The Five Tribes of Machine-Learning: A Brief Overview

Conference Paper · July 2019

CITATIONS
0

READS
299

1 author:

Jens Pohl
Tapestry Solutions (a Boeing company)
114 PUBLICATIONS **636** CITATIONS

SEE PROFILE

The Five Tribes of Machine-Learning: *A Brief Overview*

Jens Pohl, PhD
Professor of Architecture (Emeritus)
California Polytechnic State University (Cal Poly)
Vice President (CTO), Engineering & Technology
Tapestry Solutions (a Boeing Company)
San Luis Obispo, California, USA

Abstract

This paper reviews recent advances in automated computer-based learning capabilities. It briefly describes and examines the strengths and weaknesses of the five principal algorithmic approaches to machine-learning, namely: connectionism; evolutionism; Bayesianism; analogism; and, symbolism. While each of these approaches can demonstrate some degree of learning, a learning capability that is comparable with human learning is still in its infancy and will likely require the combination of multiple algorithmic approaches. However, the current state reached in machine-learning suggests that Artificial General Intelligence and even Artificial Superintelligence may indeed be eventually feasible.

Keywords

algorithms, analogism, artificial intelligence, Bayesianism, connectionism, evolutionism, generalization, genetic algorithms, induction, machine-learning, neural networks, overfitting, rules, symbolism

How can a machine learn?

There are problems that are difficult and sometimes impossible to solve by writing a traditional computer program in which we instruct the computer step-by-step how to solve the problem. The reason is that either it would be too complicated to embed in the program all of the conditions that must be considered in arriving at a solution or we simply do not know how to solve the problem. The latter category includes the interpretation of handwriting. For example, the hurriedly written number zero shown in Figure 1 in which the loop is not fully closed at the top could easily be interpreted as the number "6".

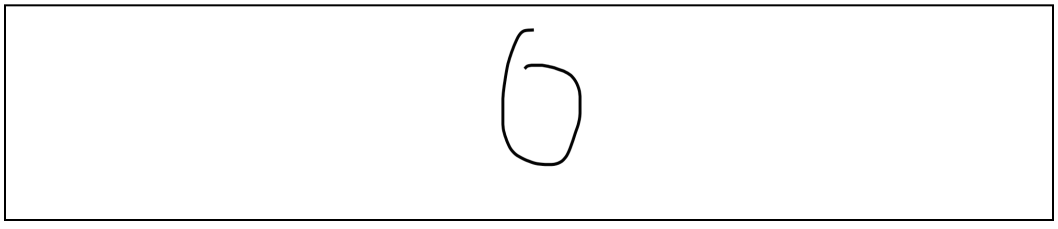

Figure 1: Handwritten "0" that is difficult to distinguish from a "6"

It would be very difficult to construct a set of rules that can solve this problem heuristically. There are many types of problems such as recognizing objects in images and comprehending speech that fall into this category. We cannot solve these kinds of problems with a traditional computer program because we do not know exactly how our own brain is able to solve such

problems successfully in most cases.

The approach that machine-learning relies on to solve such problems is very different from the traditional computer programming approach. We devise an algorithm that the computer can use to look at thousands of examples and learn to recognise the correct solution to the problem. In other words, the computer is learning by analyzing example cases. However, as intuitively simple this approach may appear, there is a potential problem. Even if the computer has analyzed thousnds of example cases that all validate the conclusion reached by the algorithm we cannot be sure that the conclusion holds true universally for other examples that are external to the corpus of data currently available to the computer.

How can we ever be justified in generalizing from what we have observed to what we have not observed? Is there any way to learn something from the past that we can be confident will hold true for the future? This is a fundamental problem for machine-learning. Analysis of a corpus of data may result in the appearance of one or more patterns. These apparent patterns may be false due to the features selected for the analysis not being sufficiently decisive (e.g., there being no single or small set of predictive factors)[1], or the corpus of data that has been analyzed not being large enough, or one or more primary features having been considered of minor importance or altogether overlooked.

The generalization dilemma

Philosophers have debated the induction dilemma ever since it was first posed by Hume in 1739 without coming to a definitive conclusion. It essentially means that we can never be certain that some conclusion that has held for all instances until now will continue to hold for all instances in the future. At best, we can possibly say that it has a very high probability of holding for all instances in the future. The questions then arise: How important is absolute certainty and how critical are any potential exceptions? The answer to each of these questions depends on the individual circumstances. When Google responds to a search query it looks in its massive logs for the same or similar search queries that have been entered by users in the past and assumes that the links that these users have clicked on in the results pages are equally applicable to this particular search query. The penalty for this assumption being incorrect is not severe. At most a second try by the user with a slightly different query and an acknowledgement that the Google search algorithms require further fine-tuning. On the other hand, in a medical diagnosis where the wellbeing or even the very life of the patient is at stake, a wrong assumption may have dire consequences. In this case, the circumstances are exacerbated by the fact that no two human beings are exactly alike and no set of symptoms are a complete match for two patients.

