MACHINE LEARNING OF THE WELL-KNOWN THINGS

V. V. Dolotin,∗†‡ A. Yu. Morozov,∗†‡ and A. V. Popolitov∗†‡

Machine learning (ML) in its current form implies that the answer to any problem can be well approximated by a function of a very peculiar form: a specially adjusted iteration of Heaviside theta-functions. It is natural to ask whether the answers to questions that we already know can be naturally represented in this form. We provide elementary and yet nonevident examples showing that this is indeed possible, and suggest to look for a systematic reformulation of existing knowledge in an ML-consistent way. The success or failure of these attempts can shed light on a variety of problems, both scientific and epistemological.

Keywords: exact approaches to QFT, nonlinear algebra, machine learning, steepest descent method

DOI: 10.1134/S0040577923030091

1. Introduction

Machine learning (ML) now tries to find its use in more and more branches of both fundamental and applied science. For nearly a decade, it has been a tool in experimental (particle) physics, where it was used to accommodate for the peculiarities of a given instance of experimental apparatus (see, e.g., [1] for a review). Recently, there have been attempts to apply it to a number of topics in theoretical and mathematical physics as well. In particular, in [2] (also see [3] for a less ambitious attempt), ML was used to find conjectural relations between various knot theoretic invariants.1 In [7], ML approaches and techniques were used as a possible substitute for honest QFT calculations of scattering amplitudes in the Matsubara diagram technique at finite temperature.

These developments make a question of whether ML is a good, or at least passable, tool for the task of discovering new (mathematical) physics laws worth thinking about. In this form, the question is very broad and abstract, and to make it more concrete and approachable, we proceed as follows in this paper.

• We concentrate on a specific ML technique that prescribes searching for an answer (a new “law” or formula) in the form of a nested application of arctangent (“sigma”) functions and matrix multiplications (see Sec. 2). At first glance, this framework seems to be rather restrictive, but in fact there is a famous theorem [8] that any smooth function can be expressed with the help of this ansatz with an arbitrary precision.

∗Moscow Institute of Physics and Technology (National Research University), Dolgoprudny, Moscow Region, Russia, e-mail: popolit@itep.ru.
†Alikhanov Institute for Theoretical and Experimental Physics, National Research Center “Kurchatov Institute,” Moscow, Russia.
‡Kharkevich Institute for Information Transmission Problems, Russian Academy of Sciences, Moscow, Russia.
This work supported in part by the Russian Science Foundation (grant No. 21-12-00400).
1Knot theory and the associated representation theoretic aspects are one of the hottest current topics in mathematical physics, similar to CFTs in the 1980s, see [4]–[6] for recent fast-paced progress.

Translated from Teoreticheskaya i Matematicheskaya Fizika, Vol. 214, No. 3, pp. 517–528, March, 2023. Received December 2, 2022. Revised December 6, 2022. Accepted December 7, 2022.
• We consider a number of next-to-elementary examples/settings where a non-ML answer is well known and the ML answer can be produced after some thought (see Sec. 3). This gives us some control over what neural network can/should learn and hence to reason about the predictive power of neural networks.

The outcome of our analysis is that ML is no silver bullet. When the answer to be obtained by a neural network (NN) is known in advance, one can restrict the NN architecture (by properly fine-tuning the number of layers and their respective dimensions) such that this preconceived answer is indeed reproduced. What is alarming is that if a different answer is envisaged in advance (and hence the NN is fine-tuned with this answer in mind), then the NN converges to this new answer instead. Therefore, one can say that no new laws (formulas) are discovered by NNs except those that were already surmised by a researcher and built into the network (deliberately or not).

If one tries to circumvent these issues by deliberately not restricting the NN enough, then the learning process never terminates, being unable to discriminate between many apparently equivalent (to a machine, not to a human) formulas/laws.²

To summarize, ML is definitely an additional instrument in a researcher’s toolbox, which may prove indispensable in the hands of a skilled researcher. Yet, additional careful investigation is needed in order to establish correct practices to use this potentially powerful set of techniques so as to not add unnecessary and undesired presumptions to the studied problem.

