The Master Algorithm

How the Quest for the Ultimate Learning Machine will remake our World

P.Domingos

March 27-29, 2018

What is a program? A program is a set of explicit step-wise instructions executed sequentially. Typically at each stage a decision has to be made, depending on the result of previous computations a new computation has to be made. A computation itself is a program. What we have here is at first sight a circular definition, or what is almost the same thing, an infinite regress, both being anathema to any logical enterprise. Of course in reality we are reduced to irreducible programs, programs that cannot be decomposed into other programs (so called sub-routines), which consists of just some very simple procedures and combinations inductively defined (this is reminiscent of Euclid, starting with self-evident axioms and then building up from there). This was succinctly illustrated by Turing in his ideal machine, the purpose of which was to explain once and for all, what is a computation, a philosophical and metaphysical task, which has had a tremendous influence on life in the last eighty years or so. Philosophical ruminations are not always confined to the lofty sphere, but can have tangible consequences indeed. The Turing machine was, in case it should be literally implemented, a very low-tech solution, and as such more of a curiosity, but that does not matter as the principle remains the same. Basically a program can be encoded as a number (a sequences of 0's and 1's) which can be represented mechanically in a device, the abacus being perhaps the earliest computer[1]. One needs not to go into detail, suffices it to say that a number can be represented as a string of switches which are either on or off. This is a first step going from something that only exists in the imagination to something that exists in the physical world. Yet the number is just lying there doing nothing. Now a machine is something which works. It is not inert but it moves, it changes its states, be it in a very limited and controlled way. What the computer can do is to switch the switches. It can switch on and off thus change a representation of a number. To decode an encoded number is to translate it into a sequence of basic movements of switching. This is a physical event taking place in real time and space. Given an input, an encoded number, in the end there will be an output (provided the process comes to a stop, which a priori is not clear). And as far as the machine is concerned that is the end of it. One encoded number has been replaced by another, so what? The next crucial step is to interpret that number and take the necessary actions. Before that an execution of a program is but a solipsistic exercise. The crucial thing as to the presence of computers is how they are integrated in the infra-structure. The mere existence of a number (all numbers exist do they not?) does not mean the destruction of the world, but could mean

[1] Roman numerals are notoriously unadapted to arithmetical manipulation, but the point is that they were never used for that, paper and pencil computation is a later invention, one is tempted to say an unfortunate deviation

it, when that existence is physically realized in some crucial way. Typically the output is used for other machines to be presented to the human agent in some convenient way, say on a monitor or as a printed sheet. But it can of course, and is often, going much further. It would be trivial in principle to connect it to the dispatch of nuclear warheads with no human intervention. In that case a given number can spell disaster, but the danger is not so much in the number itself, which will merely give a proximate cause, but in the decision to make this connection.

Now the idea behind the computer, is a very simple one, and to many people something that they immediately and instinctively take to. Throughout human history of invention this kind of thinking has been prevalent. Take such an example as a clock-work. It works on the same principles of cause and effect and how to structure them [2]. What made the great difference was the revolution of hardware, electronics rather than cumbersome mechanical devices, which made it feasible to implement those ideas physically because of the memory capacity and the speed of operations. A creator of a program is an engineer that makes a virtual machine, but as opposed to the engineer or the watchmaker of the past, the exercise is essentially totally cerebral, one can dispose of manual dexterity for one thing and many other tacit skills that are necessary in order to deal with the real world.

To be a programmer does not take anything but a certain basic understanding of the principles, i.e. the need to envision a solution of a problem and to be able to break it up in subroutines making a structure, and in particular to break it up into the basic steps[3]. A programmer is typically competent, but he or she is never a genius at her task, and hence programming cannot be compared to mathematics. A mathematician can make a break through, a programmer cannot, more or less by definition[4].

Now any program is made for a very specific purpose, and the code itself is fairly transparent. The programmer in fact has already a solution to the problem, the machine is there only to speed the process up, to actually physically accomplish what the progarmmer can do in principle in the imagination. The result of any program is both deterministic as well, apart from the trivial cases, unpredictable. Any computation is unpredictable, otherwise there would be no point in making it. But there are orders of unpredictability, the result will be of a certain form, and even if it goes badly wrong, it will not mean the destruction of the world, at worst the output will be gibberish of no concern to anyone (except possibly the programmer). The purpose of the present book is to impress on the reader that the art of programming has taken a quantum leap carrying it to a new level of unpredictability. In order to use computers you have to program them and that is a

[2] In the work of Harrison whose clock solved the problem of the longitude, one may apparently discover sub-clocks corresponding to failed experiments.

