exact ERM)**.** *For a κ-local class and any sample S of size m, a minimizer of err_S, the number of minimizers and the entire error profile can be computed in time $\mathrm{poly}(N, m)\cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\kappa m, m+1, \kappa^2\}$. In particular, for constant κ and $m = O(N)$, exact ERM takes time $2^{O(N/\sqrt{\log N})}$.*

Proof of Theorem 4.1. ⟨1⟩1. The profile E_S, the minimum number j^* of errors, and the number c_{j^*} of minimizers can be computed in the stated time.

PROOF: Lemma 2.8 gives $E_S = \sum_j c_j V^j$, and $j^* = \min\{j : c_j \ne 0\}$.

⟨1⟩2. The following procedure finds a minimizer with at most $\kappa m + 1$ further profile computations. Process the bits of $R(S)$ in order, and fix each to a value for which the profile of the descriptions that extend the current partial assignment has a nonzero coefficient at V^{j^*}. Set all other bits to 0.

⟨2⟩1. At every step some value keeps the coefficient at V^{j^*} nonzero.

PROOF: The profile before the step is the sum of the two restricted profiles, and all coefficients are nonnegative.

⟨2⟩2. The output has exactly j^* errors.

PROOF: At the end all bits of $R(S)$ are fixed, and a nonzero coefficient at V^{j^*} means that these bits make exactly j^* errors. The other bits do not affect any label.

⟨2⟩3. Q.E.D.

PROOF: By ⟨2⟩1 and ⟨2⟩2, with one profile computation per bit of $R(S)$.

⟨1⟩3. For constant κ and $m = O(N)$, the time is $2^{O(N/\sqrt{\log N})}$.

PROOF: Then $s = O(N)$, and $x \mapsto x/\sqrt{\log x}$ is increasing for $x \ge 2$.

⟨1⟩4. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩3 and Lemma 2.8.

□

Corollary 4.2 (Consistency is SAT by length)**.** *Assume $\mathrm{ETH}^{\mathrm{len}}_C$, and let $\theta > 0$. No randomized algorithm with error probability at most $1/3$ decides in time $2^{m/(\log m)^{C+\theta}}$ whether a sample of m positive examples of LD_3 is consistent. So for 3-local classes, Theorem 4.1 is optimal up to a factor $(\log m)^{C-1/2+\theta}$ in the exponent.*

Proof of Corollary 4.2. ⟨1⟩1. LET: F be a CNF of length L. LET: F' arise from F by replacing each clause $(\lambda_1 \vee \cdots \vee \lambda_r)$ with $r \ge 4$ by the chain of step ⟨1⟩3 in the proof of Lemma 4.8, with fresh

variables, and by deleting tautological clauses. Then F' is satisfiable if and only if F is, its clauses have width at most 3, and it has at most L clauses.

PROOF: A satisfying assignment extends to the fresh variables of each chain, and an assignment that falsifies a clause falsifies some clause of its chain; see steps ⟨1⟩5 and ⟨1⟩6 of the proof of Lemma 4.8. A clause of width $r \geq 4$ yields $r - 2 \leq r$ clauses.

⟨1⟩2. The sample $S = \{(x_C, 1) : C \in F'\}$ consists of $m \leq L$ positive examples of LD_3, and it is consistent if and only if F is satisfiable.

PROOF: By Example 2.3(a), the hypothesis σ labels x_C positive if and only if σ satisfies C. Apply ⟨1⟩1.

⟨1⟩3. An algorithm as excluded in the statement decides F in time $2^{L/(\log L)^{C+\theta}} + \mathrm{poly}(L) < 2^{L/(\log L)^C}$ for all large L.

PROOF: The function $x \mapsto x/(\log x)^{C+\theta}$ is increasing for large x, $m \leq L$, and the reduction takes polynomial time.

⟨1⟩4. For $\kappa = 3$, Theorem 4.1 decides consistency on m examples in time $\mathrm{poly}(N) \cdot 2^{O(m/\sqrt{\log m})}$.

PROOF: Here $s = \max\{12m, m+1, 9\} = O(m)$ and $\log s \geq \log m$.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩3, such an algorithm contradicts $\mathrm{ETH}^{\mathrm{len}}_C$. By ⟨1⟩4, the exponents of the upper and the lower bound differ by a factor of at most $O((\log m)^{C-1/2+\theta})$.

□

Theorem 4.3 (Agnostic transfer)**.** *Let $\mathcal{H}$ be κ-local with description length N and VC dimension d, and let $\varepsilon, \delta \in (0,1)$. Exact ERM on $m = \lceil m_{\mathrm{UC}} \rceil$ examples is a proper (ε, δ)-agnostic learner, and it runs deterministically after sampling in time*

$$\mathrm{poly}(N, m) \cdot \exp\Big(O\Big(\frac{\kappa(d + \log(1/\delta))}{\varepsilon^2 \sqrt{\log(2d/\varepsilon)}} + \kappa^2\Big)\Big).$$

If $\kappa/\varepsilon^2 = o(\sqrt{\log N})$ and δ is constant, the time is $2^{o(N)}$.

Proof of Theorem 4.3. ⟨1⟩1. With probability at least $1 - \delta$, every minimizer of err_S on $m = \lceil m_{\mathrm{UC}} \rceil$ examples is ε-optimal.

PROOF: Imported Theorem 2.1(c).

⟨1⟩2. Theorem 4.1 finds a minimizer in time $\mathrm{poly}(N, m) \cdot 2^{O(s/\sqrt{\log s})}$ with $s = \max\{4\kappa m, m+1, \kappa^2\} \leq 6\kappa m + \kappa^2$, and $\log s \geq \log(2d/\varepsilon) \geq 1$.

PROOF: Because $C_0 \geq 1$ and $\varepsilon < 1$, we have $s \geq 4\kappa m \geq 4d/\varepsilon^2 \geq 2d/\varepsilon \geq 2$.

⟨1⟩3. The time satisfies the displayed bound.

PROOF: By ⟨1⟩2, $s/\sqrt{\log s} \leq (6\kappa m + \kappa^2)/\sqrt{\log(2d/\varepsilon)}$, and $m = O((d + \log(1/\delta))/\varepsilon^2)$.

⟨1⟩4. If $\kappa/\varepsilon^2 = o(\sqrt{\log N})$ and δ is constant, the time is $2^{o(N)}$.

⟨2⟩1. CASE: $d \geq \sqrt{N}$. Then $\log(2d/\varepsilon) \geq \frac{1}{2}\log N$, and the exponent is $O(\kappa N/(\varepsilon^2\sqrt{\log N})) + O(\kappa^2) = o(N)$.

PROOF: Use $d \leq N$ and $\kappa^2 \leq (\kappa/\varepsilon^2)^2 = o(\log N)$.

⟨2⟩2. CASE: $d < \sqrt{N}$. Then the exponent is $O(\kappa\sqrt{N}/\varepsilon^2) + O(\kappa^2) = o(N)$.

PROOF: Use $\log(2d/\varepsilon) \geq 1$ and $\kappa/\varepsilon^2 = o(\sqrt{\log N})$.

⟨2⟩3. Q.E.D.

PROOF: The cases are exhaustive. The factor $\mathrm{poly}(N, m)$ is $2^{o(N)}$, because $1/\varepsilon^2 = o(\sqrt{\log N})$.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩4.