2. The ML approach

Imagine that we want to associate a quantity $Z$ to a data $\hat{X}$ (recognize a cat in a photo, i.e., a special pattern in a set of pixels), i.e., find a function $g: \hat{X} \rightarrow Z$. For this purpose, we use a given set of examples, labeled by $\alpha$, when we postulate the answer $Z^\alpha$ for the set $\hat{X}^{(\alpha)}$. The idea of ML is to seek the function $g: \hat{X} \rightarrow Z$ by

1. solving the least-square problem—minimization of a functional like

$$\mathcal{L} := \sum_\alpha L^{(\alpha)} := \sum_\alpha |Z^{(\alpha)} - G(\hat{X}^{(\alpha)}|W)|^2; \quad (1)$$

2. using a special ansatz for $G(\hat{X}|W)$, where it is represented as a combination of some $\sigma$-functions

$$G(\hat{X}|W) = \sum_{i_1,i} W_{i_1,i}^2 \sigma(W_{i_1,i}^2 X_i + b_{i_1,i}^2) + b_{i_2}^i. \quad (2)$$

Adjustment parameters are the coefficients $W$, and a minimum of $L$ is found by the iterative steepest descent method, i.e., by solving the evolution equation

$$\dot{W} = -\frac{\partial \mathcal{L}}{\partial W} \quad (3)$$

for all the parameters $W$ and taking the limit of infinitely large time $t \rightarrow \infty$. As a result, we obtain a function $\mathcal{G}(\hat{X}) = G(\hat{X}|W)$ that looks like (2), i.e., is expressed in terms of the $\sigma$-functions with particular (optimized) parameters $W = W^\ast$.

²We touch on this intriguing subject here very slightly, leaving a detailed analysis for the future.
Usually, $\sigma(x)$ is a somehow smoothed version of the Heaviside $\theta(x)$, like $\sigma(x) = \text{arctanh}(kx)$, but one can consider other choices as well. An obvious generalization of ansatz (2) is an iteration power of $\sigma$,

$$G(\vec{X}|W) = \sum_{i_1, i_2, \ldots, i_m} W^{i_1}_{i_1} \sigma(W^{i_2}_{i_2} \sigma(\cdots \sigma(W^{i_m}_{i_m} X_1 + b^{i_1}_{i_1}) + b^{i_2}_{i_2}) \cdots) + b^{i_m}_{m+1}. \quad (4)$$

The number $m$ of $\sigma$-functions in this “approximation” is called the depth (i.e., the number of layers) of the ML algorithm. It can also involve a more sophisticated way to combine $\sigma$-functions, which is called the architecture of the algorithm. Also the cost functional $L$ can be more complicated, for example, one does not need to make the arguments of $\sigma$ linear in $X$: already higher-degree polynomials can often lead to exponential increase in the efficiency of particular algorithms. Finally, more elaborated setups, like unsupervised learning or learning of GANs (generator adversarial networks) are possible. For simplicity, in this initiatory paper, we restrict ourself to the trivial multilayer architecture with the simplest Heaviside $\sigma(x) = \theta(x)$ and the simplest cost functional (1).

In other words, ML suggests to look for the answer in a somewhat special form (4), which does not seem familiar from other branches of science. Still, the apparent successes of ML imply that this shape (iterative combinations of $\sigma(x)$) could be adequate. Then we face a natural question: if the answers that we already know can be reformulated and brought to this form, then how easy is it and how natural is such a reformulation? These are the questions to be addressed in this paper. Its main content is just a set of elementary examples that can help to open the secret.

In these examples, we find an expression for the map $G(\vec{X})$ known from ordinary science in terms the $\sigma$ (actually, Heaviside) functions, and call it $\mathcal{G}(\vec{X})$. This then implies an expression for the ML test function $G(\vec{X}|W)$ that can converge to

$$\mathcal{G}(\vec{X}) \equiv G(\vec{X}|W) = \mathcal{G}(\vec{X}). \quad (5)$$

In other words, if the known answer can be “Heavisided,” then there is an ML process that has a chance to reproduce it. If it cannot, then there is a problem: the ML “answer” may not be true. An apparent resolution in that case is the instability of the ML “answer” with respect to the variation of sample data.

3. Examples

3.1. The zero of a function. We have a function $y = F(x)$ and we wonder what its zero is. The answer to this question is

$$\mathcal{G} \equiv \text{zero of } F = \int_{dx} x F'(x) \delta(F(x)) \quad (6)$$

provided that $F(x)$ only has one zero (otherwise we obtain the sum of coordinates of all zeroes).