[3] Now programming can be made at many different levels, few people nowadays do it at the level of machine language, on the other and if the basic steps are on too high a level, they become too many and when problems arise hard to handle as they are by intents black boxes. Then the manipulations of them becomes too boring. But this is the way it is done, because what matters is the results not the intellectual satisfaction of the programmers

[4] Of course to design clever algorithms and new ideas of how to program is something different from the actual work of translating ideas into a workable code. This I do not consider part of programming per se in the focused sense.

very labor intensive part and which makes great demands on a huge pool of competent professionals, a large fraction of whom are drawn from Third World countries. Would it not be great if that could be automated, that computers could write their own program? After all it does not take too much imagination to write programs, anyone with an instinctive knack for it can do it, and many do. There may only be a thousand serious mathematicians in the world, but surely there are at least potentially millions of competent programmers. Now we enter the realm of Artificial Intelligence (AI), or more specifically Artificial General Intelligence (AGI), which is the ultimate subject of the book. Even in the 40's computers could do some work indicative of intelligence, such as finding their ways through labyrinths, initial success of which quickly led to over-inflated expectations of the ultimate capabilities of computers as well as overly optimistic time estimates. Traditionally a computer is just the implementation of a program. The computer is not smarter than the program. The point of the hardware is just to make the spirit into flesh. A program does not exhibit any intelligence beyond that which the programmer has put into it. And a program is just designed to accomplish a well-defined task, otherwise you could not program a solution to it. But what kind of things can a program do? You can program it to play chess, but can you program it to program chess programs? I mean in any essential way. Can you program a computer to program better than you can do? The short answer is no. If you know the rules of chess it is not that difficult to get started. The program may not be very powerful, but it is not inconceivable that you will produce a program that beats you (it helps being a lousy chess player). Is that program smarter than you, or does it make up in speed and bulk what it lacks in imagination (but if imagination consists in the number of scenarios you can make up it is easy to be imaginative)? But if you try to program a computer to program it is a very different level. This should not be confused with the trivial task of letting a computer write down a computer program of your devising, but one in which it figures out the strategy itself. Can this be done without a sense of intention?

The road to excessive intelligence does not go via standard programs in which you simply have translated a solution into terms that a computer can handle. This is often referred to as knowledge engineering. In order to program a programmer you need to know how a programmer really thinks deep down, and if you have no idea of that, as to be expected, you are stuck. Something entirely new is required, and that is the revolution the author is referring to. What it amounts to is to feed the program a very large input which will guide the program in ways we cannot really anticipate. In short the program is trying to find underlying patterns in the input and once it has figured them out act upon them. This is known as learning. Learning goes on at many different levels and is a kind of evolution. Hypotheses are produced and tested and rejected if not being up to par. It can be the individual that learns to survive in a hostile world by trial and error, or it could be evolution that brings about more and more adapted organisms. Now when it comes to the individual we do not have a clean slate but many things are so to speak hardwired into it, a hard-wiring that has been put in place by evolution. The life time of an individual is short, that of natural evolution huge, and the mystery is that the latter started more or less with a clean slate[5].

[5] In learning we have inbred structures which makes learning much more efficent, but also much less flexible (as to be explicated below). You can only learn somethings. Evolution on the other hand starts

Now when it comes to constructing a learner we can one one hand be very specific in the way it learns, meaning that we specify what kind of patterns it will be looking for. It is reasonable to think that in the case of lower organisms, the slate is very much filled up and that the process of learning hence becomes simpler and also much more efficient. However, lower organisms are evolved for very specific tasks, and such learners do not learn very exciting things, what it learns we have already told it, and in that case, it makes more sense to program that directly. What we are looking for are flexible learners able to look for very different kinds of patterns and relate to very different situations. The challenge now becomes to make the learning process as general and as little ad hoc as possible. This may mean, but not necessarily so, that the programs are very simple. Classical programs tend to be longer and more complicated the more sophisticated they are, while the programs we are now looking for should be simpler the more flexible they are.

Now the success of this approach is based on the metaphysical assumption that the data instructs you, that patterns are implicit in them, and eventually emerge inevitably. This is the classical view of scientific work to which the public and politicians, and I fear, many scientists, especially the most specialized, adhere to, and which is the basis for the expressions such as the scientific method or that something has been scientifically proved. Another view of science is given by Popper who claims that there is no way of automating the finding of patterns. How hypotheses are formed is a great mystery, just as that of artistic creation. That science proceeds by trial and error is at least accepted by scientists themselves, and hypotheses are constantly tested and as a result modified to a lesser or greater degree (often rejected altogether). It is easy to come up with patterns given data, and it is a classical principle, as enunciated by Peirce and others at the end of the 19th century[6], never to use the same data that stimulated the formulation of a hypothesis to test it with. This is simply begging the question.

