Research Statement — Ido Nachum

I enjoy understanding and investigating nature across the scientific spectrum and sharing my enthusiasm with others through collaborative research and teaching. My career path is a representation of this affection. I started as an aerospace engineer, worked in RAFAEL Advanced Defense Systems for more than five years (as part of my military service) while completing math undergrad courses, and then worked on a thesis towards an MSc in pure mathematics [2]. During my PhD [3], I studied theoretical questions at the intersection between machine learning and information theory (Section 1). Furthermore, I am interested in mathematical questions arising from artificial or biological neural computation (Section 2). In the future, I intend to continue to walk this path between theory and practice while absorbing new and intriguing ideas from both theoreticians and practitioners (Section 3).

1 Past Research

Machine Learning through the Lens of Information Theory

Machine learning is quickly permeating many aspects of our lives. Examples are abundant: image and speech recognition, autonomous driving, medical diagnosis, spam and online fraud detection, and many more. Machine learning enables a computer to learn from experience, as a child does, instead of explicitly programming it to perform a specific task.

Information theory is already an inseparable part of our life. It allows efficient storage (compression) and communication of information. For example, ZIP files support lossless data compression, and wireless communication uses error-correcting and error-detecting codes. Information theory studies what are the fundamental limits of those tasks and how to achieve them.

Machine learning and information-theoretic tasks are in some sense equivalent since both involve identifying patterns and regularities in data. To recognize an elephant, a child (or a neural network) observes the repeating pattern of big ears, a trunk, and grey skin. To compress a book, a compression algorithm searches for highly repeating letters or words. So the high-level question that guided my PhD research is

When is learning equivalent to compression?

More precisely, how rigorous notions of learning are related to rigorous notions of compression? Variants of this question were studied extensively over the years. In many contexts, the ability to compress implies learnability. Here is a partial list of examples: sample compression schemes [29, 32], Occam's razor [15], minimum description length [33, 22], and differential privacy [19, 18, 12, 34, 13].

In the typical learning setting, an algorithm receives *input* comprising i.i.d. pairs $\{(x_1, f(x_1)), ..., (x_n, f(x_n))\}$ of training data of a function f, and returns *output*—a hypothesis that should fit the samples. In [4], I measured the learning algorithm's compression using information theory with the quantity $I(input; output)$, the mutual information between the training data and the output hypothesis of the learning algorithm, measured in bits. Roughly speaking, this quantity measures the number of bits the algorithm retains from the training data or how many bits of information are revealed by the algorithm (a measure of privacy). Here are my results under this formal setting.

Generalization bound. The generalization error is a key notion in machine learning. It is the difference between the empirical error (training data error) and the true error (unseen data error). Under this formal setting, in [4] my collaborators and I showed that *compression implies learning*, namely, the generalization error is bounded from above by $O(I(input; output))$. This result highlights a simple rule of thumb for designing learning algorithms: construct algorithms that have a small empirical error but at the same time reveal little information about its input.

Information complexity of learning. We showed that compression imply learning. It is therefore natural to ask whether the opposite holds true as well, namely, *does learning imply compression?* In [4, 5, 6, 11], my collaborators and I showed that the answer depends on the definitions of learning and compression. In particular— In [4] we answer this question in the negative for the *probably approximately correct* (*PAC*) learnable [37] class of threshold functions $\mathcal{H}_N = \{1_{x \geq k} \mid k \in [N]\} \subset \{0,1\}^{[N]}$. To that end, we show that the information complexity is $IC(\mathcal{H}) = \Theta(\log\log N)$, meaning that, for any learning algorithm, there exists a *worst-case* scenario that requires the algorithm to use $\Omega(\log\log N)$-bits of information. Since N is arbitrary (it does not affect how well the class can be learned), the worst-case information complexity is unbounded. Thus, learning does not imply compression in the worst case.

