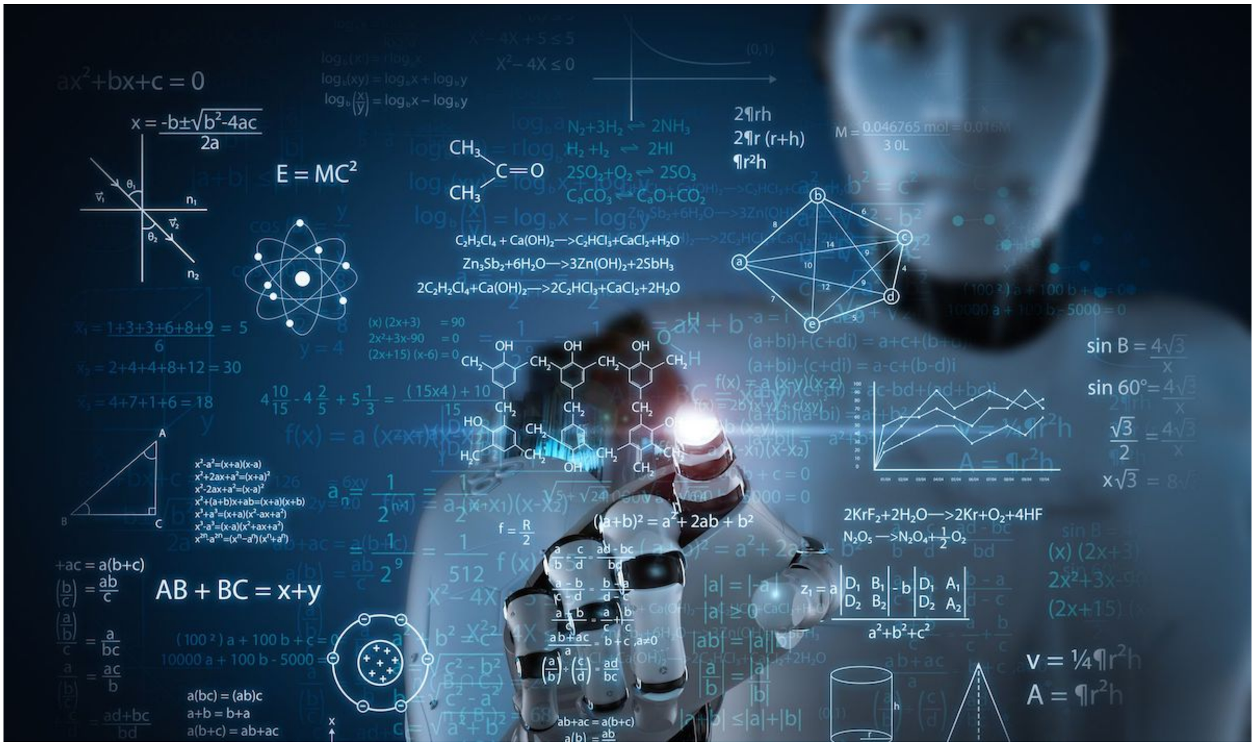
Evaluation of Theoretical Learnability and Hardness of Neural Networks



$ax^2+bx+c=0$
$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$
$E=MC^2$
$2¶rh$
$2¶r(r+h)$
$¶r^2h$
$C_2H_2Cl_4+Ca(OH)_2\rightarrow C_2HCl_3+CaCl_2+H_2O$
$Zn_3Sb_2+6H_2O\rightarrow 3Zn(OH)_2+2SbH_3$
$2C_2H_2Cl_4+Ca(OH)_2\rightarrow 2C_2HCl_3+CaCl_2+2H_2O$
$sin B = \frac{4\sqrt{3}}{x}$
$sin 60° = \frac{4\sqrt{3}}{x}$
$(a+b)^2=a^2+2ab+b^2$
$2KrF_2+2H_2O\rightarrow 2Kr+O_2+4HF$
$AB+BC=x+y$
$a^2+b^2+c^2$
$V=¼¶r^2h$
$A=¶r^2h$
$a(bc)=(ab)c$
$a+b=b+a$
$a(b+c)=ab+ac$

Image source

Table of Contents:

1. Introduction

The following paper is a synopsis of four research papers relevant to the topic of hardness in learning neural networks (NNs), in chronological order.