Now with the Popper approach it is impossible to automate the process of formulating hypotheses, what we can do is to have some general ways of generating them and hope for the best. If Popper is wrong, it means that human scientific thinking can indeed be fully automated, that the world is knowable without divine inspiration, and that indeed a master algorithm can be devised, the holy grail of all AI. The issue is of course metaphysical, as noted, and may not concern us now at our rather primitive stage.

The author presents five approaches to the learning problem. As this is a book aimed for the wide audience, in particular there is no mathematics, there is thus a waving of hands and more or less confusing analogies, which in the end leaves the expert as well as the neophyte bewildered. The lay reader may think that he understands, and get seduced, the expert reader merely becomes frustrated. Five approaches are presented, the symbolic, the connectionist, the evolutionary, the Bayesian and finally the one by analogy. All of them have their advantages as well as their disadvantages. What is needed is a master algorithm that transcends them all and becomes the ultimate learner, capable in principle

---

more or less on a clean slate, and the structures that individuals come equipped with for fast learning adaptation, are not Godgiven but the result of evolution. Evolution, which transcends individuals, can thus be thought of as a meta-learning.

[6] This insight obviously goes back much longer, but it was only when science became an industry, it became necessary to warn against it

of learning anything (if not everything). This is the master algorithm of the title, and, as already noted, the Holy Grail of all learning. When achieved the true singularity will have been reached, that referring to the last machine humanity needs to invent, after that everything invents itself. The author surmises that a combination of the five approaches are called for in order to achieve the master learner, but of course exactly how they should be combined is so far less than half-baked, although that does not prevent him from presenting an extended reverie in terms of various metaphors making up some kind of surrealistic dream, which is hard, if not impossible to follow. But what can you expect from a popular book? Not any real explanation nor any solid instruction but some idea of what is going on?

The book is written in a breezy style which I find off-putting. There is, for natural reason, a lot of emphasis on the web where so much of the commercially motivated data mining takes place and where so many readers come in contact with it, readers who are assumed to be into Facebook, ordering videos on Netflix, books on Amazon, dipping into dating sites. The author extols the glorious benefits which will result from this revolution, how we all will be relieved of all the tedious chores in life and by implication be left only with the pleasurable aspects. The trouble is of course how one can make a distinction between pleasure and pain, after all they are intertwined. There are constant references to going to work in the future, but in what will that work consist in, when most of work as we know it nowadays will have been automated away? The author suggests that in the future we will talk about the employment rate, as that being the exceptional. But if people do not work, what will they do? Even science will be automated, and a large part of the book keeps referring to the cure of cancer, when automated learners will go through all the bio-chemical pathways of cells and design tailor-made medicines towards any kind of emerging cancer. What more can we hope for? So what do we do with all our free time free of worries to boot? He proposes that the humanities will have a renaissance, as they deal with what is essential and unique in the human experience. But how much of classical humanism will be relevant when the human experience has been radically changed? Human experience is after all about confronting reality and reacting to it in a way we have been evolved to do. Human life, as it already is now, is very different from what we have been evolved to lead. Of course we are eminently flexible, something we never tire of congratulating ourselves to, along with our intelligence, but how flexible are we really? Those are deep existential questions the author is not prepared (or interested in?) to ponder. As to the most pressing one, namely whether artificial intelligence will be taking over and the approach of the singularity (which to him is the advent of the master algorithm), he shows a cavalier attitude, just as a large part of the AI-community actually does. He dismisses the threat out of hand, without really having supplied any arguments against it, but being content with claiming that robots will never develop minds of their own. To which one may counter that evolution has created entities, such as us, which do have wills of their own, and if evolution is just an algorithm in principle it can be simulated in a virtual world (but that of course is not the same thing as to claim that it is feasible in any foreseeable future). One is disappointed in not finding more solidly argued rebuttals (apart from the sweeping statement that engineering is not evolution, especially as the former is set to adopt the latter as we will discuss below), but that would take another

book, and a more demanding one to boot, and which he probably would not be fit nor motivated to write. As to the claims by Kurzweil and his ilk, he simply replies that just as functions are not sustainably linear nor are they sustainably exponential and instead he champions the S-curve (which to the mathematically literate appear as an arctan curve) as the most important function in life. A slow beginning, a rapid development, only to level off in consolidation.