□

Corollary 4.4 (Sparse Max-CSP)**.** *Let Φ be a Boolean constraint system on n variables with M constraints of arity at most k. For every j, the number of assignments that violate exactly j constraints can be computed in time $\mathrm{poly}(n, M) \cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4kM, n+1, M+1, k^2\}$. In particular, for every constant c, Max-3-SAT with n variables and at most cn clauses can be solved in time $2^{O((1+c)n/\sqrt{\log n})}$. With positive integer clause weights at most W, the same holds with $s = \max\{4kM, n+1, WM+1, k^2\}$, and Max-3-SAT takes time $2^{O((1+cW)n/\sqrt{\log n})}$.*

Proof of Corollary 4.4. ⟨1⟩1. LET: Φ' be the factor graph over $\mathcal{B}_{P,\ell}$ with $P = n+1$ and $\ell = M+1$ that has, for each constraint, a factor with entry 1 where the constraint holds and V where it fails. Then $Z(\Phi') = \sum_j c_j V^j$, where c_j is the number of assignments that violate exactly j constraints.

PROOF: Each assignment contributes V to the power of its number of violations, which is at most $M < \ell$. Each coefficient is at most $2^n < 2^P$.

⟨1⟩2. Theorem 2.7 applies with $s = \max\{4kM, n+1, M+1, k^2\}$.

PROOF: The size is at most kM, $P = n+1$, $\ell = M+1$, and the arity is at most $k \le s^{1/2}$.

⟨1⟩3. Max-3-SAT with $M \le cn$ clauses takes time $2^{O((1+c)n/\sqrt{\log n})}$.

PROOF: Here $s \le \max\{12cn, n+1, cn+1, 9\} \le (12c+1)n+10$ and $\log s \ge \log n$. The maximum number of satisfied clauses is $M - \min\{j : c_j \neq 0\}$.

⟨1⟩4. The weighted statements hold.

PROOF: Give a violated clause C the entry $V^{\mathrm{wt}(C)}$ and take $\ell = WM+1$, which exceeds every total violated weight. Then $s = \max\{4kM, n+1, WM+1, k^2\}$, and for Max-3-SAT $s \le (12c + cW + 1)n + 10$.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩4.

□

D.2 The exponential mechanism

Theorem 4.6 (Exact private learning)**.** *Let $\mathcal{H}$ be κ-local and $\varepsilon_{\mathrm{dp}} > 0$. On a sample S of size m, one can sample w with probability proportional to $\exp(-\varepsilon_{\mathrm{dp}}\, m\, \mathrm{err}_S(w)/2)$ in time $\mathrm{poly}(N, m) \cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\kappa m, m+1, \kappa^2\}$, given exact samples from explicitly given distributions on $\{0, \dots, m\}$. This is the exponential mechanism, and it is $\varepsilon_{\mathrm{dp}}$-differentially private. Without exact sampling, the output distribution is within statistical distance η of the mechanism for any $\eta \ge 2^{-\mathrm{poly}(N,m)}$, and the output is $(\varepsilon_{\mathrm{dp}}, (1+e^{\varepsilon_{\mathrm{dp}}})\eta)$-differentially private. With $m = O\big((N + \log(1/\delta)) \max\{1/(\varepsilon\varepsilon_{\mathrm{dp}}), 1/\varepsilon^2\}\big)$, the output is ε-optimal with probability at least $1-\delta$ [KLN+11], and the time is, up to polynomial factors,*

$$\exp\Big(O\Big(\frac{\kappa\,(N + \log(1/\delta))}{\varepsilon \min\{\varepsilon, \varepsilon_{\mathrm{dp}}\}\sqrt{\log N}}\Big)\Big).$$

Proof of Theorem 4.6. ⟨1⟩1. LET: $R(S)$, n_S and $\mathrm{mis}(u)$ be as in the proof of Lemma 2.8. It suffices to sample $u \in \{0,1\}^{R(S)}$ with probability proportional to $e^{-\varepsilon_{\mathrm{dp}}\mathrm{mis}(u)/2}$ and to fill the other bits uniformly at random.

PROOF: The score of w is $m \cdot \mathrm{err}_S(w) = \mathrm{mis}(w|_{R(S)})$, and every u has 2^{N-n_S} extensions.

⟨1⟩2. The following procedure samples u exactly. Compute the restricted profile $\sum_j c'_j V^j$ with $c'_j = |\{u : \mathrm{mis}(u) = j\}|$. Sample j with probability proportional to $c'_j e^{-\varepsilon_{\mathrm{dp}} j/2}$. Then sample u uniformly from $\{u : \mathrm{mis}(u) = j\}$: fix the bits of $R(S)$ one at a time, each with probability proportional to the coefficient of V^j in the two restricted profiles.

PROOF: The second stage outputs each u with $\mathrm{mis}(u) = j$ with probability $1/c'_j$, by the telescoping argument of Theorem 3.2. So u has probability $c'_{\mathrm{mis}(u)} e^{-\varepsilon_{\mathrm{dp}} \mathrm{mis}(u)/2} / (Z' c'_{\mathrm{mis}(u)})$, where $Z' = \sum_j c'_j e^{-\varepsilon_{\mathrm{dp}} j/2}$.

⟨1⟩3. The mechanism is $\varepsilon_{\mathrm{dp}}$-differentially private.

PROOF: Changing one example changes the score of every w by at most 1, and the exponential mechanism with weights $e^{-\varepsilon_{\mathrm{dp}} \cdot \mathrm{score}/2}$ for a score of sensitivity 1 is $\varepsilon_{\mathrm{dp}}$-differentially private [MT07].

⟨1⟩4. An output distribution within statistical distance η of the mechanism, on every sample, is $(\varepsilon_{\mathrm{dp}}, (1 + e^{\varepsilon_{\mathrm{dp}}})\eta)$-differentially private, and it can be produced in time $\mathrm{poly}(N, m, \log(1/\eta))$ times the cost of the profiles.

PROOF: For neighboring samples S, S' and every event E, with μ_S the exact mechanism, $\Pr[E \mid S] \le \mu_S(E) + \eta \le e^{\varepsilon_{\mathrm{dp}}} \mu_{S'}(E) + \eta \le e^{\varepsilon_{\mathrm{dp}}} \Pr[E \mid S'] + (1 + e^{\varepsilon_{\mathrm{dp}}})\eta$. All probability in ⟨1⟩2 can be computed to any precision in time polynomial in N, m and the number of bits, so fair coins approximate each choice within any desired distance.

⟨1⟩5. With the stated m, the output is ε-optimal with probability at least $1 - \delta$.

PROOF: This is the accuracy guarantee of the exponential mechanism over a class of size at most 2^N [KLN+11].

⟨1⟩6. With the stated m, the time is as displayed.

PROOF: We may take $m \ge N \ge \kappa$. Then $s \le 5\kappa m$ and $\log s \ge \log N$, so the exponent is $O(\kappa m / \sqrt{\log N})$, and $\max\{1/(\varepsilon\varepsilon_{\mathrm{dp}}), 1/\varepsilon^2\} = 1/(\varepsilon \min\{\varepsilon, \varepsilon_{\mathrm{dp}}\})$. The procedure of ⟨1⟩2 uses at most $n_S + 1$ profile computations (Lemma 2.8).