Generalizing on the basis of existing data to future instances (i.e., cases that we have not seen before) is the fundamental machine-learning problem (Domingos 2015, 59)[2]. Memorization on its own cannot solve this problem, because the number of combinations is simply too large. For example, if we have a corpus of data with just a million records and each record with just 100 Boolean fields with a yes/no question, then how many possible cases are we likely to have seen. For 100 questions with yes/no answers the total number of cases is 2^{100}, which is equal to 1.2676506×10^{30} and more than a hundred thousand times as large as our original corpus of one

[1] In other words, there is no basis to select one generalization over another.

[2] Frequent reference is made in this paper to Pedro Domingos' comprehensive review of the principal approaches to machine-learning in his book entitled "The Master Algorithm: How the Quest for the Ultimate Learning Machine will Remake the World". This excellent book is highly recommended for further reading.

million records. When we scale this to a more likely database of a billion or a trillion records the available data becomes a miniscule percentage of the number of cases. Accordingly, the probability that a new case on which a decision has to be made is already in the database is so miniscule small that we are forced to resort to generalization if we want to make a decision about this new case.

The appropriate conclusion is that data alone is not sufficient for successful machine-learning. What is also needed is knowledge, so that the theoretically determined enormous number of possible cases is reduced to a much smaller set of relevant cases by incorporating context in the machine-learning algorithm. This of course also introduces a degree of bias into the machine-learning algorithm. Although we typically consider bias leading to preconceived notions as undesirable in our daily activities, such preconceived notions are a very necessary component of machine-learning (Mitchell 1997, 39; Domingos 2015, 64). In the case of human learning both our genes and the external interventions of our environment, particularly during our early formative years, embed preconceived notions and beliefs in our brain. These embedded notions, beliefs and biases are an essential part of our learning process. Similarly, the objective in machine-learning is to include just sufficient knowledge in our program so that the learner can continue to acquire knowledge ad infinitum by reading and synthesizing data. This is a gamble because machine-learning will not always make the correct decision. In fact, it is likely to make many mistakes. However, we can deal with this inaccuracy by discarding the misses and building on the hits. The cumulative result is what is important to us and that has been shown to be generally reliable.

How much seed knowledge does machine-learning need to be cumulatively successful? The nature of the knowledge that we embed in the learning algorithm appears to be quite primitive, but it is in fact very powerful. It starts with the selection of the features of the data that we consider to be relevant. This is indicative of the gamble that we face. If we chose the wrong features then the learning algorithm is unlikely to produce any useful results, or worse, the results may be misleading. Then we apply initial weights to these features on the understanding that these weights will be progressively automatically adjusted by the algorithm as it reads and synthesizes more and more data. In this respect, mechanisms that allow the learning algorithm to change features and adjust weights as it gains more knowledge are of critical importance.

The inductive process in machine-learning

In the treaties *Principia* (Cohen and Whitman 1999) Newton enunciated his well-known three laws of motion as well as four much lesser known rules of induction. The third of his induction rules states: *Whatever is true of everything we have seen is true of everything in the universe.* While this rule represents an enormous leap to generalization, it is also at the very heart of machine-learning. We start the automated inductive learning process by applying the most widely applicable rules and then proceed to reduce the scope of these rules when the data forces us to do so. For example, we might initially assume that the rate of car accidents is mostly governed by the speed at which a car is traveling. When the application of this rule to the data produces contradictory results we are forced to reduce the scope of the rule by adding a second rule; namely, that the age of the driver is also a factor. If the conjunction of these two rules fails as well then we add a third rule, such as male drivers are more likely to have accidents, and so on. Until we have a set of *conjunctive concepts*[3] that correlate with the data and allow us to fairly

[3] Dictionary definitions are typically *conjunctive concepts* (e.g., a chair has a seat, a back, and one or more legs).

accurately predict the rate of car accidents. The larger the set of conjunctions, the greater the number of possible combinations of features that have to be analyzed, requiring an ever increasing amount of computation. It now becomes clear that machine-learning requires huge volumes of data to reduce the risk of false results and an enormous amount of computation due to many plausible conjunctions.