Engineers are not educated but very narrow and technically specialized. This is of course a prejudice, and as with most prejudices in addition to having a lot of exceptions, also have a lot of corroboration. Sprinkled in the text there are lot of things that grate on my reading nerves. As a sample. He refers to Darwin lamenting his lack of mathematical ability and comments that had he been born a century later he would instead have wished for programming prowess, as natural evolution is very hard to capture in equations but is instead elucidated by algorithms. Be that statement correct, nevertheless it shows an almost autistic incapacity to understand what mathematics (and programming) is all about. I cannot but dismiss it as unusually stupid, a stupidity ameliorated only by profound ignorance. There is further a mention of the so called Baldwin effect, when learned behavior can become genetically hard-wired without resource to Lamarckianism. The example of dog-like mammals learning to swim leading eventually to seals. The example is vacuous. Bayes formula is presented and actually an argument is provided for it, which I found confusing. To a mathematician it is a triviality, one simply expresses $P(A \cap B)$ in two different ways, in fact it is obviously $P(A)P(B|A)$[7] and the argument is symmetrical. But of course the significance of a mathematical fact lies not in how hard it is to prove, even if logically a tautology, as in this case, the proof of its worth is in the way it is used. Now when he claims that before the Monte Carlo method scientists were severely limited in their ability to compute integrals to those that could be done explicitly or low-dimensional ones which could be approximated by trapezoids. The latter refers to all computable integrals. Obviously the author has not given it serious thought but is only repeating hearsay. Malthus played a pivotal role starting Darwin's thinking, but not so much because of any parallelism between economy and life but for the simple fact that if unchecked, exponential growth would overflow the planet by each species. What kept it in check? Admittedly though, social thinkers of the 18th century did have a strong influence on natural science of the 19th century, as opposed to social thinkers of the 20th century on the 21st? That Bohr should have been inspired by the solar system in order to pose his atom model, and in particular that he would inferred by analogy that electron moves, is just plain stupidity and ignorance (which I fear is endemic among computer scientists). The solar system model is a posteriori and was meant as a metaphor for popularization and as with most metaphors just becoming silly when literally interpreted. I remember as a teenager being opposed to the one-dimensional representation of politics on a left to right scale and instead suggesting a 2-dimensional representation. This is actually corroborated here by suggesting two axi, one for economics and one for social values. This is now very fashionable. People on the left with a more confrontational attitude stick to the one-dimensional representation, while people on the right with a more manipulative agenda,

[7] In the space of outcomes, the probability of landing in X is by definition the measure of X and the conditional probability $P(Y|X)$ the measure of $Y \cap X$ normalized by a factor $1/P(X)$,

try to sell higher dimensional ones. Children who are more able to postpone gratification (as in waiting for a few minutes eating a marshmallow with the reward of a second one) will do better in school and adult life (all according to a famous study). This might be true, yet the statement, or rather the value imposed on it, induces a spontaneous, if not readily identifiable disgust.

But now to the core of the book. The five approaches to learning, what do they really entail? Although as noted, the purpose of the book is to explain them, this is not done very well. Let me make some attempt at interpretation.

First we have the symbolists. They believe that intelligence can be reduced to the manipulations of symbols. Their idea of a master algorithm is inverse deduction which figures out what knowledge is missing in order to make a deduction go through. What is inverse deduction? Let us return to it. What is presented here has so far no content.

The connectionists try to emulate the brain. The brain is about connecting neurons. Thus connectionists think in terns of neural networks, i.e. networks where knowledge is encoded in graphs, and the strategy is to build graphs, meaning how to create or to remove edges. In order to do so one need to be able to figure out what edges to change, when a test is given and the output is not the desired one. This works via back-propagation, whatever that means. This presentation has marginally more content, but what is meant by back-propagation?

The evolutionaries try to evolve programs the same way natural evolution has evolved organisms. Thus trying out variations of programs which more closely adhere to what we want, i.e. performing to our standards. In this way unforeseen combinations of subroutines will emerge. And a program actually defines a new structure, not just a given structure with parameters which we can tweet as in back-propagation. Now this has definitely more content than the previous presentations. We all have a feeling for what evolution means and this is enough to get us started. It is a good idea, and it seems to have had some success. The reference to back-propagation starts to give us some idea of what that might actually be (and more importantly its limitations).

The Bayesian take as their maxim that all knowledge is uncertain and hence that statistical methods are called for. Information is noisy erroneous and contradictory and as learners we need to handle this and not give up as logicians do once they encounter their first contradiction. The key thing is Bayes theorem and the strategy is to update probabilities in the light of new information. This has definitely content, in view of Bayes observation, still we do not know how to start from this.