More generally, we show in [5] that, for any N and d, there exists a space with $|\mathcal{X}| = N$ and a class $\mathcal{H} \subset \{0,1\}^{\mathcal{X}}$ of VC dimension d with $IC(\mathcal{H}) = \Theta(d \log\log N)$.

In contrast, I show in [6] that for every learnable class of VC dimension d, there exists a learning algorithm that retains $O(d)$-bits of information in the *average case*. That is, in every scenario, most functions will not require revealing many bits to learn them. Learning implies compression in the average-case, or equivalently, the average-case information complexity for VC classes is finite.

Online learning and information complexity. The complexity measure for online learning is captured by the Littlestone dimension [28]. For example, the simple class of thresholds $\mathcal{H}_N$ has $\log N$ Littlestone dimension (unbounded), although its VC dimension is one. In [11], we showed that a binary hypotheses class has finite Littlestone dimension (combinatorial definition) if and only if it is learnable with finite information complexity (probabilistic definition). This is reminiscent of the equivalence in [14] between the VC dimension (combinatorial definition) and PAC learning (probabilistic definition).

2 Current Research

Computation with Artificial Neural Networks

The following summarizes my research on neural computation. In this line of research, I moved methodically from the most fundamental building block of a single neuron (which can be analyzed in great detail) to higher levels of complexity: larger networks with non-linear activation functions, convolutional neural networks at initialization, and finally the training process and how it affects generalization.

A single neuron

Neural networks are intricate mathematical objects to study. That is why I first investigated the perceptron algorithm, which is an abstraction of a biological neuron that was introduced in the 1950s by Rosenblatt [35], and has been extensively studied in many works (see the survey [31]).

Given a linearly separable binary labeled dataset of vectors, the perceptron algorithm outputs a separating hyperplane between two classes of vectors. However, if we assume the measurements of the data points are noisy, there is no guarantee that the algorithm's output will still classify the actual data vectors correctly. The maximal distortion the vectors may be perturbed by such that the dataset would remain linearly separable is called the *margin* of the dataset.

In [7], my collaborators and I identified compression properties of the perceptron and suggested improvements and modifications to the algorithm that make it less sensitive to noise. That is, the algorithm outputs a hyperplane with an almost optimal margin: the hyperplane is as far away as possible from all vectors in the dataset. Furthermore, I extended and implemented this modification to a general neural network. For a network with a single hidden layer containing 800 neurons and over the MNIST dataset of handwritten digits, I achieved a test error of 1.35% versus a test error of 1.6% for a typical training with cross-entropy loss [27].

Neural networks initialization

The performance of neural networks is susceptible to the choice of architecture and initialization. I demonstrated this in [9] for the class of symmetric functions

$$\mathbb{S}_n = \Big\{ \sum_{i=0}^{n} a_i \cdot \mathbf{1}_{|x|=i} \mid a_1, \ldots, a_n \in \{\pm 1\} \Big\},$$

where $x \in \{0,1\}^n$ and $|x| = \sum_i x_i$. The functions in this class are invariant under arbitrary permutations of the input coordinates. The parity function $\pi(x) = (-1)^{|x|}$ and the majority function are well-known examples of symmetric functions.

For this class, a careful choice of initialization for a fully connected network with one hidden layer can make the difference between success and failure in learning. This difference follows in three parts. First, the functions in this class can all be represented by a neural network with one hidden layer, so a solution exists in the parameter space. Second, for random initialization, which is how neural networks are typically initialized, the network fails to learn most functions from the class. Third, for a specialized initialization, stochastic gradient descent reaches zero empirical error and achieves a small test error. And to emphasize, the small test error is achieved for an over-parameterized network (the number of free parameters greatly exceeds the number of available samples). This counter-intuitive behavior is what makes neural networks so remarkable.