Starting with broad (i.e., restrictive) assumptions and gradually relaxing them if they fail to explain the data is the typical machine-learning approach. This process is normally carried out automatically by the algorithm. First, it uses all single factors, then conjunctions of two factors, then conjunctions of three factors, and so on. The limit to this process is time and computational power (Domingos 2015, 66-68). However, even with the enormous computational power that has become available over the past two decades this approach is likely to be too time consuming. To overcome this problem we could start off by assuming that all matches are good and then prune the large number of matches by deleting all matches that do not include a particular feature (i.e., attribute). If we repeat this for each feature and in this way identify those features that delete the least good matches and the most bad matches, then we are likely to be left with a set of conjunctive concepts that have been pre-qualified.

The rule-based approach

The problem with the conjunctive concepts approach is that it allows for neither exceptions nor alternative ways of achieving a goal. For example, birds fly unless they are ostriches, kiwis, penguins, or are housed in a cage. A chess game can be won in many different ways. In the early 1980s Michalski (1980, 1983, 1986) showed that the conjunctive concepts methodology could be used to generate a set of rules. After the system learns a rule all positive examples that it accounts for are removed from the data. Then the next rule is applied to the remaining data and all cases that it can account for are discarded, and so on until there are no remaining cases. For example, applying this to the previously discussed corpus of traffic accident data, we might apply the following set of rules: you are more likely to have a car accident *if you are under 25 years of age; if you have been drinking alcohol; if you are using your mobile phone; if you are driving in peak-hour traffic*. While rules are more powerful than conjunctive concepts, they are also more dangerous. Let us assume that a rule set covers the exact positive cases in a corpus of data. Then it will predict that every new case that is not exactly like at least one of the positive cases is negative. In other words, the ability of computers to remember every detail of every instance is not especially useful for machine-learning. For example, while a computer has no difficulty to remember every detail of every e-mail spam message, a rule that suggests that an e-mail is spam only if it is exactly like a previous spam message would be useless.

The *overfitting* problem

Whenever machine-learning finds a pattern in the data that is not actually true in the context of our real world then we refer to that as *overfitting* the data. Unfortunately, learning algorithms are particularly vulnerable to finding patterns in data that do not exist in the real world. The reason is that the computer has the ability to process vast amounts of data and therefore an unlimited capacity to identify potential patterns. For example, *The Bible Code* published in 1989 (Drosnin 1997) became a bestseller due to its claim that the Bible contains potential predictions of future events when one constructs new words by skipping letters at regular intervals. It turns out that there are so many ways of doing this that one is bound to construct something useful as long as the text is long enough. The claim was easily countered by demonstrating the same phenomenon

in transcripts of U.S. Supreme Court rulings.

As Silver (2012) points out, *overfitting* is greatly exacerbated by noise in data, which equates to errors and random occurrences in machine-learning that can lead to false hypotheses. However, even without considering noise in data, for a learning algorithm based on conjunctive principles the number of possible instances of a conjunctive concept is an exponential function of the number of features (attributes). In the case of Boolean attributes, each new attribute doubles the number of possible instances by virtue of its required yes/no value. As an analogy, if we take the 64 squares on a chessboard and place two grains of wheat on the first square, four grains on the second square, and so on doubling the number of grains for each square, we end up with 2^{64} or over one trillion tons of wheat on the 64^{th} square alone[4].

Domingos (2015, 74-75) makes the case that machine-learning is a trade-off between the number of hypotheses considered and the amount of data available to test each hypothesis. While more data can exponentially reduce the number of positive hypotheses and increase the probability that a hypothesis is correct, there is no absolute certainty that the hypothesis will hold for all new data. The question then arises: When can we believe that a hypothesis that appears to have been validated by a learning algorithm is in fact correct? The core response is that it has to be verified with new data. It is not sufficient for a new theory to explain past data. The theory must also hold true for new data. To accomplish this verification in machine-learning it is common practice to divide the available data arbitrarily into two sections; - namely training data and validation data. The training data is used to create the hypothesis and the validation data is used to test it against new data.

While testing the learning algorithm on new data is of critical importance in machine-learning, even then we cannot be sure that the possibility of overfitting has been eliminated. Apparently, in an early military application, a relatively simple learning algorithm was able to detect with almost 100% accuracy in both the training and testing data the presence of tanks in photographic images. It was later discovered that all of the images containing tanks were lighter than the other images and that was all that the algorithm had learned to base its recognition of tanks on.