Finally the analogists look for similarities between objects, and the idea is that if two objects share lot of feature it stands to reason that they share much more, and knowledge about one object will enable us to transfer that to another one and so on. It seems reasonable and part of the maxim that information can be reduced to simpler principles and play a crucial point in all scientific endeavor. However, it is hard to pinpoint what an analogy is although it is easy to recognize it. Engineers do not get bogged down in existential questions but take a more pragmatic instrumental view. Come up with some simplified notion of what is an analogy and see how well it works. Anyway this presentation is still too vague to be very useful so far.

Now the only method on which I so far can make any comments is the evolutionary.

Clearly evolution does not proceed by checking all possibilities (DNA strings in the case of natural evolution) but in a piecemeal way in which each step then depends on the environment. If you want to guess a 10 digit code you may have to go through ten billion combinations. But if you get feedback whether or not each digit is correct the search is so much shorter, down to a hundred. You try the first digit say, and find out that 7 is correct, given that you know that you can discard anything that does not start with seven, and go on for the second. Now in a computer setting you have definite goals what the program should accomplish, natural selection has no such explicit goal except for the implicit of local survival to reproduce. A common misunderstanding of evolution was that it has an ultimate goal, often seen as man, as the crowning result. In fact there is nothing in evolution that forces increased complexity and sophistication, mere survival to reproduce is all there is to it. However, there are striking success stories, and the existence of phenomena of convergent evolution is the most fascinating aspect of evolution giving some credence to the idea that it is inevitable that there is an urge so to speak to increased sophistication as an inevitable response to the environment. This has fueled the optimism of evolutionary programming, that it is bound to succeed. However, yet the great majority of living organisms are still prokaryotic and they do not seem to have evolved into eukaryotes repeatedly. One should also recall that in the metaphor of the fitness landscape, climbing along gradients only will lead you into local maxima and evolution is filled with clumsy solutions, which to Darwin was the strongest argument for evolution by natural selection. In short in the case of evolutionary programming you have the convergence problems.

Now let us look at the five approaches in more detail as presented by the author in as many chapters. First the symbolists.

Hume is presented as the finest English [sic] philosopher ever and I cannot but applaud and then as the foremost symbolist and here I only puzzle, what is meant by that? And what is meant by inverse deduction? Given a simple syllogism such as the classical one involving the mortality of Socrates. Say that we only know the specific assumption that Socrates is a man and the conclusion that he is mortal. What is the missing link? Clearly all men are mortal, and to claim that is to work by induction. Now this I would say is slightly misleading, it is true under the assumption that we have a correct deduction and this is the missing link; but generally we do not have this reassurance but are presented with a sequence of facts such as Socrates is mortal, Plato is mortal. Aristotle is mortal. We can generalize this in any number of ways. A timid one is 'All classical Greek philosophers are mortal' and a more daring one is 'All men are mortal'. The best strategy is to choose the most daring one the one most likely to be falsified and hold on to that until it becomes falsified (if ever). It is only when something is falsified that something happens to our hypotheses and they will be modified, thus strong daring assumptions get us much further than timid ones, because after all we only come up with the fruitful hypotheses when they have been repeatedly reformulated.

Now in the symbolist mode a hypothesis is a rule that tells us whether something is true and false, and a rule is being formed by looking at data and while being tested. When testing no longer seems to make much difference, the rule is accepted as a law, at least provisionally. This is all Popper and its says, as little as Popper insists on saying, on how

to generate the hypotheses. Clearly for this to be a feasible way we need an algorithm to generate hypotheses. Here is what the author proposes as the divide and conquer approach.

Concrete example: You have a certain number n of attributes, thus there will be 2^n such distinct objects (depending on whether they have it or not). A set of such objects is divided into bad and good. Can you find a rule that characterizes the good objects and the bad ones, i.e. a pattern in the form of a rule in terms of the attributes? There is a trivial solution, you simply list the good or bad examples. This is only possible if there are only a finite number of objects, and unfeasible if there are many of them. What you want is a shorter description. In general this will not be possible, but the idea is that our world is specific and structured and hence knowable. In practice we have a certain low number of sets, much smaller than 2^n, and the idea is that the rule we will come up with will in principle work for all the subsets. The author suggests the following algorithm. For each attribute consider those that have it or not and compare with the good and the bad. An attribute is relevant if there is a good fit. Having an attribute will exclude a certain number of bad ones as well as good ones, as will not having the attribute. The author now wants us to choose the attribute which excludes as few good ones as possible and includes as few bad ones as possible, and then continue, in the end we will have a rule that characterizes the good from the bad. This is nonsense as it is written down and will not work in general. Say there are g good ones and b bad ones, and having attribute i say there will be g_i good ones and b_i bad ones, but even if g_i i close to g does not mean that b_i is close to 0. There is no way in general to decide what is an optional attribute. What the author may have in mind is that a basic rule is of the form containing a subset A and a general rule or concept is formed by using a combination of ANDs and ORs. How many objects are there? 2^n and how many concepts 2^{2^n} i.e a double exponential. How to check them all? The notorious combinatorial explosion is about to happen. How to cut it down? By hoping that we only have to consider rules including very few basic rules. But anyway when only seeing a small sample of the whole thing there will be a profusion of rules that fit, so called over-fitting, and to rule them out rule by rule will be very time-consuming.