1.1. Summary of 'On the Complexity of Learning Neural Networks' [1]

This paper uses a generalized version of the SQ model to prove the hardness of learning a function generated by a NN with one hidden layer, any known activation function, and any log-concave input distribution. It states that there exists (and gives) a family of functions which take the NN exponential updates (queries) to learn for any learning algorithm. In the worst case, the target function is one such function, and hence the lower bounds are exponential. This paper is from 2017, the other three are from 2020.

1.2 Summary of 'Hardness of Learning Neural Networks with Natural Weights' [2]

Previous work has already shown hardness for arbitrary NN weights, leading to the idea that learnable NNs have weights from "natural" distributions (obeying some general property) instead of arbitrary weights. This paper's finding is that most "natural" weight distributions still lead to hardness of learning, and if distribution properties exist for efficient learning, then they are more rare than natural. The weight distributions studied include: the multivariate normal distribution, the uniform distribution on a sphere, and each weight vector component drawn i.i.d. from a normal, uniform, or Bernoulli distribution. The analysis is extended to Convolutional Neural Networks (CNNs).

1.3. Summary of 'Superpolynomial Lower Bounds for Learning One-Layer Neural Networks using Gradient Descent' [3]

The first 'Complexity' [1] paper is most like this research paper, because both prove lower bounds for learning in gradient descent (modeled by SQ). Unlike 'Complexity', this paper's lower bounds are not exponential, but only superpolynomial. Improving on 'Complexity' however, the lower bounds hold for any inverse-polynomial tolerance parameter, polynomially-large batch sizes, and with no Lipschitz constraint. Addressing the 'Hardness' [2] paper's results, this paper mentions that the hard distributions are not Gaussian like in this paper, and that the hardness results require a nonlinear clipping output activation.

1.4. Summary of ‘Algorithms and SQ Lower Bounds for PAC Learning One-Hidden-Layer ReLU Networks’ [4]

Published concurrently to the previous ‘Superpolynomial’ [3] paper, this paper shows hardness of learning one-hidden-layer NNs, by stating that there exists (and creating) a family of functions which require exponential samples/time to learn, while the previous paper only achieved superpolynomial lower bounds. However, for NNs with strictly positive coefficients, an efficient/polynomial learning algorithm is provided. Hence, the concept classes with arbitrary coefficients versus positive coefficients are separated in theoretical learnability.

2. Analysis of ‘On the Complexity of Learning Neural Networks’ [1]

2.1. Paper highlights

NNs can approximate any function, and usually learn by stochastic gradient descent. This paper shows the hardness of learning a function generated by a single hidden layer NN, with any known activation function, and any log-concave input distribution. From this hardness finding, it suggests that stronger assumptions are needed for learning tractability. It is shown theoretically and experimentally (section 4) that when $s\sqrt{n}$ (*n* being input dimension and *s* being the sharpness parameter of the activation function) goes over a threshold (roughly 5 in the experiment), the target function is hard to learn. In the later 'Superpolynomial' paper [3], its experiment also supports its theoretical findings. Finally, as a corollary (Corollary 5), the Lipschitz assumption is not necessary to the hardness results when the concept class’s function outputs have finite precision.