Machine-learning algorithms

Machine-learning algorithms essentially program themselves by drawing inferences from large volumes of data. Their application is pervasive in our daily activities. Our alarm clock wakes us up in the morning with a tune that we like and that has been selected by an algorithm that has learned our musical taste. The temperature in our house is comfortable and yet our electricity bill is lower after we installed a smart thermostat that has learned how to conserve energy. We drive to work guided by a navigation system that selected the best route based on typical traffic patterns applied to current conditions. Our car is able to optimize fuel consumption by continually adjusting the fuel injection system based at least partly on our observed driving habits. When we arrive in the office, our e-mail has been conveniently sorted into categories based on our preferences. There is a message in a foreign language that can be immediately translated into English by Google's translator. As we prepare a report the word processing system automatically checks our spelling and grammar. When we make travel reservations our Travel App suggests that we delay purchasing the selected airline ticket because the price is

[4] 2^{64} is equal to 1.8447×10^{19}. If there are 7,000 grains in one pound then there are 15,680,000 grains in one ton (i.e., 2,240 lb) and 1.176×10^{12} tons in 2^{64} grains of wheat. The weight of wheat required to fill the entire chess board is therefore far beyond the world's annual wheat production of approximately 65 million tons.

likely to reduce within the next 24 hours.

Machine-learning is fast becoming a foundational methodology in the field of Artificial Intelligence. It already plays a role in every part of our lives, as it analyzes and learns from the keystrokes on our computer, the calls we make and receive on our mobile phone, what we buy and how frequently in our local supermarket, the approval of our credit card when we make a purchase, and so on. The rapid rise of machine-learning is due to its ability to automatically extract information and knowledge from any large corpus of data; - it learns by finding and testing hypothetical relationships and dependencies.

While there are many new learning algorithms proposed each year, there are really only five principal approaches (Domingos 2015, xvii): *connectionist algorithms* endeavor to simulate the way the human brain works; *evolutionary algorithms* draw on genetics and evolutionary biology; *Bayesian algorithms* rely on statistical probabilities; *analogy algorithms* rely on similarity judgments; and, *symbolistic algorithms* use inverse deduction. These five approaches are commonly referred to in the literature as the five tribes of machine-learning.

Connectionist Approach: In the human brain the recognition of any image starts with the activation of neurons that respond to very low-level properties of the image such as the luminance of one or more distinct areas or dots or lines or planes in the image. The firing of these neurons will activate other connected neurons that respond to other features that build on the partial interpretations of the previous level neurons, and so on. This is referred to as a synaptic chain in which the chemical connections (i.e., synapses) between neurons play a decisive role. With each successive level of response the comprehension of the whole image becomes more complete and the contribution of the next level builds more and more on the cumulative results of the previous levels.

In the connectionist approach to machine-learning these synaptic chain operations of the brain are simulated mathematically not only by multiple layers of neurodes (i.e., simulated neurons) in each neural network but also by stacking multiple neural networks vertically with mathematically computed connections between them. Like a neuron, a neurode receives inputs from its connection to other neurodes within the network and multiplies these inputs (X) by a weighting factor (W) to compute a function (F). If this function exceeds a cumulative value then the neurodes will *fire* by sending an output to all other neurodes that are connected to it (Figure 2).

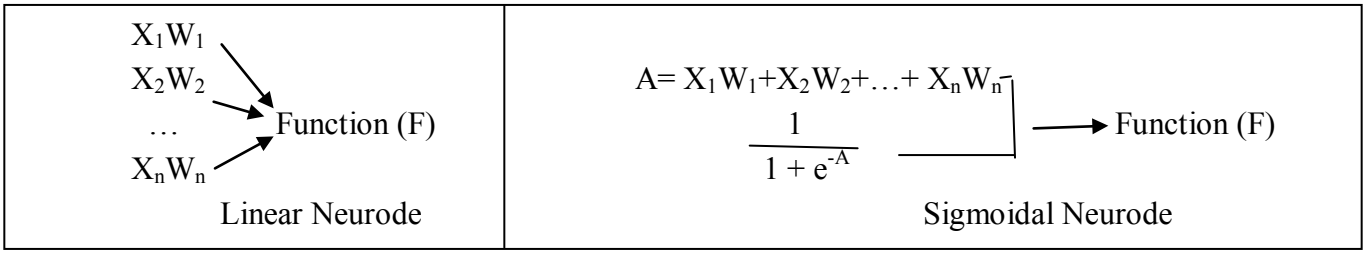

Figure 2: Linear and sigmoidal neurode *firing* functions

When the total input to the function (F) is very negative its value is close to zero. If the input is very large and positive then the value of the function is close to one. The neurodes are connected in one or more (hidden) layers between the input layer and the output layer.