The conversion into the ML form in this case is immediate:

$$\mathcal{G} = \int_{dx} x \theta'(F(x)). \quad (7)$$

The left-hand side gives the correct answer when the function has a single zero (i.e., when the problem is well posed), and the right-hand side expresses this function in the ML form (2).

The discrete version of the formula, when the integral is replaced with a sum over the points of, say, a unit segment $x \to k/N$, $k = 1, \ldots, N$, takes the form

$$\mathcal{G} = \sum_{k=1}^{N} \frac{k}{N} \left\{ \theta \left( F \left( \frac{x_k}{N} \right) \right) - \theta \left( F \left( \frac{x_{k-1}}{N} \right) \right) \right\} = \sum_{k=1}^{N} \frac{k}{N} (\theta(f_k) - \theta(f_{k-1})). \quad (8)$$
The Heaviside function converts the values of the function \( f = F(x) \) into just two,

\[
\theta(f) = \begin{cases} 
1, & f \geq 0, \\
0, & f < 0, 
\end{cases}
\]

(9)
i.e., does not change unless the graph of a smooth function crosses zero. The formula picks up the crossing and weights it appropriately.

Thus, the ML cost function \( L \) in (1) can be taken in the form

\[
L\{f\} = \left| \sum_{i=1}^{N} \frac{1}{N} (\theta(f_i) - \theta(f_{i-1})) \right|^2 - \sum_{i,i,i} W_{1i}^{\prime} \sigma(W_{1i}^{\prime} f_i) \]

(10)
and we can expect the steepest descent to converge to the point

\[
W_{1i}^{\prime} = \delta_{i,i}, \quad W_{2i}^{\prime} = \frac{i_1}{N} - \frac{i_1 + 1}{N}.
\]

(11)
It can be verified that this is indeed the case for \( \sigma \) close enough to \( \theta \).

### 3.2. An alternative architecture.

Similar considerations, even if slightly different in details, lead to “another” shape (architecture) of the answer to the zero-finding problem. Here they are.

The first step in the (supervised) ML task is to provide computer with a list of “correct” solutions to the problem (the so-called training set), from which it would hopefully extract the general recipe. In this case, training data should consist of a large number of functions together with their zeroes. We assume for simplicity that every function in the training set does indeed have exactly one zero. The representation of the training data is significant for the efficiency of training. We assume that every function is given by its values on some mesh, say, \( x_i = i, i = 1, \ldots, M \). In this case, the recipe for finding putative zeroes of a function would be as follows: loop through consecutive pairs of function values \((y_i, y_{i+1})\), ...; if the signs of the pair differ, we have a conjectural zero between \( x = i \) and \( x = i + 1 \).

This recipe translates to the ML-type formula

\[
x_{\text{zero}} = \text{AvgCoord} \left( \sum_{m=1}^{M} \sum_{l=1}^{2M} \sum_{k=1}^{M} \sum_{i=1}^{M} \sigma(B_{m}^{(3)} + W_{m}^{(2)} \sigma(B_{l}^{(2)} + W_{l}^{(2)} \sigma(B_{k}^{(1)} + W_{k}^{(1)} y_{i}))) \right),
\]

(12)
where

\[
W_{k}^{(1)} = \delta_{k,i} - \delta_{k,i+M}, \quad B_{k}^{(1)} = 0,
\]

\[
W_{l}^{(2)} = \delta_{l,k} + \delta_{l,k+M+1} + \delta_{l+M-1,k}, \quad B_{l}^{(2)} = -1.5,
\]

\[
W_{m}^{(3)} = \delta_{m,l} + \delta_{m+M,l}, \quad B_{m}^{(3)} = 0,
\]

\[
\text{AvgCoord}(v) = \sum_{i=1}^{M} i \cdot v_i.
\]

Indeed, after the application of the first network layer \( \sigma(B_{k}^{(1)} + W_{k}^{(1)} y_{i}) \), we have a \( 2M \)-dimensional vector of (roughly) 1s and 0s, where a unit occurring in any of positions 1, \ldots, \( M \) indicates that the corresponding \( y_{i} \) is positive, and a unit occurring in one of the positions \( M+1, \ldots, 2M \) indicates that the corresponding \( y_{i} \) is negative.