This ties in with the well known intelligence test to continue a series of numbers (intelligence tests seem to concern the ability to spot patterns compatible with those designed by the test makers). Given any series of numbers there is an infinity of various patterns that give rise to them. In particular there is an infinity of polynomials which can fit them. More specifically any $k+1$ numbers are the subsequent values of a polynomial of degree k. But there are of course many ways you can generate a series of numbers, such as taking the nth decimal place of $\sqrt{n}$ which is a fairly simple rule to state if not to compute. How would you come up with such a rule in the first place given the output?

Now why this is called symbolism is a puzzle to me. Maybe because it is the simplest and most direct approach to universal learning and the natural route initially taken. But it is being criticized for being too rigid and too categorical, either things are in or not, and it cannot easily handle continuous data or uncertainties. But of course that could be gotten around. I would not be surprised if any of the five methods could be simulated on any other, but whether feasible or not is another question. The main weakness is how to come up with restraints on the rules unless we a priori restrict the structure of the rules, and then we come back to the initial approach of encoding knowledge. Thus if we want to

instruct a universal learner we need to have precise knowledge, i.e. encodable in computer language, of what goes on in learning. The whole point is to generate a universal learner without knowing or understanding how it actually works[8].

Brains are not computers they consist of neurons and connections. A huge amount of neurons, but an even huger amounts of connections. Neurons fire depending or not on the firing of neurons inputting into them. If enough of them do it will. Now this can be tweeted a little. Different inputting neurons may have different weights and firing can be of more or less probability. Anyway we get a lot of structure into the whole thing and if there is enough structure there will be room for storing and processing information. What we have is the neural network and a key concept is what is called back-propagation. When outputs do not match what is desired (what is really meant by that is not explained) one may change the parameters i.e. the weights of the inputting neurons to the sinning one in the step-wise gradient way known to us all. Those neurons in their turn may have the weights of their incoming neurons changed, and so on, hence the name back-propagation. In the end there will be stable configurations, corresponding to local maxima or minima in some hyperspace landscape. The obvious criticism is that only uninteresting local extremes may be found, but the proof is in the pudding and the method have had some notable successes as in spotting and riding trends in the stock market, the value of which apart from individual gains, seems opaque to me. A key thing to note is that information has nothing to do with individual neurons but to combinations thereof, and a neuron may of course be part of may different configurations. This may excuse a short digression. Are there more neural connections in a human brain than there are atoms in the Sun? Obviously not, even if for most people it might take more than a flash of thought to realize the absurdity of the proposition. But is the human brain more complex than the Sun, i.e. more structured? The Sun does not think in spite of all the abundance of material that serves as a basis for such activity. But if we look at connected subsets of neurons, each potentially a thought so to speak, their numbers easily exceed by combinatorial necessity any number of atoms in the Sun. Of course the conscious mind cannot think them all, but they are there to be thought? Or take a hundred marble balls. They fit easily into a small box. But then consider all the subsets of those marble balls, they too fit into the same box, but they are so many (2^{100}) but the spaces they take are not exclusive, each marble ball will be part of many (2^{99}) subsets, and that many boxes are needed to house them all if all the subsets should be physically separated. One may wonder whether a subset really is something merely in the mind or a physical thing. Once one is thought of it can be pointed out and physically pointed at, but did it exist at all before? This is a silly question maybe but at the root of Platonism.

Anyway in spite of the explanation of the technical terms it is not clear really why neural networks could work and if so why they work, let alone why they should be superior to the symbolist approach. How do they encode patterns and how do they process data. The examples given are just too simple to provide any clues. In so much of the book it all becomes just hearsay.