⟨1⟩7. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩6.

□

D.3 Short PCPs and constant accuracy

Lemma 4.8 (Short PCPs as LD_3 samples)**.** *Let $\varepsilon_0 = 1/(q2^{q+2})$ and $c_q = 2q2^q c_V$. There is a deterministic polynomial-time reduction that maps a CNF F of length $L \ge 2$ to a list S_F^{pcp} of positive examples of LD_3 in dimension $N \le c_q L (\log L)^\beta$, with the following property. Let $\mathcal{D}_F$ be the uniform distribution over the entries of S_F^{pcp}. If F is satisfiable, then $\mathrm{OPT}(\mathcal{D}_F) = 0$. If F is unsatisfiable, then $\mathrm{OPT}(\mathcal{D}_F) \ge 2\varepsilon_0$.*

Proof of Lemma 4.8. ⟨1⟩1. CASE: some seed τ has $Q_\tau = \emptyset$ and $D_\tau = 0$. Then the list $((e_1, 1), (-e_1, 1))$ in dimension 1 has the required property.

PROOF: The seed τ rejects every proof, so F is unsatisfiable by completeness. Every σ labels exactly one of $e_1, -e_1$ positive, so every hypothesis has error $1/2 \ge 2\varepsilon_0$.

⟨1⟩2. CASE: no seed rejects any pattern. Then the list $((e_1, 1))$ has the required property.

PROOF: Every proof is accepted for every seed, so F is satisfiable by soundness, and $\sigma_1 = 1$ has error 0.

⟨1⟩3. CASE: neither of the above. LET: S_F^{pcp} be built as follows. For every seed τ and every pattern α on Q_τ with $D_\tau(\alpha) = 0$, LET: $C_{\tau,\alpha}$ be the clause on the proof variables that is false exactly at

α. Replace each $C_{\tau,\alpha} = (\lambda_1 \vee \cdots \vee \lambda_t)$ with $t \geq 4$ by the chain

$$(\lambda_1 \vee \lambda_2 \vee \zeta_1),\ (\neg\zeta_1 \vee \lambda_3 \vee \zeta_2),\ \ldots,\ (\neg\zeta_{t-4} \vee \lambda_{t-2} \vee \zeta_{t-3}),\ (\neg\zeta_{t-3} \vee \lambda_{t-1} \vee \lambda_t)$$

with fresh variables $\zeta_1, \ldots, \zeta_{t-3}$ for this clause, and keep clauses of width at most 3. Each resulting clause C becomes the positive example x_C of LD_3, with entry $+1$ at positive and -1 at negative literals, and S_F^{pcp} lists these examples, one entry per occurrence.

PROOF: In this case every seed with $Q_\tau = \emptyset$ accepts, so every clause $C_{\tau,\alpha}$ is nonempty. It has width at most q and no complementary literals, so each x_C is well defined, and by Example 2.3(a) the hypothesis σ labels x_C positive if and only if σ satisfies C.

⟨1⟩4. The list has at most $q2^q|\Omega_F|$ entries, and its dimension is $N \leq 2q2^q|\Omega_F| \leq c_q L(\log L)^\beta$.

PROOF: There are at most $2^q|\Omega_F|$ clauses $C_{\tau,\alpha}$. Each produces at most q entries and at most q fresh variables, and the proof has at most $q|\Omega_F|$ positions (Imported Theorem 4.7).

⟨1⟩5. If F is satisfiable, then $\mathrm{OPT}(\mathcal{D}_F) = 0$.

PROOF: Let π be a proof that is accepted for every seed. Then π satisfies every $C_{\tau,\alpha}$. In a chain, pick a true literal λ_j, and put $\zeta_i = 1$ for $i \leq j-2$ and $\zeta_i = 0$ for $i \geq j-1$. The first clause holds by λ_j if $j \leq 2$ and by ζ_1 otherwise. A middle clause $(\neg\zeta_i \vee \lambda_{i+2} \vee \zeta_{i+1})$ holds by $\neg\zeta_i$ if $i \geq j-1$, by ζ_{i+1} if $i \leq j-3$, and by λ_j if $i = j-2$. The last clause holds by $\neg\zeta_{t-3}$ if $j \leq t-2$ and by λ_j otherwise. So π extends to a hypothesis with no errors.

⟨1⟩6. If F is unsatisfiable, then every hypothesis has error at least $2\varepsilon_0$.

PROOF: Let σ assign all N variables, and let π be its restriction to the proof positions. The verifier rejects π for at least $|\Omega_F|/2$ seeds. For each of them, $C_{\tau,\alpha}$ with $\alpha = \pi|_{Q_\tau}$ is false under σ. Its chain then contains a clause that is false under σ: if all its clauses were true while all λ_j are false, the chain would force $\zeta_1 = \cdots = \zeta_{t-3} = 1$ and falsify its last clause. Distinct seeds own disjoint entries. So σ errs on at least $|\Omega_F|/2$ of at most $q2^q|\Omega_F|$ entries, an error of at least $1/(q2^{q+1}) = 2\varepsilon_0$.

⟨1⟩7. Q.E.D.

PROOF: The three cases are exhaustive, and ⟨1⟩1, ⟨1⟩2 and ⟨1⟩4–⟨1⟩6 give the property in each. The verifier runs in polynomial time, $|\Omega_F|$ is polynomial in L, and there are at most 2^q patterns per seed.

□

Theorem 4.9 (Hardness at constant accuracy)**.** *If LD_3 has a proper realizable $(\varepsilon_0, 1/3)$-PAC learner that runs in time $T(N)$, with T nondecreasing, then satisfiability of CNF formulas of length L can be decided with one-sided error $1/3$ in time $T(c_q L(\log L)^\beta) \cdot \mathrm{polylog}(L) + \mathrm{poly}(L)$. Consequently, under $\mathrm{ETH}_C^{\mathrm{len}}$, no proper realizable $(\varepsilon_0, 1/3)$-PAC learner for LD_3, and so no proper $(\varepsilon_0, 1/3)$-agnostic learner, runs in time $2^{N/(\log N)^{C+\beta+1}}$.*

Proof of Theorem 4.9. ⟨1⟩1. LET: the algorithm compute S_F^{pcp} by Lemma 4.8 and run the learner on $\mathcal{D}_F$ for at most $T(N)$ steps, simulating each example draw in time $\mathrm{polylog}(L)$. If the learner does not halt in time, the algorithm rejects. Otherwise it computes the exact error of the output σ on S_F^{pcp} and accepts if and only if the error is below $2\varepsilon_0$.

⟨1⟩2. If F is satisfiable, the algorithm accepts with probability at least $2/3$.

PROOF: By Lemma 4.8, $\mathcal{D}_F$ is realizable. With probability at least $2/3$ the learner halts in time with an output of error at most $\varepsilon_0 < 2\varepsilon_0$.

⟨1⟩3. If F is unsatisfiable, the algorithm rejects.

PROOF: By Lemma 4.8, every hypothesis has error at least $2\varepsilon_0$.

⟨1⟩4. The algorithm runs in time $T(c_q L(\log L)^\beta) \cdot \mathrm{polylog}(L) + \mathrm{poly}(L)$.