Since the initialization plays such a pivotal role, and conceptually, neural networks are successful because each subsequent layer allows for a better representation of the input space until linear separability is achieved, I asked in [10] the following question:

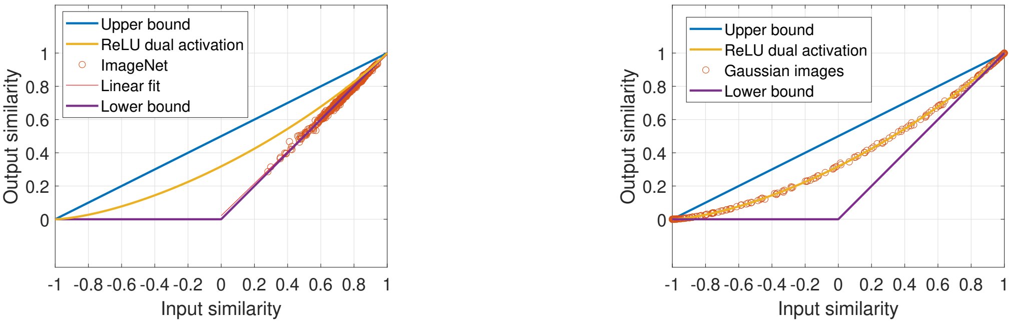
How does the geometric representation of a dataset change after the application of each randomly initialized layer of a neural network?

The Johnson–Lindenstrauss lemma [26] provides an answer for a linear fully connected network layer: With high probability over the weights of the network W, the geometry roughly stays the same, that is $\langle W \cdot x, W \cdot y\rangle \approx \langle x, y\rangle$. Similarily, [16, 21, 17] provided the following answer for a fully connected network layer with the ReLU ($:= \max\{0, x\}$) activation: for vectors x and y with cosine similarity $\rho := \frac{\langle x,y\rangle}{\|x\|\|y\|}$, the output cosine similarity $\frac{\langle \mathrm{ReLU}(W\cdot x), \mathrm{ReLU}(W\cdot y)\rangle}{\|\mathrm{ReLU}(W\cdot x)\|\|\mathrm{ReLU}(W\cdot y)\|}$ is concentrated around the **ReLU dual activation**:

$$\widehat{\mathrm{ReLU}}(\rho) := \frac{1}{\pi}\big(\sqrt{1-\rho^2} + \big(\pi - \cos^{-1}(\rho)\big)\,\rho\big). \tag{1}$$

In [10], my collaborators and I extended these results for convolutional neural networks (CNN) with the ReLU activation. This setting is intricate since the output similarity $\rho_{out} := \frac{\langle \mathrm{ReLU}(W*x), \mathrm{ReLU}(W*y)\rangle}{\|\mathrm{ReLU}(W*x)\|\|\mathrm{ReLU}(W*y)\|}$ ($* :=$ convolution) is no longer only a function of the input similarity ρ. See Figure 1: For natural images (red scatter plot in Figure 1a), the output similarity equals the input similarity, so the network behaves as a linear fully connected network; for Gaussian data (red scatter plot in Figure 1b), the output similarity follows the ReLU dual activation (yellow curve), so the network behaves as a ReLU fully connected network.

In [10], we explained this behavior: (1) Extension for the Johnson–Lindenstrauss lemma for linear convolutional networks: With high probability, $\langle W * x, W * y\rangle \approx \langle x, y\rangle$. (2) Tight lower and upper bounds for the output similarity $\max\{\rho, 0\} \leq \rho_{out} \leq \frac{1+\rho}{2}$ (purple and blue curves in Figure 1). (3) For Gaussian data, ρ_{out} is indeed concentrated around the ReLU dual activation. (4) For a model for natural images, a ReLU CNN behaves as a linear network $\rho_{out} \approx \rho$, the CNN preserves the geometry of the dataset.



(a) ImageNet, filter size $11 \times 11 \times 3$

(b) Gaussian images, filter size 11×11

Figure 1: Input and output cosine similarities of a single randomly initialized convolutional layer with 100 filters. Each red circle in the figures represents a random pair of images chosen from the corresponding dataset.