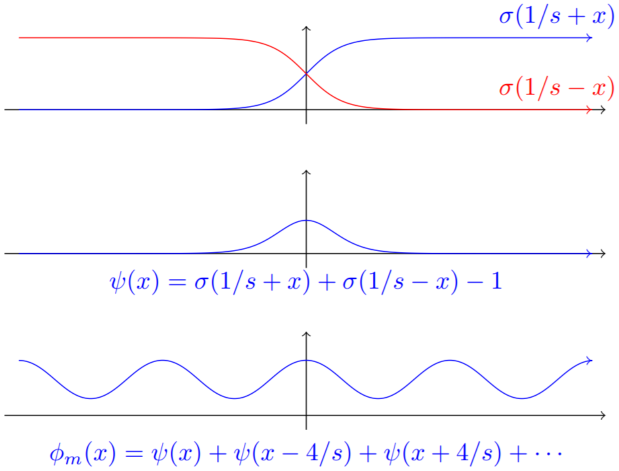
2.2. Proof overview

The lower bound proof uses a generalization of the SQ framework, VSTAT(*t*) instead of STAT(τ), by a 2013 paper [5] (the most referenced source in this paper), which gives a nearly optimal lower bound on the complexity of any SQ algorithm for detecting planted bipartite clique distributions. The hardness of this problem has been used to prove hardness of many other problems, such as in this paper.

The update of an NN can be modeled as a statistical query to the input distribution. Each (gradient) step is calculated by averaging a batch of random examples, so we can simulate this in the generalized SQ model with a VSTAT(*t*) call. The query function which we ask the oracle is the gradient of the loss function (which we assume is poly(*s*,*n*)-Lipschitz, i.e. does not change too rapidly relative to *s* or *n*).

Using NNs which approximate parity functions gives too weak bounds on tolerance, so we use the statistical dimension of a family of *s*-wave functions (precise statements in section 2.1), similar to the correlated concepts methodology for lower bounds. This paper makes a stronger *statistical dimension for average covariances* ϵ-SDA(C,D,$\bar{\gamma}$) for regression problems using *the average covariances of indicator functions* $\bar{\gamma}$, using ideas from the VSTAT(t) paper [5]. Then, Theorem 2.1 (based on Theorem 2 in another paper [6]) proves that, for any randomized algorithm using VSTAT(t), the lower bound for amount of queries comes from the ϵ-SDA(C,D,$\bar{\gamma}$) of the concept class C and distribution D. Finally, finding the ϵ-SDA(C,D,$\bar{\gamma}$) of the concept class of *s*-wave functions proves the paper's findings (Theorem 1.1). Now with this overall outline, we go into more detail explaining the proof.

First, Lemma 3.1 gives exponential (in $\Omega(n)$) bounds for ϵ-SDA(C,D^n,$\bar{\gamma}$) for quasiperiodic functions. Next, the paper shows that combining any typical activation functions results in quasiperiodic functions (Lemma 3.2). To combine typical activation functions σ into quasiperiodic functions ø$_m$, there is an intermediate step at ψ functions: functions with a bounded integral over all real numbers (see Figure 1.1a, reproduced here).



(a) The sigmoid function, the L^1-function ψ constructed from sigmoid functions, and the nearly-periodic "wave" function ϕ constructed from ψ.

On the intermediate functions ψ, it defines the *essential radius* and *mean bound property*. ψ is defined as an affine (parallel) mix of sigmoid gates in the above image, but ψ is always defined as some combination of activation gates so that it satisfies the properties of Lemma 3.2, which uses the essential radius and mean bound property to combine ψ functions into quasiperiodic ø$_m$ functions. The amount of ψ-units needed for quasiperiodic behavior in ø$_m$ is polynomial in the Lipschitz parameter *λ* and input dimension *n*.

3. Analysis of ‘Hardness of Learning Neural Networks with Natural Weights’ [2]

3.1. Paper highlights and relevance to other papers

Existing results already give hardness of learning for arbitrary weights, so the next logical question is if there is a natural property on weights which allows efficient learning. We define a *natural property* of a network’s weights if the property holds with respect to “natural” weight distributions. This paper’s findings say: If properties for efficient learning exist, they are not natural (they are rare with respect to typical weight distributions). The next steps would be addressing the clipping operation in the second layer used in this paper as well as finding (rare) properties for efficient learning.

Unlike other papers, the hardness is not based on $P \neq NP$complexity assumptions, but based on the assumption that refuting a RSAT (random K-SAT) formula is hard. The hardness results in this paper only hold for depth-2 networks but for all algorithms. According to related work from another 2020 paper [7], deep NNs, with depth ω(log(*n*)) and arbitrary activation units (that must be nonlinear and under certain normalization), are hard to learn for SQ algorithms.