PROOF: By Lemma 4.8, $N \le c_q L(\log L)^\beta$, and T is nondecreasing. A uniform entry of a list of polynomial length takes $\mathrm{polylog}(L)$ time to draw.

⟨1⟩5. Under $\mathrm{ETH}^{\mathrm{len}}_C$, no realizable learner runs in time $T(N) = 2^{N/(\log N)^{C+\beta+1}}$.

PROOF: Let $N_L = c_q L(\log L)^\beta$. Because $\log N_L \ge \log L$, we get $T(N_L) \le 2^{c_q L/(\log L)^{C+1}}$, and $T(N_L)\mathrm{polylog}(L) + \mathrm{poly}(L) < 2^{L/(\log L)^C}$ for all large L. The function T is nondecreasing for large N. By ⟨1⟩2–⟨1⟩4, such a learner would contradict $\mathrm{ETH}^{\mathrm{len}}_C$.

⟨1⟩6. Q.E.D.

PROOF: By ⟨1⟩2–⟨1⟩5. An agnostic learner is in particular a realizable one.

□

Corollary 4.10 (Constant accuracy is SAT by length)**.** *Fix a constant $\kappa \ge 3$, and let T and T' be nondecreasing.*

(a) If satisfiability of CNF formulas of length L can be decided in time $T(L)$ with error at most $1/3$, then every κ-local class is properly (ε, δ)-PAC learnable in time $O(N \log(N/\delta)) \cdot T(\kappa 2^\kappa m + N) + \mathrm{poly}(N, m)$, where $m = \lceil (N \ln 2 + \ln(2/\delta))/\varepsilon \rceil$.

(b) Conversely, a proper realizable $(\varepsilon_0, 1/3)$-PAC learner for LD_3 that runs in time $T'(N)$ decides satisfiability of CNF formulas of length L in time $T'(c_q L(\log L)^\beta) \cdot \mathrm{polylog}(L) + \mathrm{poly}(L)$.

With the new algorithms, (a) gives time $2^{O_{\kappa,\varepsilon,\delta}(N/\sqrt{\log N})}$. Under $\mathrm{ETH}^{\mathrm{len}}_C$, (b) gives the lower bound $2^{N/(\log N)^{C+\beta+1}}$ for every $\varepsilon \le \varepsilon_0$. A SAT algorithm with exponent $L/f(L)$ yields learners with exponent $O(N/f(\Theta(N)))$, and a learner with exponent $N/f(N)$ yields SAT algorithms with exponent $O(L(\log L)^\beta / f(\Theta(L(\log L)^\beta)))$.

Proof of Corollary 4.10. ⟨1⟩1. Part (a) holds.

⟨2⟩1. LET: S be a sample of m examples. With probability at least $1 - \delta/2$, every $w \in \mathrm{VS}(S)$ has error at most ε.

PROOF: Imported Theorem 2.1(a) with $\delta/2$.

⟨2⟩2. The version space is $\mathsf{SAT}(\Gamma_S)$ for a CNF Γ_S of length at most $\kappa 2^\kappa m$.

PROOF: Lemma 3.1.

⟨2⟩3. A self-reduction finds an element of $\mathsf{SAT}(\Gamma_S)$ with N satisfiability decisions on CNFs of length at most $\kappa 2^\kappa m + N$, all correct with probability at least $1 - \delta/2$.

PROOF: Fix $w_1, \dots, w_N$ in order: set $w_i = 0$ if Γ_S with the fixed values and $w_i = 0$ as unit clauses is satisfiable, and $w_i = 1$ otherwise. In the realizable case $\mathsf{SAT}(\Gamma_S) \ne \emptyset$ with probability 1. Repeat each decision $O(\log(N/\delta))$ times and take the majority, so that each is wrong with probability at most $\delta/(2N)$.

⟨2⟩4. Q.E.D.

PROOF: By ⟨2⟩1–⟨2⟩3 and a union bound, the output is ε-accurate with probability at least $1 - \delta$, within the stated time.

⟨1⟩2. Part (b) holds.

PROOF: Theorem 4.9.

⟨1⟩3. The stated consequences hold.

PROOF: With the new algorithms, $T(L) = 2^{3C_1 L/\sqrt{\log L}}$ (Section 2.5), and $\kappa 2^\kappa m + N = O_{\kappa,\varepsilon,\delta}(N)$. The lower bound is ⟨1⟩2 with Theorem 4.9. The transfer of exponents follows from (a) with $L = \Theta(N)$ and from (b) with $N = \Theta(L(\log L)^\beta)$.

⟨1⟩4. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩3.

□

Corollary 4.11 (Gap-3-SAT under $\mathrm{ETH}^{\mathrm{len}}_C$)**.** *Assume* $\mathrm{ETH}^{\mathrm{len}}_C$*. No randomized algorithm with error probability at most* $1/3$ *runs in time* $2^{n/(\log n)^{C+\beta+1}}$ *and distinguishes satisfiable* 3*-CNF clause multisets from those in which every assignment violates at least a* $2\varepsilon_0$ *fraction of the clauses. Here* n *is the number of variables plus the number of clauses, and no bound on the number of occurrences of a variable is assumed.*

Proof of Corollary 4.11. ⟨1⟩1. LET: F be a CNF of length L, and let S^{pcp}_F be the list of Lemma 4.8, read as a multiset of clauses of width at most 3. Its size is $n \le 2c_qL(\log L)^\beta$, it is satisfiable if F is, and otherwise every assignment violates at least a $2\varepsilon_0$ fraction of its clauses.

PROOF: By Lemma 4.8, with step ⟨1⟩4 of its proof for the size.

⟨1⟩2. An algorithm as excluded in the statement decides F in time $2^{2c_qL/(\log L)^{C+1}} + \mathrm{poly}(L) < 2^{L/(\log L)^C}$ for all large L.

PROOF: Let $n_L = 2c_qL(\log L)^\beta \ge n$. The function $x \mapsto x/(\log x)^{C+\beta+1}$ is increasing for large x, and $\log n_L \ge \log L$, so $n/(\log n)^{C+\beta+1} \le n_L/(\log n_L)^{C+\beta+1} \le 2c_qL/(\log L)^{C+1}$ for large n; for bounded n the time is $O(1)$.

⟨1⟩3. Q.E.D.

PROOF: The bound of ⟨1⟩2 contradicts $\mathrm{ETH}^{\mathrm{len}}_C$.

□

D.4 Hardness at inverse-polynomial accuracy

Theorem 4.12 (Hardness at inverse-polynomial accuracy)**.** *Assume Hypothesis 2.12. For every* $\gamma > 0$*, no randomized proper* $(\varepsilon, 1/3)$*-agnostic learner for* LD_3 *in dimension* n *with* $\varepsilon = 1/(8n^3)$ *runs in time* $O(2^{(1-\gamma)n})$*.*

Proof of Theorem 4.12. ⟨1⟩1. LET: F be a 3-CNF on n variables, and LET: F' be F without its t_0 tautological clauses. Then F' has $M \le 2n + 2n^2 + \frac{4}{3}n^3 < 8n^3$ clauses, and a maximum assignment for F' satisfies the maximum number of clauses of F.

PROOF: Tautological clauses are always satisfied. There are at most $\binom{n}{j}2^j$ non-tautological clauses of width j.