Noise and neural network training

Neural networks can approximate well any continuous function (assuming a sufficient number of adjustable weights), and their training process produces solutions with zero training error. Those properties combined were typically considered a hindrance to good generalization, yet, in practice, the networks generalize well. That is why it is likely that the training process has some form of *implicit regularization*.

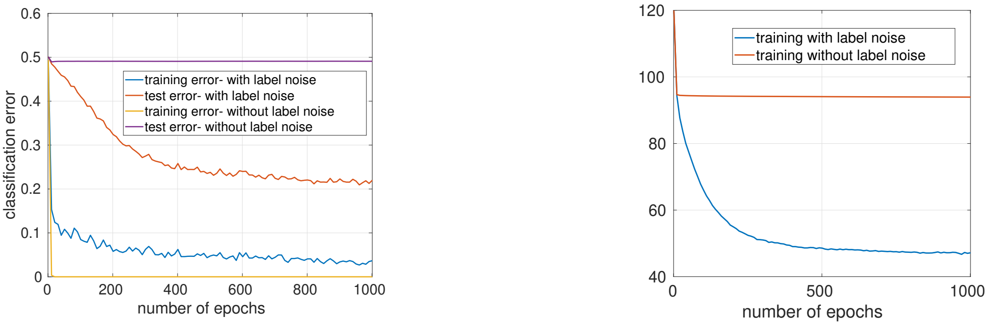
I focused in [1] on the role played by label noise in such implicit regularization. When label noise is applied, the true label is changed with probability p to a uniformly random label at each iteration during training. With $p = 0$, there is no noise, and with $p = 1$, the label is uniform.

Why would it make sense to add label noise on carefully collected data?

The answer is subtle since noise may cause difficulties in learning, and yet, noise can sometimes improve the accuracy of the model; label smoothing [36] is a canonical example. In [1], my collaborators and I demonstrated that label noise drives the network to a sparse solution in the following sense: for a typical input, a small fraction of neurons are active, and the firing pattern of the hidden layer is sparser.

Definition 1 *Let* $N(x) = W_2 \cdot \mathrm{ReLU}(W_1 \cdot x + B_1) + B_2$ *be a fully connected network with one hidden layer. For* x *in the dataset, the number of active neurons is* $A_N(x) = |\{i \mid w_i \cdot x + b_i > 0\}|$ *where* (w_i, b_i) *correspond to the weights of neuron* i *in the hidden layer. The typical number of active neurons is* $\mathbb{E}_x A_N(x)$*, where* x *is uniformly distributed in the dataset.*

In fact, an appropriate amount of label noise not only sparsifies the network activity but also reduces the test error. For example, see Figure 2: label noise induces a substantial improvement, 21% test error vs. 49% test error. For a theoretical analysis of such sparsification mechanisms, we focused on the extremal case of p = 1. In this case, the network withers, but surprisingly, in different ways that depend on the learning rate and the presence of bias weights, with weights either vanishing or neurons ceasing to fire (no activity of neurons = absolute sparsity). Then, a careful application of the intermediate value theorem shows that changing p tunes the activity of the network's hidden layer between no activity at $p = 1$ and the baseline activity at $p = 0$.



(a) error evolution

(b) typical number of active neurons

Figure 2: Learning the hypercube indicator function $\mathbb{1}_{\|x\|_\infty=1}$. The input dimension is $d = 60$, there are $4d$ neurons in the hidden layer and the learning rate is $h = 1/d$.

3 Future Research

Here are two related themes that I set for my research. In each, I briefly present my vision and long-term goal, followed by concrete initial research steps.

From observation to mathematics. I like exploring mathematical concepts that stem from real-life observations. Such observations can inspire new mathematical language that may change our scientific perspective. With that motivation, I transitioned from pure mathematics in my MSc to learning theory during my PhD. The transition was initiated by a fascinating lecture by May-Britt Moser in 2013 on her discovery of grid cells [23] (which led to her winning the Nobel Prize in 2014). Grid cells are neurons in the entorhinal cortex and are part of the brain's "navigation system" (see Figure 3). Can new mathematical theories be derived to explain such a geometrical phenomenon? As a long-term goal, I would like to retrace my steps and use my engineering and mathematical knowledge to address questions that stem from such neurological discoveries.