3.2. Proof overview

Before diving into the proof overview, we must cover some preliminary definitions. Samples are considered *scattered* if the sample labels are fair coin flips (independent of the sample’s evaluation by a concept). If for all samples $x_i \in S$, the sample $x_i \in A$, then we say *S* is *contained* in *A*. Finally, an algorithm *distinguishes* if, with high probability, it correctly outputs a decision on the samples’ labels of either *scattered* or H-realizable (for some hypothesis class H).

Using this terminology, we define a series of problems based on distributions over matrices D_{mat} (for NNs) or distributions over vectors D_{vec} (for CNNs):

- $SCAT^{A}_{m(n)}(H)$: problem of distinguishing samples contained in *A* as H-realizable or scattered.
- $SCAT^{A}_{m(n)}(D_{mat})$: problem of distinguishing samples contained in *A* as D_{mat}-realizable or scattered.
- $SCAT^{A}_{m(n)}(D_{vec}, n)$: problem of distinguishing samples contained in *A* D_{vec}-realizable or scattered .

The paper’s proof takes the following logical steps. First, it reduces $SCAT(D_{mat})$ to learning D_{mat}-random NNs; i.e. if an algorithm can learn a D_{mat}-random NN, then the algorithm can solve $SCAT(D_{mat})$ (Theorem 5.1). Then, it shows $SCAT(H_{sign-cnn})$ is RSAT-hard (section 5.2). Next, it reduces $SCAT(H_{sign-cnn})$ to $SCAT(D_{mat})$, because both are decision problems. Since $SCAT(H_{sign-cnn})$ is RSAT-hard, we know by the reduction that $SCAT(D_{mat})$ is also RSAT-hard (Lemmas 5.5 and 5.6). Finally, the paper reaches the conclusion that learning D_{mat}-random NNs is RSAT-hard (Theorems 3.1-3.3). The entire proof is extended to learning CNNs with D_{vec} instead of D_{mat} (Theorems 3.4-3.6).

The proof explanation above is simplified to stay high-level. Among other simplifications, it glosses over the multiple weight distributions studied for which different properties, lemmas, and theorems hold. For example, Theorems 3.1-3.3 give the same hardness results on learning NNs, but for the symmetric subgaussian distribution, the multivariate normal distribution, and the uniform distribution over a sphere, respectively.

4. Analysis of 'Superpolynomial Lower Bounds for Learning One-Layer Neural Networks using Gradient Descent' [3]

4.1. Paper highlights and relevance to other papers

The most relevant source for this paper is the first 'Complexity' paper [1]. Similarly to the 'Complexity' paper, this paper gives lower bounds for learning with gradient descent with respect to square loss or logistic loss, except the 'Complexity' paper's bounds hold only for small batch sizes, sharp activations, and specific classes of queries. In contrast, this paper's lower bounds hold for any inverse-polynomial tolerance parameter, polynomially-large batch sizes, and with no Lipschitz constraint. The paper uses 1-Lipschitz ReLU or sigmoid activations (with zero bias), but the results hold for any activation with a nonzero Hermite coefficient of degree *k* or higher (for $2^k = m$ where *m* is hidden units). In the 'Complexity' paper, the lower bounds are exponential in the input dimension. The lower bounds in this paper are quasipolynomial in hidden units, which the following 'Algorithms' paper [4] supersedes with stronger exponential bounds. Both 'Complexity' and this paper have experimental evidence that supports the theoretical findings.

This paper also mentions the results from another 2020 paper [9], that stochastic gradient descent (SGD) is universal, i.e. it can encode all polynomial time learners, so unconditional lower bounds on SGD would prove $P \neq NP$. To use the SQ model for proving lower bounds, a batch of samples is required, as opposed to the one sample per epoch/update in SGD. Thus, hardness results on SGD are unlikely to be proven.

Even more elements of the 'Complexity' paper [1] are useful in this paper. The VSTAT(*t*) oracle used in 'Complexity' is mentioned in a footnote, but this paper uses the STAT(τ) oracle from the original SQ model. This paper also makes a *statistical dimension on average*, which is denoted $SDA_D(C,\gamma)$ for threshold γ. To prove SQ lower bounds on probabilistic concepts, it follows the analysis of the most referenced source

[5] from 'Complexity'.