⟨1⟩2. LET: $\mathcal{D}$ be uniform on $\{(x_C, 1) : C \in F'\}$, where x_C has entry $+1$ at positive and -1 at negative literals of C. Then $\mathrm{err}_{\mathcal{D}}(\sigma)$ is the fraction of clauses of F' that σ violates.

PROOF: By Example 2.3(a), $h_\sigma(x_C) = 1$ if and only if σ satisfies C.

⟨1⟩3. Every output with $\mathrm{err}_{\mathcal{D}}(\sigma) \le \mathrm{OPT} + \varepsilon$ is an optimal assignment for F'.

PROOF: All errors are multiples of $1/M$, and $1/M > 1/(8n^3) = \varepsilon$.

⟨1⟩4. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩3, a learner in time $O(2^{(1-\gamma)n})$, with each example draw simulated in time $\mathrm{poly}(n)$, computes in time $O(2^{(1-\gamma)n}\mathrm{poly}(n)) \subseteq O(2^{(1-\gamma/2)n})$ the maximum number of satisfied clauses of F, namely t_0 plus the number of clauses of F' that its output satisfies, with error probability at most $1/3$. This contradicts Hypothesis 2.12 for $\gamma/2$. If F' is empty, the answer is t_0.

□

E Proofs for Section 5: the treewidth transition

Theorem 5.1 (Inference)**.** *Let Φ be a factor graph on n Boolean variables with nonempty scopes, arity at most a, size $\|\Phi\|$, and $|\mathcal{F}|$ factors.*

(a) If all tables have nonnegative integer entries below 2^b, then $Z(\Phi)$ and, if $Z(\Phi) > 0$, the marginal $\Pr[x_i = 1]$ for every i can be computed exactly in time $\mathrm{poly}(|\Phi|) \cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\|\Phi\|,\ n + b|\mathcal{F}| + 1,\ a^2\}$.

(b) If the factors carry integer energies in $\{0, \dots, b'\}$, then the minimum total energy, a minimizer, and the number of minimizers can be computed in time $\mathrm{poly}(|\Phi|) \cdot 2^{O(s/\sqrt{\log s})}$, where $s = \max\{4\|\Phi\|,\ n + 1,\ b'|\mathcal{F}| + 1,\ a^2\}$.

For constant degree, arity, b and b', all of these take time $2^{O(n/\sqrt{\log n})}$.

Proof of Theorem 5.1. ⟨1⟩1. Part (a) holds for $Z(\Phi)$.

PROOF: Each term of $Z(\Phi)$ is a product of $|\mathcal{F}|$ entries below 2^b, so $0 \le Z(\Phi) < 2^{n+b|\mathcal{F}|}$. Apply Theorem 2.7 over the counting ring $\mathcal{B}_{P,1}$ with $P = n + b|\mathcal{F}| + 1$. The hypotheses hold with the given s, because $a \le s^{1/2} \le s^{9/10}$.

⟨1⟩2. Part (a) holds for the marginals.

PROOF: $\Pr[x_i = 1] = Z(\Phi \mid x_i = 1)/Z(\Phi)$, where $\Phi \mid x_i = 1$ restricts the tables that contain x_i. A factor whose scope becomes empty is a constant (Theorem 2.7). The size and P do not grow.

⟨1⟩3. Part (b) holds.

PROOF: Use the profile ring with $P = n + 1$ and $\ell = b'|\mathcal{F}| + 1$, and give the factor f with energy E_f the table V^{E_f}. The partition function is $\sum_j c_j V^j$, where c_j counts the assignments of total energy $j \le b'|\mathcal{F}| < \ell$, and $c_j \le 2^n < 2^P$. The minimum energy is $\min\{j : c_j \ne 0\}$, and the number of minimizers is its coefficient. A minimizer follows by the self-reduction in the proof of Theorem 4.1, with $n + 1$ computations.

⟨1⟩4. For constant degree Δ, arity, b and b', the time is $2^{O(n/\sqrt{\log n})}$.

PROOF: Then $|\mathcal{F}| \le \|\Phi\| \le \Delta n$, so $s = O(n)$, and $s \ge n + 1$ gives $\log s \ge \log n$.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩4.

□

Corollary 5.2 (Separation)**.** *Let (G_n) be a family of n-vertex graphs of maximum degree $\Delta = O(1)$ with $\mathrm{tw}(G_n) = \Omega(n)$, for example bounded-degree expanders. Let Φ_n be the hardcore model on G_n: one variable per vertex and the factor $\mathbf{1}[\neg(x_u \wedge x_v)]$ on every edge, so that $Z(\Phi_n)$ counts independent sets.*

(a) The partition function $Z(\Phi_n)$ and all marginals can be computed in time $2^{O(n/\sqrt{\log n})}$.

(b) Variable elimination along any order, and junction-tree propagation over any tree decomposition, with explicitly stored tables, take time $2^{\Omega(n)}$.

(c) Every structured DNNF that represents the CNF $\bigwedge_{uv \in E(G_n)} (\neg x_u \vee \neg x_v)$ has size $2^{\Omega(n)}$ [ACMS20]. This includes every structured d-DNNF and every OBDD.

Proof of Corollary 5.2. ⟨1⟩1. Part (a) holds.

PROOF: The model has degree Δ, arity 2 and 0/1 tables. Apply Theorem 5.1.

⟨1⟩2. Part (b) holds.

PROOF: Every elimination order has induced width at least $\mathrm{tw}(G_n)$. So some elimination step multiplies factors whose scopes together contain at least $\mathrm{tw}(G_n) + 1$ variables, and tabulating

their product takes time $2^{\mathrm{tw}(G_n)}$. Likewise, every tree decomposition has a bag of size at least $\mathrm{tw}(G_n)+1$, and junction-tree propagation tabulates a potential over each bag. Finally $\mathrm{tw}(G_n)=\Omega(n)$.

⟨1⟩3. Part (c) holds.

PROOF: The substitution $x \mapsto \neg x$ maps the formula to the monotone 2-CNF $\bigwedge_{uv}(x_u \vee x_v)$ of degree Δ, whose primal graph is G_n. It does not change the size of any structured DNNF. Amarilli, Capelli, Monet and Senellart [ACMS20] prove that every structured DNNF for a monotone CNF of bounded arity and degree has size $2^{\Omega(\mathrm{tw})}$.

⟨1⟩4. Q.E.D.

PROOF: By ⟨1⟩1–⟨1⟩3.

□

Lemma 5.4 (OR-chains with copying)**.** *Given a CNF F on N variables with M clauses of total length L, one can compute in polynomial time a factor graph Φ_F with 0/1 tables, arity and maximum degree at most 3, and at most $N+3L$ variables, together with a path decomposition of width at most $N+2$, such that $Z(\Phi_F)=\#F$.*

Proof of Lemma 5.4. ⟨1⟩1. LET: Φ_F be built as follows. Each variable z_i gets an initial copy $z_i^{(0)}$. Process the literal occurrences in a fixed order, clause by clause. For the t-th literal of clause j, on the variable z_i, create a fresh copy z_i' and a chain variable $\zeta_{j,t}$. Add the factor $\mathbf{1}[z_i' = z_i^{\mathrm{cur}}]$, where z_i^{cur} is the most recent copy of z_i, and make z_i' the most recent copy. Add the factor $\mathbf{1}[\zeta_{j,t} = \zeta_{j,t-1} \vee \lambda(z_i')]$, where λ applies the sign of the literal. For each clause j with r_j literals, add a chain variable $\zeta_{j,0}$ with the unary factor $\mathbf{1}[\zeta_{j,0}=0]$, and the unary factor $\mathbf{1}[\zeta_{j,r_j}=1]$.