After the application of the second layer \( \sigma(B_{l}^{(2)} + W_{l}^{(2)} \cdot) \), we have a \( 2M \)-dimensional vector with a unit in the \( l \)th position meaning that \( y_{l} > 0 \) and \( y_{l+1} < 0 \) and a unit in the \( l + M \)th position, that \( y_{l-1} < 0 \) and \( y_{l} > 0 \). Such behavior is ensured by the “bias” term \( B_{l}^{(2)} \).
After the application of the third layer, we have an $M$-dimensional vector with a unit in the $l$th if the function changes sign in the vicinity of $l$.

Finally, the AvgCoord calculates the average coordinate of a vector.

We can already see the problem/virtue of the ML approach. Depending on our prior knowledge about the problem, which we introduce into our ML model in the form of architecture (an ansatz), the process converges to either (11) or (13), in any case confirming our presupposition. We see the potential sprouts of the unfalsifiability problem: the ML engine confirms (fulfills) both our initial plausible assumptions (prophecies). Therefore, additional considerations are required to select the correct (most economical) hypothesis.

3.3. Multiplication problem. Even more fundamental could be an ML approach to addition and multiplication of integers. Every operation allows splitting a quantity into elementary objects. For addition, such a building block is just 1: every natural number can be obtained by adding units. But the elementary objects for multiplication are numerous and form a badly conceived set of prime numbers—the most mysterious of all phenomena in nature, to which all the other mysteries are being unhurriedly reduced (through the study of adels, zeta functions, motives, and so on). As one often says, “integers are the creatures of God”—as is, probably, the human intelligence. It is a crucial question whether ML can learn the laws of multiplication and the properties of primes.

In fact, addition and multiplication are regarded as trivial (well understood) inputs in (4), and one can think that there is no need to express them through Heaviside compositions. But we note that Eq. (4) involves addition and multiplication of real and complex numbers, not integers. For integers and rationals, the problem of Heavisidization makes sense, and can easily be resolved.

For addition of positive integers, a possible formula is

$$a + b = \sum_{i=1}^{\infty} \theta(a - i) + \sum_{i=1}^{\infty} \theta(b - i),$$

(14)

where the two sums respectively contain $a$ and $b$ units. This is independent of the upper limit, which can be just some big number, exceeding all relevant $a$ and $b$. In the ML language, this solution implies that we use the test function

$$G = \sum_i w_i^1 \sigma(w_0^1 x_1 + w_2^1 x_2 + b_0^1) + b_1.$$ (15)

Similarly, for multiplication

$$a \cdot b = \sum_{i,j} \theta(\theta(a - i) + \theta(b - j) - 2).$$ (16)

Contributions are made only by the points $(i,j)$ where the argument of the outer $\theta$ is nonnegative, i.e., where both $\theta(a - i)$ and $\theta(b - j)$ are units. That there are exactly $a \cdot b$ such points is not difficult to verify.

Of course, these rules can easily be extended to rationals, at least with a somehow limited denominator, but this is unsurprising: the true borderline is drawn in number theory between rationals and the other types of numbers, like algebraic or real.

For multiplication, the problem looks more interesting. A possibility to simplify it is to work in the binary system and devise a Heaviside-based multiplication method for zeros and units.

3.4. Application to classification problems. A standard ML task is classification: building a function on a configuration space $X$ with values in a discrete set $Z$ of classes.
3.4.1. Example: 1D. We take an example where \( X = \mathbb{R}^1 \) and \( Z = \{0, 1\} \), i.e., we classify points on the real line as belonging to two possible classes. The map that we want to approximate by our network is given by

\[
g: X \to \begin{cases} 
0, & x < 2, \\
1, & 2 \leq x \leq 3, \\
0, & x > 3,
\end{cases}
\]

i.e., points in the segment \([2, 3]\) belong to the class 1, and the rest to 0.

We take a level-1 network. The output value has the general form

\[
G = \sum_i w_i^1 \sigma(w_i^0 x + b_i^0) + b_1.
\]

If we take \( \sigma \) to be a Heaviside function, then we can indeed express the well-known answer in the form

\[
G(x) = \theta(x - 2) - \theta(x - 3) + 0.
\]

This means that we can make the single-layered network with two cells, whose parameters converge to the values

\[
\begin{align*}
\overline{w}_0^1 &= \overline{w}_0^2 = 1, \\
\overline{b}_0^1 &= -2, \\
\overline{b}_0^2 &= -3,
\end{align*}
\]

We see that in this way we can actually express any step function on the line, which solves the classification problem for a 1-dimensional configuration space \( X \).