For additional results, this paper rules out efficient learning for classification (labels restricted to $\{\pm\ 1\}$) with any loss function calculable in polynomial time. This analysis uses distinguishing problems from another paper [8], similarly to the distinguishing-problems analysis in the previous 'Hardness' paper [2] but with probabilistic concepts. Also, the hardness results from the 'Hardness' paper are for distributions which are not Gaussians like in this paper, and require a nonlinear clipping output activation.

4.2. Proof overview

It is impossible to rule out SQ algorithms learning real-valued functions using only orthogonal function families, but SQ reductions hold for real-valued functions if the algorithm only uses inner-product queries (and the norms of the functions are sufficiently large). Prior work establishes that any SQ learner which can only make inner-product queries needs $\Omega(d)$ queries of tolerance $\tau=\sqrt{\gamma}$ for $d=\mathrm{SDA}_D(C,\gamma)$ (Theorem 4.1 with proof in Appendix B). Gradient descent with square/logistic loss can be implemented using only inner-product queries, hence this paper's lower bounds.

The core of proving SQ lower bounds is finding pairwise approximately orthogonal functions with respect to the underlying marginal distribution. In this paper, the result relies on a construction of a simple family of NN functions that are exactly orthogonal with respect to all spherically symmetric distributions (including the spherical Gaussian this paper uses as its marginal distribution).

First, the paper defines the family of functions (section 3). Then, the paper proves that the functions are not exponentially close to the zero function, which it does by proving a norm lower bound (Theorem 3.8, proved in Appendix A). Next, the orthogonality of the family of functions is proved (Theorem 3.5). Lemma 2.6 gives lower bounds on $\mathrm{SDA}_D(C,\gamma)$ where pairwise correlation bounds are known, which is extrapolated to Corollary 3.7 that gives lower bounds on $\mathrm{SDA}_D(C,\gamma)$ for orthogonal functions.

Finally, the hardness results follow from careful choices of ϵ, γ, and τ. Pick γ such that $\mathrm{SDA}_D(C,\gamma)$ is still $n^{\Theta(k)}$ (for $2^k = m$ hidden units). Theorem 3.8's norm lower bound allows picking $\epsilon = e^{-\Theta(k)}$ and $\tau = = n^{-\Theta(k)}$. All the parameters obey the required properties for the proof of the main results in Theorem 3.9: the superpolynomial lower bound on queries for learning the orthogonal function family.

For the additional results on any general SQ algorithm (not just inner-product queries) learning classification, we introduce probabilistic concepts from the learning model in the paper [5] most referenced by the 'Complexity' paper [1]. The analysis follows said model: define a distinguishing problem, give a lower bound, then show learning is at least as hard as distinguishing, thereby getting a lower bound for learning. This paper defines probabilistic concepts and the distinguishing problem for a class of probabilistic concepts (Definition 4.2). Then, Theorem 4.5 proves the lower bound of $\mathrm{SDA}_D(C,\gamma)$ queries of tolerance $\tau=\sqrt{\gamma}$ on the distinguishing problem with C as a probabilistic concept class. Lemma 4.4 gives a reduction to show learning is as hard as distinguishing. The resulting lower bound on probabilistic concepts for any SQ learner is at least $\mathrm{SDA}_D(C,\gamma)-1$ queries (Corollary 4.6).

5. Analysis of 'Algorithms and SQ Lower Bounds for PAC Learning One-Hidden-Layer ReLU Networks' [4]

5.1. Paper highlights and relevance to other papers

There are two main results in this paper: a polynomial-time algorithm for learning NN with positive coefficients, for k hidden units up to $\tilde{O}(logd)$ (d being input dimension) and additive random label noise; an exponential lower bound for learning NN with arbitrary real coefficients for $k=\omega(1)$ using the SQ model. C_k^+ is the concept class of single hidden layer NNs with ReLU activations and positive weight coefficients, and C_k is the same concept class but with arbitrary real weight coefficients α. Therefore, this paper separates the classes $C_k^+$and C_k in terms of hardness.