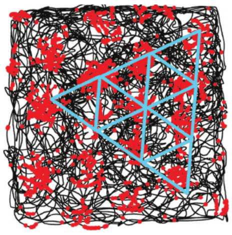


Figure 3: A grid cell from the cortex of the rat brain. The black trace shows the trajectory of a foraging rat in part of a 1.5m diameter wide square enclosure. Spike locations of the grid cell are superimposed in red on the trajectory. Each red dot corresponds to one spike. Blue equilateral triangles have been drawn on top of the spike distribution to illustrate the regular hexagonal structure of the grid pattern.

Over a shorter time frame, I take inspiration from the geometrical structure that arises in artificial neurons. My work in [10] is an example of such research. There, I rigorously explain Figures 1a and 1b that are a product of simple numerical experiments with nevertheless an intriguing outcome. Different types of datasets are embedded substantially differently through a CNN at initialization. Since a CNN is only one of many types of neural network architectures, many empirical patterns are waiting to be discovered, accompanied by follow-up geometrical questions. It is fruitful ground for research. An immediate example is explaining the behavior of a neural network at initialization when batch normalization [24] is added between each layer. I want to explain why after enough layers with batch normalization, the representation of the input dataset induced by the deep layers becomes almost orthogonal, as observed empirically.

Strengthening the connections between theory and practice. So often, theoretical work is initially motivated by real-life problems but slowly loses track of the original problem it was set to help and solve. My long-term goal is to bridge such gaps between theory and practice. Such a gap exists in the study of neural networks. For example, many published theoretical works focus on neural networks with no non-linearity. However, such linear neural networks can only solve linear problems. So it is unlikely to translate such works for practical gains.

This divide is most pronounced in our understanding of the generalization power of neural networks. This is evidenced in [25], where the authors survey and show that almost all generalization bounds are vacuous[1] and do not correlate well with generalization, so new complexity measures are required outside the scope of classical statistical learning.

Therefore, my research will focus on a closely-knit empirical and theoretical search for measures that predict the generalization of neural networks well. As a promising initial step, I will start with the PAC-Bayes bound [30], which is closely connected to the mutual information generalization bound presented in Section 1. PAC-Bayes is the only framework that proved to be non-vacuous in [25], an approach pioneered in [20]. In broad strokes, the PAC-Bayes framework predicts that neural networks generalize well because of a combination of two reasons: (1) the solution found is not too far away from the initialization. (2) the solution found is a flat minimum, that is, a minimum surrounded by a large volume of solutions with small empirical error.

Accordingly, my first step would be empirically measuring which of the two properties correlates better with generalization. Naturally, this empirical approach can be extended to other geometrical quantities not necessarily associated with established generalization bounds, such as activation sparsity, as in my work in [1]. A strong correlation between one of the new generalization measures will then lead the theoretical research in the right direction and help us design better networks.

[1] For example, plugging the network's parameters into the bound would yield the vacuous statement that the generalization error of the trained network will exceed 1234% ($\gg 1$) with probability of at most 123 ($\gg 1$).

My Publications

[1] Elisabetta Cornacchia, Jan Hązła, Ido Nachum, and Amir Yehudayoff. Regularization by misclassification in ReLU neural networks. `https://arxiv.org/abs/2111.02154`.

[2] Ido Nachum. Coarse symmetries of groups (MSc thesis). `https://www.graduate.technion.ac.il/Theses/Abstracts.asp?Id=28665`.

[3] Ido Nachum. On the information complexity of learning (PhD thesis). `https://www.graduate.technion.ac.il/Theses/Abstracts.asp?Id=31345`.