Genetic programming is the easiest to understand and grasp for the simple reason

[8] Which we already stated in the beginning, showing the shortcomings of a so called knowledge approach

that the principles of natural selections are known to us almost as far back as childhood. In neural networks a single hypothesis is entertained and being gradually perfected by so called back-propagation. While in genetic programming there is a multitude of hypotheses, each corresponding to a pattern, or a rule , which are being recombined and those most closely adhering to some predetermined fitness function, (in real natural selection there is no predetermined fitness, it emerges and changes with time, corresponding to the fact that it has no goal unlike an engineered process) are more likely to continue in the process. We have the same problem as with back-propagation that undesirable local maxima (as to fitness) maybe reached, but that can to some extent at least, be counteracted by some random movement. As the terms goes, exploration versus exploitation. If there is too much exploration we just have a random walk in a huge landscape which will lead us nowhere, exploitation is to take advantage of opportunities that arise, but we should be leery of becoming obsessed with them. The whole point of the game is to evolve structures that we as humans might never have been able to think up, and not by a random process that will only produce noise, but by a structured one. Genetic programming is thus creative in a way that the connectionists are not. I the latter case we provide rigid structures with parameters that can be changed in a similar way step-wise way, but the imposed structure sets limits. As is well known continuous functions on a compact interval can be uniformly approximated by polynomials, provided that the degrees can be arbitrarily high. Starting with a given function and polynomial we can then design a procedure that changes the coefficients step-wise to get a better and better approximation. With genetic programming we may design other approximating functions which might do the job faster. However, in any designing of a genetic programming approach there will always be some hidden constraints to the creativity depending on how we set about the structure of the program. Those constraints will only emerge in running the programs repeatedly, and as noted natural selection does not always work, in fact it seldom works, the in retrospect glorious exceptions bias our perceptions.

Statisticians swear by Bayes although Bayes did not do much it was really Laplace who turned it into a method. It boils down to wanting to know the probability of the causes given the effects, while knowing the probability of the effects given the causes. As a graduate student I had an American mother-in-law who was blue-eyed, her husband had brown eyes, what was the probability that he had a blue gene given that their four children were all brown eyed? If he had had none then the probability of four children with brown eyes would have been one, but if he had had one, the corresponding effect would have had only a probability of 1/16, for some reason I decided that there was a probability of 16/17 that he did not have a blue gene and I felt very clever. But what sense is made of this? I did not think further at all (perhaps because I held probability in contempt?), but now I will, wiser (and humbler) as we get with age. If we would know that in the population out of which the father was drawn there was only one brown gene we would know that the other one was blue, and that the probability hence was one. Thus if we would know that the blue gene was very common, that would influence the conditional probability towards one, hence way from the suggested 1/17. If we would test thousands of brown eyed fathers having (four) children with blue-eyed mothers and keep track of how many of those would have blue eyes would we then be able to instead get an estimate of the prevalence of blue

genes in the gene pool of the fathers by the same kind of reasoning? But to make sense of that we would have to make some initial assumptions of the probability distribution of *a priori* probabilities for blue genes, which we no doubt would implicitly assume to be uniform (just as I may have subconsciously assumed that blue genes were as common as brown ones). What we get is an infinite regress, where we always have to modify an initial probability assumption in the light of new evidence, which is of course in the popperian spirit of doing science. This actually leads to an existential question of what is probability? Does it have an objective existence or is it just in our minds? The axiomatic approach by Kolmogorov sets an a priori probability by fiat, thus nothing to discuss. Given that you can derive all kinds of probabilities from derived models and that is of course a purely mathematical exercise. If coins are fair and subsequent throws independent you easily set up a space of outcomes with the 'right' probability, but of course it depends on two axiomatic assumptions. Are axioms true? You cannot prove them unless setting up new axioms to which the same questions will apply, what you can do is to derive consequences and reject them if the consequences will not agree with you, but that of course will never prove truthness in a finite process no matter how long. In the same vein we can ask to what extent the statistical models we provide correspond to reality, whether the probability of a given coin is indeed a half, or very close to it. The frequentists claim that there is an objective probability and it reduces to a frequency count. By Bernoulli the frequency of toss coins will converge to the assumed probability which gives them an anchor on reality, of course any finite throws leave a margin of doubt, be it a diminishing one. And of course a specific coin will physically change with each throw leaving it in a slightly different physical shape no longer being really the same as before. To the Bayesian probability is something we have in our mind and which we will have to continually modify in the light of new experience. Any computation of probability will rest on some more or less hidden assumptions, which we may have to revise as we go along.

Now this can be applied to neural networks or to give probabilities to different hypotheses (for say some pattern) which will change or rather evolve as we encounter more and more data, never in principle ruling out any hypothesis as unlikely (although of course for practical reasons that has to be done). In a Bayesian world nothing is certain, but so it is not in the real world either. The advantage is that true hypotheses are not ruled out prematurely.

Analogies are hard to define in any precise way, and just as with the case of ideas, any attempts of precise definitions, threatens to cut out essential features of the concept. Ideas, of which analogies form an important aspect, are also intimately related to will, namely an idea is not just an idea, a living idea also wills us to try and implement it. If will is beyond the algorithm so should also idea. There is explicit thought which can be articulated and codified, and the metaphysics of the context, which is implicit.