This and 'Hardness' [2] are the only papers of the four which are explicitly working in the PAC learning model. Importantly, this paper is learning in the PAC model without assumptions on the weight matrix. In the 'Complexity' paper [1], the hardness results require an upper bound on the condition number of the weight matrix, because there exist SQ algorithms for learning certain families of NN functions in time complexity which is polynomial in the condition number. With a small condition number, learning is tractable. This paper gives an algorithm for C_k^+ with complexity independent of the condition number. Note that there is no solution for C_k, even with strong assumptions on the condition number. C_k is information-theoretically solvable with polynomially many samples, but the question remains open if a computationally efficient algorithm exists. Even the concept class C_k^+, for which this paper gives an efficient algorithm under the Gaussian distribution, was shown NP-hard for arbitrary distributions, even for $k=2$, in the related work from another paper [10].

The 'Superpolynomial' paper [3] studied the concept class C_k concurrently (published on June 22 as opposed to this paper on June 23), and achieved superpolynomial lower bounds. The 'Complexity' [1], 'Superpolynomial', and this paper all prove their lower bound using the correlational/inner-product SQ framework [5], since the model applies to many algorithms including first-order methods (e.g. gradient descent), dimension-reduction, and moment-based methods.

5.2. Proof overview

Before overviewing the proof, we will first explain the results in greater detail.

Theorem 1.3 is the positive result of the existence of an algorithm for C_k^+ which draws $poly(k/\epsilon)*\tilde{O}(d)$ noisy examples and runs in time $poly(d/\epsilon)+(k/\epsilon)^{O(k^2)}$ to get ϵ-close to the true function in L_2-norm. This theorem is the specific case of ReLU activations (with parameters $L=1$, $C=1/\sqrt{2\pi}$) in the following more general theorem: for any non-negative L-Lipschitz activation functions (with lower bounds of C on the expected values of the activation functions) and noisy samples, there exists an algorithm which draws $d*poly(k,1/\epsilon)*poly(L/C)$ samples and runs in time $poly(m)+\tilde{O}((1/\epsilon)^{k^2})$ (Theorem 3.1). The theorem also extends to the agnostic case, in which the data and labels do not necessarily come from a function in C_k^+(the algorithm is proper, though). The running time of the algorithm scales with e^k, and the existence of an algorithm with running time scaling polynomially in k is still unknown. The given algorithm is also robust to small (dimension-independent) adversarial L_2-error. Now, we outline the proof.

First, the fact that the target function only depends on the k-dimensional subspace spanned by the weights allows us to do essentially a dimension reduction from d to k (Lemma 3.6). By approximating the reduced-dimension subspace and brute forcing the k^2-dimensional space of functions (by a covering method), the algorithm gets a hypothesis function approximating the true function. To approximate the subspace V', we use the degree-2 chow parameters of the true function f. The degree-2 Chow parameters matrix $\boldsymbol{A}$ can be estimated with $\tilde{O}(dk/\epsilon^2)$ samples, since the activation function is well-behaved and the marginal distribution is Gaussian. The estimation of degree-2 Chow parameters is in Lemma 3.4, and the proof relies on matrix concentration and concentration of polynomials of Gaussian random variables (Appendix B Lemma B.1). If V' is the span of the k largest eigenvalues of a second moment matrix $\boldsymbol{A}$, the weight vectors corresponding to the important components of f will still be close to V'. All these steps are polynomial, and approximate f with high probability. Finally, the complete algorithm for learning the NN is given (Algorithm 1).