3.4.2. Example: 2D. For simplicity, we consider just a sector on the plane \( X = \mathbb{R}^2 \). This means that the map we want to approximate by our network is

\[
g: (x^1, x^2) \mapsto \begin{cases} 
1, & x^1, x^2 > 0, \\
0, & \text{otherwise},
\end{cases}
\]

which is the characteristic function of the sector.

The solution is provided by a two-layered network. For such a network, in general,

\[
G = \sum_j w_j^2 \sigma \left( \sum_i w_i^j \sigma(w_i^0 x_1 + w_i^0 x_2 + b_i^0) + b_i^j \right) + b_2.
\]

The approximation to our characteristic function becomes exact for \( \sigma = \theta \):

\[
G(x_1, x_2) = -\theta(\theta(-x_1) + \theta(-x_2) - 1) + 1.
\]

When both \( x_1 \) and \( x_2 \) are positive, the argument of the outer \( \theta \) is \(-1\), and \( y = 1 \). When any of \( x_1 \) or \( x_2 \) is negative, the argument is 0 or 1, and \( y = 0 \).

Thus the relevant network is two-layered with \( 2 + 1 \) cells, whose parameters converge to

\[
\begin{align*}
\overline{w}_0^{11} &= \overline{w}_0^{22} = -1, \\
\overline{w}_1^{11} &= \overline{w}_1^{12} = 1, \\
\overline{w}_2^{1} &= -1, \\
\overline{b}_2 &= 1.
\end{align*}
\]

A combination of sector characteristic functions may provide an approximation (with the accuracy depending on the number of network cells) to any step function on the plane.

The generalization to higher dimensions seems straightforward: describing a sector in \( D \) dimensions requires a \( D \)-level network.
3.5. **Beyond classification.** We have learned how to find a root of \( F(x) \) and demonstrated that the method is indeed the same as the standard ML approach to classification problems.

But is this method powerful enough to obtain more explicit formulas, such as

\[
x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2}
\]

for \( F(x) = ax^2 + bx + c \)?

It is hard to give an immediate affirmative answer; at least, there is an apparent problem to address and examine.

4. **The moral**

The above examples are instances of the archetypal scientific question: solve the equation \( F(x) = 0 \). As we just saw, it can be answered by ML. Moreover, we identified the reason for this: the existence of formula (8) for the answer \( G(x) \) made of the Heaviside functions \( \theta(F(x)) \). Thus we should ask ourselves: can all the available answers to all scientific questions we know so far be expressed as linear combinations of iterated Heaviside functions?

If so, then we can hope to adjust the coefficients of these linear combinations by ML to the true values.

If not, ML will not provide us with a true answer.

If we use a convergent algorithm, some answer will be given: the program will return definite parameter values. It may even perfectly match the sample data, but would still have nothing to do with the true answer. In particular, it may not be stable under changing the training sample: the adjusted coefficients would change dramatically when the sample is modified, even slightly.

Given this problem, we can approach it from two opposite sides:

A) find a representation of the answer in terms of Heaviside functions or prove that it does not exist for any finite number of iterations;

B) check the stability of the ML output under the change of the training sample.

Approach A is a task for theorists, and it is not altogether simple, as we have hopefully demonstrated with the above examples. Approach B can be handled by pure computer methods, at least if we restrict the number of layers in the network to be reasonably low.

More questions arise:

1. Can we express the answer to any scientific problem in the form of a Heaviside composition?

2. If so, is there a universal way to convert any known answer to such a form?

3. If not, what are the obstructions to such Heavisidization?

This set of questions resembles the one that led to constructive mathematics, only the current restrictions seem to be much stronger.

It is intuitively clear for many that in most scientific problems, experiments with approach B will demonstrate that the ML method does not work. Still, as we have explicitly demonstrated with the simplest examples of addition and multiplication and of finding zeroes of a function, the answer, perhaps surprisingly, can be obtained in this way.
Moreover, the answer for the simplest case of finding a zero of a function was even not unique, and this is the tip of the iceberg of another collection of subtleties worth diving into. Another task is to look at explicit formulas for algebraic numbers, i.e., solutions of polynomial equations, both when explicit formulas are known and when only cohomological constructions in the spirit of nonlinear algebra [9], [10] are available.