[4] Raef Bassily, Shay Moran, Ido Nachum, Jonathan Shafer, and Amir Yehudayoff. Learners that use little information. In *Proceedings of Algorithmic Learning Theory*, volume 83 of *Proceedings of Machine Learning Research*, pages 25–55, 2018.

[5] Ido Nachum, Jonathan Shafer, and Amir Yehudayoff. A direct sum result for the information complexity of learning. In *Proceedings of the 31st Conference On Learning Theory*, pages 1547–1568, 2018.

[6] Ido Nachum and Amir Yehudayoff. Average-case information complexity of learning. In *Proceedings of the 30th International Conference on Algorithmic Learning Theory*, pages 633–646, 2019.

[7] Shay Moran, Ido Nachum, Itai Panasoff, and Amir Yehudayoff. On the perceptron's compression. In *Beyond the Horizon of Computability: 16th Conference on Computability in Europe, CiE 2020, Proceedings*, page 310–325. Springer-Verlag, 2020.

[8] Emmanuel Abbe, Jan Hazla, and Ido Nachum. Almost-reed-muller codes achieve constant rates for random errors. *IEEE Trans. Inf. Theory*, 67(12):8034–8050, 2021.

[9] Ido Nachum and Amir Yehudayoff. On symmetry and initialization for neural networks. In *Latin American Symposium on Theoretical Informatics*, pages 401–412. Springer, 2021.

[10] Ido Nachum, Jan Hazla, Michael Gastpar, and Anatoly Khina. A Johnson-Lindenstrauss framework for randomly initialized CNNs. In *International Conference on Learning Representations*, 2022.

[11] Aditya Pradeep, Ido Nachum, and Michael Gastpar. Finite littlestone dimension implies finite information complexity. In *2022 IEEE International Symposium on Information Theory (ISIT)*, pages 3055–3060, 2022.

References

[12] Raef Bassily, Kobbi Nissim, Adam Smith, Thomas Steinke, Uri Stemmer, and Jonathan Ullman. Algorithmic stability for adaptive data analysis. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 1046–1059. ACM, 2016.

[13] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 464–473. IEEE, 2014.

[14] Anselm Blumer, A. Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, oct 1989.

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

[16] Youngmin Cho and Lawrence Saul. Kernel methods for deep learning. 2009.

[17] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: the power of initialization and a dual view on expressivity. pages 2261–2269, 2016.

[18] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 117–126. ACM, 2015.

[19] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

[20] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In Gal Elidan, Kristian Kersting, and Alexander T. Ihler, editors, *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press, 2017.

[21] Raja Giryes, Guillermo Sapiro, and Alex M Bronstein. Deep neural networks with random Gaussian weights: A universal classification strategy? *IEEE Transactions on Signal Processing*, 64(13):3444–3457, 2016.

[22] Peter D Grünwald. *The minimum description length principle*. MIT press, 2007.

[23] Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801, 2005.

[24] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456, 2015.

[25] Yiding Jiang*, Behnam Neyshabur*, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020.

[26] William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

[27] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

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

[29] Nick Littlestone and Manfred Warmuth. Relating data compression and learnability. Technical report, Technical report, University of California, Santa Cruz, 1986.

[30] David A. McAllester. Pac-bayesian model averaging. In Shai Ben-David and Philip M. Long, editors, *Proceedings of the Twelfth Annual Conference on Computational Learning Theory, COLT 1999, Santa Cruz, CA, USA, July 7-9, 1999*, pages 164–170. ACM, 1999.

[31] Mehryar Mohri and Afshin Rostamizadeh. Perceptron mistake bounds. *CoRR*, abs/1305.0208, 2013.

[32] Shay Moran and Amir Yehudayoff. Sample compression schemes for VC classes. *Journal of the ACM (JACM)*, 63(3):21, 2016.

[33] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[34] Ryan Rogers, Aaron Roth, Adam Smith, and Om Thakkar. Max-information, differential privacy, and post-selection hypothesis testing. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 487–494. IEEE, 2016.

[35] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[36] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[37] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, November 1984.