So what is the idea of analogy in this context of learning? Or rather the idea of implementing it. Things that are alike as to some attributes are also alike in others. This means that to codifying things we need less attributes than we think, a comparatively few are enough to imply more or less the rest. Of course the way of expressing this idea does not determine the way we can implement it, but just suggest things to us, which only become apparent when we decide to do them.

How can we distinguish human faces among pictures? We have a space of pictures in terms of their pixel representation. To each certified human picture in the data space we draw the Voronoi polyhedron (although the author does not call it so) which consists of all the points in the space, corresponding to potential pictures, which are closer to the given picture than to any other picture in the data space. Human pictures will then become the union of all those polyhedrons. It does of course depend on what kind of metric we imposes, the devil is always in the details, but if the data space is big enough maybe you do not need to be very sophisticated? This is the governing idea of big data, the bigger the data is, the less sophisticated you need to be. It is better to process as much data as possible than to spend an excessive amount of time on each. Empiricism taking precedence over rationality. The plodder wins at the end, and after all plodding is what computers are good at.

But of course a model is not fun if you cannot tinker with it. Tinkering is what modeling is all about, while when it comes to theory building, the aristocracy of models according to Manin, tinkering is plebeian, and is not to be done frivolously. No one thought of tinkering with Newton's beautiful formula for gravitation (neither did Einstein, he came to his alternatives through another route). Tinkering lies at the heart of the Ptolemy system, which after all was never intended as an explanation of the movements but as a model to save the appearances. And this attitude survives to this day, most of science is on the level of Ptolemy. The nearest neighbor could be replaced by the k-nearest neighbors, so if the closest image is a face, but the others are not, one may decide on a vote and not classify it as a face after all. Then this can be refined by adding weights to the different neighbors depending on how close they are. And so on. You can never tell on first principles if anything will work, you just run things and see what happens. But of course this approach as any other approach runs the risk of combinatorial explosion and all kinds of tricks need to be considered to avoid it, inevitably reducing the flexibility and power of the learning process in order to make it feasible.

The final discussion on how to combine the five approaches to the true master algorithm - the Holy Grail - and hence singularity, is too speculative and confusing to be of any interest to retell. Finally one may ask what are the real benefits? The applications seen so far are mostly on the net when various businesses want to get as much information on its potential customers as possible just to target ads and increase their chances of catching them as customers. The ultimate goal seems to me depressing and the actual performances in selecting the items of ostensible interest to me risible and transparently stupid (but as they say, this is just the beginning, the baby state of ineptness, when it grows up, it will be otherwise). As to predicting stock markets and acting accordingly what good does it do globally after all? Things like this tend to be self-fulfilling, and the industries and corporations which are supported, are so not on the basis of their beneficial aspects as their potential for profit for the investor. The market, which is of course very old, is seen as a huge learning machine, in which only the best will survive. But 'the best' in what sense? Similarly with the applications to polling, identifying key voters. The candidates which profit do so not by virtue of their policies but their petty tactics. Most of us are grateful that Obama had the edge on Romney as to big voters data, but it could as easily been the other way around (as with Trump versus Clinton, the supposed Russian

intervention could very well be another red herring). And finally as to science. Will those learning algorithms really allow us to cure cancer? Is it just a matter of simulating the biochemistry of cells and modify known toxics to them? Tedious work that is beyond human 'manual' capacity that can be automated. What is left for the scientist to do when science has taken over by computers? Will he just be some kind of administrator? Or will the algorithms necessitate continuous supervision after all, and the scientist is reduced to a tinker? Assume that Newtonian mechanics had not seen the light of the day, would then the movements of the celestial bodies be predicted by big data? An ever refining addition of epicycles in each case? The ancient Babylonians were able to predict quite a lot of things by virtue of their extensive data. Just think of how different matter is by Newtonian 'understanding' or rather simple principles (no one understands the law of gravitational attraction apart from Einstein's geometric explanation of it). Is the medical establishment actually engaged in a Ptolomean project, given the emphasis on cures and correlations to support them. Or will computers free scientists to delve deeper into issues of understanding, but maybe we will come up to the limits of human understanding and can only watch in amazement, soon turning into boredom, what is being given to us? The tenor of the scenarios of technological development of the beginning of the 21st century is very different from the refreshing optimism that characterized them at the end of the 19th.

March 30 - April 1, 2018' **Ulf Persson:** *Prof.em, Chalmers U.of Tech., Göteborg Sweden* ulfp@chalmers.se