We hope that this short essay can stimulate the search along both directions listed above and help understand the limits of ML and the fate of human mind in the times of artificial intelligence. Is science reducible to the steepest descent?

5. Discussion and conclusion

Machine learning [11] nowadays penetrates all the aspects of our everyday life and promises to provide humanity with a new magic tool for every new problem that we can happen to face. Computer facilities do not seem to be restricted in any fundamental way and can one day overcome the capacities of human mind—which, by contrast, we cannot improve in any way without interfering into the basic biology of ourselves and giving up our humanness for something else, biological or not.

This progress of artificial intelligence once again poses the question of what our intelligence is and what, if any, its potential advantages are over the artificial one. This is one of eternal questions, but today we can pose it more sharply. This is because all the spectacular successes of modern ML make use of the simplest possible steepest descent method, which is tremendously far from much more advanced products of human mind, like physical theories and the dreams of the “theory of everything” [12]–[15], which at the moment looks “relatively simple” but still considerably more sophisticated than the offer of ML. Is it possible that just the most primitive optimization method allows learning and comprehending everything that human genius has achieved over the course of centuries?

The answer seems obvious, at least to us. And it is an emphatic no: artificial intelligence, as we know it now, is extremely efficient in solving optimization problems, i.e., in adjusting parameters, but is quite useless in finding suitable algorithms, i.e., understanding the laws governing particular phenomena. In other words, if we have no idea of the structure within which the parameters are to be adjusted, the efficiency of the adjustment is pretty low. We cannot generate new knowledge in this way, only improve the quality of the existing one. Laws of nature, as we know them, are not expressible in terms of Heaviside functions.

This seems to be a rather obvious idea, but papers keep appearing ([16]–[20] to list just a few) where ML is “applied” to science, which might result in the overall impression that scientific problems can indeed be solved in that way. The aim of this short essay was to explain why and in what sense this impression is probably wrong.

6. Reservation/disclaimer

To avoid possible misunderstanding, the problems raised in this note concern the particular method of ML. It should not be confused with other kinds of Big Data analysis, including the powerful induction method based on computer analysis of various phenomena, which we advocate and use for many years [10], [21]–[29], and which is gaining in importance as the computer capacities keep increasing. However, using computers to strengthen human intellect is not the same as using computers to supersede it. The question addressed in this note is whether the latter is also possible and what can be done to understand whether it is an option for ML.

Acknowledgments. We are indebted to A. Anokhin and S. Barseghian for the useful discussions.

---

3In fact, some references in [16]–[20] also deal with this kind of approach rather than with the true ML.
Conflicts of interest. The authors declare no conflicts of interest.