⟨1⟩2. $Z(\Phi_F)=\#F$.

PROOF: Fix $z \in \{0,1\}^N$. The equality factors force every copy of z_i to equal z_i. The chain factors force $\zeta_{j,t}$ to be the disjunction of the first t literals of clause j. So z has exactly one extension with all factors equal to 1 if z satisfies F, and none otherwise. Variables of F that occur in no clause contribute a factor 2 on both sides.

⟨1⟩3. The factor graph Φ_F has arity and degree at most 3 and at most $N+3L$ variables.

PROOF: A copy lies in at most two equality factors and one chain factor. A chain variable lies in at most two factors. Every factor has at most 3 variables. There are N initial copies, L further copies, and $L+M \le 2L$ chain variables.

⟨1⟩4. The bags $B_{j,t}$, one for each occurrence (j,t) in the processing order, form a path decomposition of width at most $N+2$. Here $B_{j,t}$ contains the most recent copies of all N variables before the occurrence, the new copy z_i', and $\zeta_{j,t-1}$ and $\zeta_{j,t}$.

PROOF: Each bag has at most $N+3$ elements. The equality and chain factors of (j,t) lie in $B_{j,t}$, and the unary factors of clause j lie in its first and last bags. A copy occurs from its creation until it is superseded, and $\zeta_{j,t}$ occurs only in $B_{j,t}$ and $B_{j,t+1}$, which are consecutive. So every variable occurs in a contiguous run of bags.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩2–⟨1⟩4. The construction takes polynomial time.

□

Corollary 5.3 (Inference is SAT by length)**.** *Assume $\mathrm{ETH}_C^{\mathrm{len}}$, and let $\theta > 0$. No randomized algorithm with error probability at most 1/3 decides whether the partition function of a 0/1 factor graph of arity and degree at most 3 is positive in time $2^{\mathrm{tw}/(\log \mathrm{tw})^{C+\theta}}\mathrm{poly}(n)$, or in time $2^{n/(\log n)^{C+\theta}}$. For $C<1$, no algorithm runs in time $2^{O(\mathrm{tw}/\log \mathrm{tw})}\mathrm{poly}(n)$.*

Proof of Corollary 5.3. ⟨1⟩1. LET: F be a CNF of length L in which every variable occurs, and LET: Φ_F be the factor graph of Lemma 5.4. Then Φ_F has arity and degree at most 3, at most $n \leq 4L$ variables, treewidth at most $L+2$, and $Z(\Phi_F) > 0$ if and only if F is satisfiable.

PROOF: Variables that occur in no clause do not affect satisfiability, so we may delete them. Then $N \leq L$, and Lemma 5.4 gives at most $N + 3L \leq 4L$ variables and a path decomposition of width at most $N + 2 \leq L + 2$. Also $Z(\Phi_F) = \#F$.

⟨1⟩2. An algorithm that decides positivity in time $2^{\mathrm{tw}/(\log \mathrm{tw})^{C+\theta}}\mathrm{poly}(n)$ decides F in time at most

$$2^{(L+2)/(\log(L+2))^{C+\theta}} \cdot \mathrm{poly}(L),$$

which is below $2^{L/(\log L)^C}$ for all large L.

PROOF: The function $x \mapsto x/(\log x)^{C+\theta}$ is increasing for large x, and $\mathrm{tw} \leq L+2$. The reduction takes polynomial time.

⟨1⟩3. An algorithm that runs in time $2^{n/(\log n)^{C+\theta}}$ decides F in time below $2^{L/(\log L)^C}$ for all large L.

PROOF: Here $n \leq 4L$, and $4L/(\log(4L))^{C+\theta} \leq 4L/(\log L)^{C+\theta}$, which is at most $L/(\log L)^C - \log^2 L$ for all large L.

⟨1⟩4. If $C < 1$, an algorithm in time $2^{O(\mathrm{tw}/\log \mathrm{tw})}\mathrm{poly}(n)$ contradicts $\mathrm{ETH}^{\mathrm{len}}_C$.

PROOF: Take $\theta = (1-C)/2$. For every constant K, we have $K\mathrm{tw}/\log \mathrm{tw} \leq \mathrm{tw}/(\log \mathrm{tw})^{C+\theta} + O(1)$, and ⟨1⟩2 applies.

⟨1⟩5. Q.E.D.

PROOF: By ⟨1⟩2–⟨1⟩4, each such algorithm contradicts $\mathrm{ETH}^{\mathrm{len}}_C$.

□

Theorem 5.5 (Inference lower bound)**.** *Assume Hypothesis 2.10. For every $\gamma > 0$ there is c such that no algorithm that is given a 0/1 factor graph of arity at most 3 and degree at most 4 with n variables, together with a path decomposition of width t, computes its partition function in time $O(2^{(1-\gamma)t})$, even when $n \leq 5t^{c+1}$ and the treewidth is at least $\sqrt{t}/2$. Under Hypothesis 2.9, the same holds for deciding whether the partition function is positive, and so for MAP inference.*

Proof of Theorem 5.5. ⟨1⟩1. LET: c be the constant that Hypothesis 2.10 gives for γ, and LET: F be a CNF on $N \geq 3$ variables with between 1 and N^c clauses, none of them tautological. Then F has length $L \leq N^{c+1}$.

PROOF: Deleting tautological clauses changes neither the count nor the hypothesis, and a formula without clauses is trivial. A non-tautological clause has at most N literals.

⟨1⟩2. LET: $r = \lceil\sqrt{N}\rceil$, and LET: Φ consist of Φ_F from Lemma 5.4 and a disjoint $r \times r$ grid of new variables with the all-ones factor on each grid edge. Then $Z(\Phi) = 2^{r^2}\#F$, and Φ has arity at most 3 and degree at most 4.

PROOF: All-ones factors leave the grid variables free. A grid vertex lies in at most 4 edge factors, and Φ_F has arity and degree at most 3.

⟨1⟩3. Φ has a path decomposition of width $t = N+2$, at most $5N^{c+1} \leq 5t^{c+1}$ variables, and treewidth at least $\sqrt{N} \geq \sqrt{t}/2$.

PROOF: The $r \times r$ grid has treewidth and pathwidth r. Every bag of the decomposition of Lemma 5.4 has exactly $N+3$ elements: the N current copies, the new copy and two chain variables. Placing a path decomposition of the grid of width r before it gives a path decomposition of width $\max\{r, N+2\} = N+2$. The number of variables is at most $N+3L+r^2 \leq 5N + 3N^{c+1} \leq 5N^{c+1}$, because $r^2 \leq 4N$ and $N^c \geq 3$. The treewidth is at least that of the grid, and $\sqrt{N} \geq \sqrt{N+2}/2$.

⟨1⟩4. An algorithm in time $O(2^{(1-\gamma)t})$ would count the satisfying assignments of F in time $O(2^{(1-\gamma)N})$.
PROOF: It computes $Z(\Phi)$ in time $O(2^{(1-\gamma)(N+2)}) = O(2^{(1-\gamma)N})$, and $\#F = Z(\Phi)/2^{r^2}$. The reduction takes polynomial time.