For the negative result with arbitrary coefficients in C_k, the paper rules out any algorithms with complexity polynomial in d for $\epsilon = \Omega(1)$ and any $k=\omega(1)$. To prove this result, it first defines its correlational SQ model: an oracle takes query q: $\mathbb{R}^d \rightarrow [-1,1]$ and returns an estimate γ such that $|\gamma\text{-E}_{x\sim D}[f(x)q(x)]| \leq \tau$ tolerance (Definition 4.1). For this paper's correlational SQ model, the concept $f(x)$ is not bounded pointwise but only in the L_2-norm: $E_{x\sim D}[f^2(x)] \leq 1$. Theorem 1.4 states the negative result: any correlational SQ algorithm for C_k that guarantees $\epsilon = \Omega(1)$ requires either $2^{d^{\Omega(1)}}$ queries or accuracy $d^{-\Omega(k)}$. If using a query accuracy of $d^{-\Omega(k)}$, simulating one query would require $d^{\Omega(k)}$ samples and hence time. Instead, if using $2^{d^{\Omega(1)}}$ queries, since each query is at least one time unit, the algorithm would take $2^{d^{\Omega(1)}}$ time. Either way, there are exponential lower bounds for running time. Theorem 4.3 is the negative result of Theorem 1.4 generalized to a broader family of activations. Once again, we continue to the proof outline.

The paper creates a family of functions which are pairwise far from each other and have small pairwise correlations, which can be done by having all the functions have low-degree moments going to 0. Having low-degree moments go to 0 is done by defining a function in two dimensions with the correct moments and embedding it in a randomly chosen subspace. The paper constructs such functions as a 2D mixture of *2k* ReLUs whose first *k-1* moments vanish. Then, the paper proves that randomly chosen subspaces have small correlation and are likely far apart. Using the family of functions embedded into the random subspaces, the paper can go into SQ lower bounds. First, the correlational SQ dimension SDA(C,D,ϱ) is defined (Definition 4.4). Then, Lemma 4.5 gives lower bounds on the number of queries for any correlational SQ algorithm to at least SDA(C,D,τ) queries of tolerance τ. The SDA($F^{W}_{\sigma,ø}$,D,τ') of the constructed function family is calculated in the proof for Theorem 4.3, which states the lower bound hardness result.

6. Conclusion

- Certain configurations of neural network learning problems have been shown for theoretical learnability and for hardness.
- The statistical query model, its generalized variant, and the statistical dimension are important tools for the theoretical analysis of neural networks ([1], [3], [4]).
- The empirical success of neural networks is exciting, and theoretical explanations are soon to follow.

References

[1] Song, L., Vempala, S., Wilmes, J., and Xie, B., "On the Complexity of Learning Neural Networks", *arXiv e-prints*, 2017.

[2] Daniely, A. and Vardi, G., "Hardness of Learning Neural Networks with Natural Weights", *arXiv e-prints*, 2020.

[3] Goel, S., Gollakota, A., Jin, Z., Karmalkar, S., and Klivans, A., "Superpolynomial Lower Bounds for Learning One-Layer Neural Networks using Gradient Descent", *arXiv e-prints*, 2020.

[4] Diakonikolas, I., Kane, D. M., Kontonis, V., and Zarifis, N., "Algorithms and SQ Lower Bounds for PAC Learning One-Hidden-Layer ReLU Networks", *arXiv e-prints*, 2020.

[5] Feldman, V., Grigorescu, E., Reyzin, L., Vempala, S., and Xiao, Y., "Statistical Algorithms and a Lower Bound for Detecting Planted Clique", *arXiv e-prints*, 2012.

[6] Szörényi, B. "Characterizing Statistical Query Learning: Simplified Notions and Proofs," in *Algorithmic Learning Theory*, Berlin, Heidelberg, 2009.

[7] Agarwal, N., Awasthi, P., and Kale, S., "A Deep Conditioning Treatment of Neural Networks", *arXiv e-prints*, 2020.

[8] M. J. Kearns and R. E. Schapire, "Efficient distribution-free learning of probabilistic concepts," *Journal of Computer and System Sciences*, vol. 48, no. 3, Jun. 1994, doi: 10.1016/S0022-0000(05)80062-5.

[9] Abbe, E. and Sandon, C., "Poly-time universality and limitations of deep learning", *arXiv e-prints*, 2020.

[10] Goel, S., Klivans, A., Manurangsi, P., and Reichman, D., "Tight Hardness Results for Training Depth-2 ReLU Networks", *arXiv e-prints*, 2020.