REFERENCES

1. D. Guest, K. Cranmer, and D. Whiteson, “Deep learning and its application to lhc physics,” Ann. Rev. Nucl. Part. Sci., 68, 161–181 (2018).
2. J. Craven, M. Hughes, V. Jejjala, and A. Kar, “Illuminating new and known relations between knot invariants,” arXiv:2111.03470.
3. J. Craven, M. Hughes, V. Jejjala, and A. Kar, “Learning knot invariants across dimensions,” arXiv:2112.00016.
4. E. Lanina and A. Morozov, “Defect and degree of the Alexander polynomial,” Eur. Phys. J. C, 82, 1022, 16 pp. (2022).
5. E. Lanina, A. Sleptsov, and N. Tselousov, “Implications for colored HOMFLY polynomials from explicit formulas for group-theoretical structure,” Nucl. Phys. B, 974, 115644, 30 pp. (2022).
6. V. Mishnyakov, A. Sleptsov, and N. Tselousov, “A novel symmetry of colored HOMFLY Polynomials Coming from s(N|M) Superalgebras,” Comm. Math. Phys., 384, 955–969 (2021).
7. S. Chen, O. Savchuk, S. Zheng, B. Chen, H. Stoecker, L. Wang, and K. Zhou, “Fourier-flow model generating Feynman paths,” arXiv:2211.03470.
8. K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” Neural Networks, 2, 359–366 (1989).
9. I. M. Gel’fand, M. M. Kapranov, and A. V. Zelevinsky, Discriminants, Resultants, and Multidimensional Determinants, Birkhäuser, Boston, MA (1994).
10. V. Dolotin and A. Morozov, Introduction to Non-Linear Algebra, World Sci., Singapore, arXiv:hep-th/0609022 (2007).
11. Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, 521, 436–444 (2015).
12. A. Grothendieck, “Sketch of a programme,” in: Geometric Galois Actions. I. Around Grothendieck’s Esquisse d’un Programme (Luminy, France, August, 1995), Proceedings of the Conference on Geometry and Arithmetic of Moduli Spaces, London Mathematical Society Lecture Note Series, Vol. 242 (L. Schneps and P. Lochak, eds.), Cambridge Univ. Press, Cambridge (1997), pp. 243–283.
13. R. P. Langlands, “Problems in the theory of automorphic forms to Salomon Bochner in gratitude. Modern Harmonic Analysis and Applications,” in: Lectures in Modern Analysis and Applications III, Lecture Notes in Mathematics, Vol. 170 (T. C. Taam, ed.), Springer, Berlin, Heidelberg (2006), pp. 18–61.
14. M. B. Green, J. H. Schwarz, and E. Witten, Superstring Theory, Vol. 1–3, Cambridge Univ. Press, Cambridge (2012).
15. A. Yu. Morozov, “String theory: what is it?,” Phys. Usp., 35, 671–714 (1992).
16. G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborova, “Machine learning and the physical sciences,” Rev. Modern Phys., 91, 045002, 39 pp. (2019), arXiv:1903.10563.
17. H. Erbin and R. Finotello, “Machine learning for complete intersection Calabi–Yau manifolds: a methodological study,” Phys. Rev. D, 103, 126014, 40 pp. (2021), arXiv:2007.15706.
18. H. Chen, Y. He, S. Lal, and S. Majumder, “Machine learning Lie structures & applications to physics,” Phys. Lett. B, 817, 136297, 5 pp. (2021), arXiv:2011.00871.
19. Y.-H. He, K.-H. Lee, T. Oliver, and A. Pozdnyakov, “Murmurations of elliptic curves,” arXiv:2204.10140.
20. J. Bao, Y.-H. He, E. Heyes, and E. Hirst, “Machine learning algebraic geometry for physics,” arXiv:2204.10334.
21. A. Levin and A. Morozov, “On the foundations of the random lattices approach to quantum gravity,” Phys. Lett. B, 243, 207–214 (1990).
22. V. Dolotin and A. Morozov, “Algebraic geometry of discrete dynamics. The case of one variable,” arXiv:hep-th/0501235; “Introduction to Khovanov homologies. III. A new and simple tensor-algebra construction of Khovanov–Rozansky invariants,” Nucl. Phys. B, 878, 12–81 (2014), arXiv:1308.5759.
23. A. Yu. Morozov and Sh. R. Shakirov, “New and old results in resultant theory,” Theoret. and Math. Phys., 163, 587–617 (2010), arXiv:0911.5278.
24. A. Mironov, A. Morozov, Sh. Shakirov, and A. Sleptsov, “Interplay between MacDonald and Hall–Littlewood expansions of extended torus superpolynomials,” *JHEP*, **05**, 070, 11 pp. (2012), arXiv:1201.3339.

25. http://wwwth.itep.ru/knotebook/ and references therein.

26. A. Mironov, A. Morozov, An. Morozov, and A. Sleptsov, “Gaussian distribution of LMOV numbers,” *Nucl. Phys. B*, **924**, 1–32 (2017), arXiv:1706.00761.

27. A. Anokhina, A. Morozov, and A. Popolitov, “Nimble evolution for pretzel Khovanov polynomials,” *Eur. Phys. J. C*, **79**, 867, 18 pp. (2019), arXiv:1904.10277; “Khovanov polynomials for satellites and asymptotic adjoint polynomials,” *Internat. J. Modern Phys. A*, **36**, 2150243, 24 pp. (2021), arXiv:2104.14491.

28. A. Anokhina, “Talk at StringMath-2019,” https://www.stringmath2019.se/wp-content/uploads/sites/39/2019/07/Gong_Show_StringMath2019.pdf.

29. A. Mironov and A. Morozov, “Superintegrability summary,” *Phys. Lett. B*, **835**, 137573, 10 pp. (2022), arXiv:2201.12917.