⟨1⟩5. Q.E.D.
PROOF: The algorithm of ⟨1⟩4 contradicts Hypothesis 2.10. For the decision version, $Z(\Phi) > 0$ if and only if F is satisfiable, and Hypothesis 2.9 applies in the same way. A MAP assignment of a 0/1 model has weight 1 if and only if $Z(\Phi) > 0$.

□

Corollary 5.6 (The treewidth transition)**.** *Consider Boolean models of bounded degree and arity, with small nonnegative integer weights.*

(a) If $n \le \mathrm{tw} \cdot (\log \mathrm{tw})^{1/2-\theta}$ for some $\theta > 0$, exact inference takes time $2^{o(\mathrm{tw})}$, and treewidth-based algorithms are not optimal.

(b) Under Hypothesis 2.10, for every $\gamma > 0$ no algorithm runs in time $2^{(1-\gamma)\mathrm{tw}}\mathrm{poly}(n)$ on all such models. This already holds for models with $n \le \mathrm{tw}^{O(1)}$ variables. So treewidth-based algorithms are optimal up to γ in the exponent once polynomially many variables are allowed.

Proof of Corollary 5.6. ⟨1⟩1. Part (a) holds.
PROOF: By Theorem 5.1, the time is $2^{O(n/\sqrt{\log n})}$. Because $n \ge \mathrm{tw}$, we have $n/\sqrt{\log n} \le \mathrm{tw}(\log \mathrm{tw})^{1/2-\theta}/\sqrt{\log \mathrm{tw}} = \mathrm{tw}(\log \mathrm{tw})^{-\theta} = o(\mathrm{tw})$. Treewidth-based algorithms take time $2^{\Omega(\mathrm{tw})}$ on the instances of Corollary 5.2, which satisfy $n = O(\mathrm{tw})$.

⟨1⟩2. Part (b) holds.
PROOF: Let c be as in Theorem 5.5 for $\gamma/2$. Its instances have degree at most 4 and arity at most 3, and they satisfy $\sqrt{t}/2 \le \mathrm{tw} \le t$ and $n \le 5t^{c+1} \le 5(4\mathrm{tw}^2)^{c+1}$, so $n \le \mathrm{tw}^{O(1)}$. An algorithm in time $2^{(1-\gamma)\mathrm{tw}}\mathrm{poly}(n)$ would run there in time $2^{(1-\gamma)t}\mathrm{poly}(t) = O(2^{(1-\gamma/2)t})$.

⟨1⟩3. Q.E.D.
PROOF: By ⟨1⟩1 and ⟨1⟩2.

□

F Proofs for Sections 6 and 8

Proposition 6.1 (Sparse CNF-SAT is subexponential)**.** *For all constants $c, \varepsilon' > 0$, the number of satisfying assignments of a CNF formula with n variables and at most cn clauses of arbitrary width can be computed in time $O(2^{\varepsilon' n})$.*

Proof of Proposition 6.1. ⟨1⟩1. LET: $k \ge 1$ be a constant with $\log(e(ck+1))/k \le \varepsilon'/2$, and LET: F be the formula without its tautological clauses. Run $\textsc{Reduce}_k(F)$ of Section C.3.
PROOF: The constant k exists because $\log(e(ck+1))/k \to 0$. Deleting tautological clauses changes neither the count nor the number of variables.

⟨1⟩2. The tree has at most $(n+1)(e(ck+1))^{n/k} \le (n+1)2^{\varepsilon' n/2}$ leaves.
PROOF: By Claim C.2 with $m \le cn$ clauses, the number of leaves is at most $\sum_{r \le n/k} \binom{m+r}{r} \le (n+1)\binom{m+\lfloor n/k \rfloor}{\lfloor n/k \rfloor} \le (n+1)\big(e(mk/n+1)\big)^{n/k}$, and $mk/n \le ck$.

⟨1⟩3. Each leaf formula can be counted over its free variables in time $2^{o(n)}$.
PROOF: It is a k-CNF on at most n variables. If at least 4 variables occur in it and it has at least 2 clauses, Imported Theorem 2.4 counts the occurring variables in time $2^{C_1 k^2 g(n)} = 2^{o(n)}$,

because k is constant. Otherwise brute force takes constant time. Each free variable that does not occur doubles the count.

⟨1⟩4. Q.E.D.

PROOF: By Claim C.1, the count of F is the sum of the leaf counts. The tree has at most twice as many nodes as leaves, with polynomial work per node. With ⟨1⟩2 and ⟨1⟩3, the total time is $(n+1)2^{\varepsilon' n/2} \cdot 2^{o(n)}\text{poly}(n) = O(2^{\varepsilon' n})$.

□

Proposition F.1 (Proper k-term DNF learning)**.** *Assume Hypothesis 2.9, and let $k \geq 2$ be a constant. For every $\gamma > 0$, no randomized algorithm decides in time $O(2^{(1/2-\gamma)n})$ whether a sample of polynomial size in $\{0,1\}^n$ is consistent with a k-term DNF. The same holds for proper PAC learning of k-term DNF to some inverse-polynomial accuracy.*

The proof encodes a CNF as a hypergraph on its literals and applies the reduction of Pitt and Valiant [PV88]. The factor $1/2$ comes from encoding literals as separate elements. Whether proper k-term DNF learning needs time $2^{(1-\gamma)n}$ under CNF-SETH, or admits a $2^{(1-\Omega(1))n}$ algorithm, is open.

Proof of Proposition F.1. ⟨1⟩1. LET: F be a CNF on N variables. LET: $U = \{z_i, \bar{z}_i : i \leq N\} \cup \{\bot\} \cup \{s_1, \ldots, s_{k-2}\}$, of size $2N + k - 1$, and LET: $\mathcal{E}$ be the hypergraph on U with the following edges: $\{z_i, \bar{z}_i\}$ for all i; $C \cup \{\bot\}$ for every clause C, where the literal $\neg z_i$ is the element $\bar{z}_i$; $\{s_a, s_b\}$ for all $a \neq b$; and $\{s_a, u\}$ for all a and all $u \notin \{s_1, \ldots, s_{k-2}\}$. Then F is satisfiable if and only if $\mathcal{E}$ has a k-coloring in which no edge is monochromatic.

⟨2⟩1. In such a coloring, the elements s_a get $k-2$ distinct colors that no other element uses.

PROOF: By the edges $\{s_a, s_b\}$ and $\{s_a, u\}$.

⟨2⟩2. If $\mathcal{E}$ has such a coloring χ, then F is satisfiable.

PROOF: By ⟨2⟩1, the other elements use two colors. Call them 0 and 1, with $\chi(\bot) = 0$. The edge $\{z_i, \bar{z}_i\}$ gives z_i and $\bar{z}_i$ opposite colors, so under the assignment $z_i = \chi(z_i)$ the color of each literal is its truth value. The edge $C \cup \{\bot\}$ is not monochromatic, so C contains a literal of color 1, which is true.

⟨2⟩3. If F is satisfiable, then $\mathcal{E}$ has such a coloring.

PROOF: Color each literal by its truth value under a satisfying assignment, $\bot$ by 0, and s_a by $a+1$. Each edge $C \cup \{\bot\}$ contains a true literal of color 1 and $\bot$ of color 0. The other edges are bichromatic by construction.

⟨2⟩4. Q.E.D.

PROOF: By ⟨2⟩2 and ⟨2⟩3.

⟨1⟩2. LET: U be identified with $[n]$, $n = 2N + k - 1$. LET: the sample consist of the positive examples $p_u = \mathbf{1} - e_u$ for all u and the negative examples $q_Y = \mathbf{1} - \mathbf{1}_Y$ for all edges Y of $\mathcal{E}$. Then $\mathcal{E}$ has a k-coloring without monochromatic edges if and only if some k-term DNF is consistent with the sample [PV88].

⟨2⟩1. If χ is such a coloring, then $T_1 \vee \cdots \vee T_k$ with $T_a = \bigwedge_{u:\chi(u)\neq a} x_u$ is consistent.

PROOF: The term $T_{\chi(u)}$ does not contain x_u, so $T_{\chi(u)}(p_u) = 1$. For every a, an edge Y contains some u with $\chi(u) \neq a$, so x_u is in T_a and $T_a(q_Y) = 0$.

⟨2⟩2. If $T_1 \vee \cdots \vee T_k$ is consistent, then coloring each u by some a with $T_a(p_u) = 1$ leaves no edge monochromatic.

PROOF: Suppose that an edge Y is monochromatic with color a. Every edge has at least two elements, so T_a accepts p_u and $p_{u'}$ for two distinct $u, u' \in Y$. A negative literal $\neg x_v$ is

true on p_u only if $v = u$, so T_a has no negative literal. A literal x_v with $v \in Y$ is false on p_v, which T_a accepts, so T_a contains no such literal. Thus $T_a(q_Y) = 1$, which contradicts consistency.

⟨2⟩3. Q.E.D.

PROOF: By ⟨2⟩1 and ⟨2⟩2.

⟨1⟩3. Q.E.D.

PROOF: The sample has polynomial size. An algorithm in time $O(2^{(1/2-\gamma)n})$ would decide F in time $O(2^{(1/2-\gamma)(2N+k-1)}) = O(2^{(1-2\gamma)N})$, which contradicts Hypothesis 2.9 with slack 2γ. For learners, use the argument of Theorem 3.4(iii) with the uniform distribution on the sample.

□

Proposition 8.1 (ERM for halfspaces on difference vectors)**.** *Let $G = ([n], A)$ be a digraph with nonnegative arc weights* wt*. Consider the sample $S_G = \{(e_i - e_j, 1) : (i, j) \in A\}$, in which the example of the arc (i, j) has weight* $\mathrm{wt}(i, j)$*, and homogeneous halfspaces $h_w(x) = \mathbf{1}[\langle w, x\rangle > 0]$ with $w \in \mathbb{R}^n$. Then the minimum over w of the weighted error $\sum_{(i,j)\in A} \mathrm{wt}(i, j)\, \mathbf{1}[h_w(e_i - e_j) = 0]$ is the minimum weight of a feedback arc set of G.*

Proof of Proposition 8.1. ⟨1⟩1. For every w, the misclassified arcs form a feedback arc set.

PROOF: The arc (i, j) is classified correctly if and only if $w_i > w_j$. A cycle of correctly classified arcs would give $w_{i_1} > w_{i_2} > \cdots > w_{i_1}$.

⟨1⟩2. For every feedback arc set B, some w misclassifies only arcs of B.

PROOF: The graph $([n], A \setminus B)$ is acyclic. Let $\mathrm{pos}(i)$ be the position of i in a topological order of it, and put $w_i = n - \mathrm{pos}(i)$. Every arc of $A \setminus B$ goes forward in this order, so it is classified correctly.

⟨1⟩3. Q.E.D.

PROOF: By ⟨1⟩1 and ⟨1⟩2, the minimum weighted error equals the minimum weight of a feedback arc set.

□

G Notation

The table lists the global symbols. Three of them have two closely related meanings, which the context always fixes: n, k and c. Symbols that occur only inside one proof are defined there.

Symbol	Meaning	Defined in
N	description length: number of description bits	Definition 2.2
n	number of variables of a CNF or factor graph; dimension of the data	Section 2.1
m, M	sample size; number of clauses or constraints	Section 2.1
k, κ	clause width or data sparsity; locality	Definition 2.2
$R(x)$, f_x, $R(S)$	bits read by example x; its predicate; $\bigcup_i R(x_i)$	Definition 2.2
w, σ, h_w	description; sign vector; hypothesis	Section 2.2
S, $\mathrm{VS}(S)$	sample; version space	Section 2.1
err_S, $\mathrm{err}_{\mathcal{D}}$, OPT	empirical error; true error; best true error	Section 2.1
$E_S(V)$, c_j, $\mathrm{mis}(u)$	error profile; number of descriptions with j errors; errors of a restriction u	Section 2.1
d	VC dimension	Imported Theorem 2.1
ε, δ	accuracy; confidence	Section 2.1
$\varepsilon_{\mathrm{dp}}$, η	privacy parameter; statistical distance	Theorem 4.6
m_{Occ}, m_{VC}, m_{UC}	sample sizes; constants C_0, C_B	Imported Theorem 2.1
C_1	constant of Imported Theorem 2.4	Imported Theorem 2.4
g, $T_\kappa(N)$	rate function; time bound of the realizable transfer	Section 3
Γ_S	CNF whose solutions are the version space	Lemma 3.1
$\mathcal{B}_{P,\ell}$, V	ring $(\mathbb{Z}/2^P\mathbb{Z})[V]/(V^\ell)$; its variable	Section 2.3
Φ, $\mathcal{F}$, I_f, ψ_f	factor graph; factors; scope; table	Section 2.4
$Z(\Phi)$, $\|\Phi\|$, $\lvert\Phi\rvert$, a	partition function; size; bit length; arity	Section 2.4
G, ∂v, ψ_v	tensor network; edges at v; tensor at v	Section 2.3
s	size parameter of the engine	Theorem 2.7
ρ	rate of an algorithm or hypothesis of the form $2^{\rho n}$	Lemma 3.5
γ, θ	slack of a hypothesis, as in $2^{(1-\gamma)n}$; slack of an exponent	Section 2.5
μ, ν	exponents of k and of $\log n$ in a k-SAT algorithm	Corollary 3.7
c	clause exponent (n^c clauses) or clause density (cn clauses), as stated	Hypothesis 2.9
K_0, K', K	branching width and hypothesis width of the width reduction; sparsity constant of the barrier	Lemma 3.5 and Theorem 3.8
L, C	length of a CNF; rate of $\mathrm{ETH}^{\mathrm{len}}_C$	Hypothesis 2.14
q, β, c_V, c_q	PCP queries; polylog exponent; size constant; $c_q = 2q2^q c_V$	Imported Theorem 4.7
Ω_F, τ, Q_τ, D_τ	seeds; a seed; its query set; its predicate	Imported Theorem 4.7
ε_0	accuracy of the PCP reduction, $1/(q2^{q+2})$	Lemma 4.8
tw, t	treewidth; width of a given path decomposition	Section 5
LD_k, BP_k	literal disjunctions; binarized perceptrons	Example 2.3