Backpropagation is the connectionist's principal machine-learning algorithm. It was invented

by David Rumelhart in 1986 (Rumelhart et al. 1986). After the neural network has processed the input to the output during its forward pass, the output is compared with the desired output and the error is propagated back through the layers of the network and the input. Each neurode adjusts its weighting based on the error and the input that it received during the forward pass. After thousands of repetitions the neural network's output neurodes will closely match the desired output that recognizes the particular input that it was trained to recognize. During this *gradient descent* operation there is no certainty that the neural network has reached the best (i.e., optimum) output condition even after thousands of iterations; - i.e., it may have reached a *local minimum* of the error. However, experience has shown that this is not as serious a drawback of backpropagation as originally thought. First, a local minimum may be quite acceptable since in most cases the error plane resembles a quilt with many peaks and troughs. Second, the local minimum is less likely to have *overfitted* the data than if we were to insist on reaching the global minimum.

Deep Learning utilizes interconnected neural networks to simulate the ability of the human brain to comprehend a visual sensory input (i.e., a scene in the local real world environment). Several neural networks are stacked vertically to conceptually simulate synaptic chain operations in our brain. During training the neural networks are shown a large number of positive (i.e., correct) examples and the weights between neurodes are incrementally adjusted to produce a positive output (i.e., value of function (F) close to one).

Evolutionist Approach: This approach is based on Darwin's theory of natural selection, using genetic algorithms. A genetic algorithm replaces the selective breeding process of plants and animals with a learning algorithm and reduces the time between generations to a few seconds of computer time. Holland (1992) recognized the importance of incorporating the mechanism of sexual reproduction in the learning algorithm. In reproduction two new chromosomes are produced; one consisting of the mother's chromosome before and the father's after reproduction and the other consisting of the father's chromosome before and the mother's after reproduction (Figure 3).

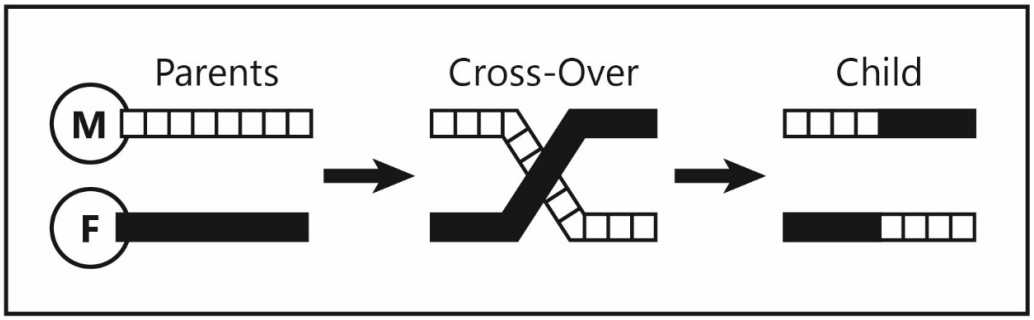

Figure 3: Sexual reproduction (Domingos 2015, 124)

Based on a fitness function the genetic algorithm creates variations that can be evaluated according to meeting the fitness goal. Similar to the way that in nature DNA encodes an organism as a sequence of chromosome pairs, the genetic algorithm uses a string of bits. During each generation the algorithm creates the fittest cases in the data by crossing over their bit strings at a random point in time. In this virtual computational world each mutated case receives a fitness score, with the result that each computed generation is fitter than the previous one. The process terminates when either the desired fitness level has been reached

or time runs out.

Koza took Holland's work a major step forward by replacing the bit strings in genetic algorithms with software code, arguing that a computer program is really a tree of subroutines (Koza 1992). Using crossover (Figure 4) and mutation to swap subroutines between program trees he was able to show that genetic programs can include a wide range of programming constructs such as if-then comparisons, loops, and recursion.

Genetic programming's first major success in the mid-1990s was in the design of electronic circuits (Koza et al. 1999). Koza was able to reinvent a previously patented design for a low-pass filter that could be used to selectively enhance a particular frequency band in a musical recording (Carnett and Heinz 2006). In 2005, the US Patent Office awarded a patent to a factory optimization system that was designed using genetic programming.

Bayesian Approach: At the core of Bayes' theorem is the simple concept of starting with a hypothesis in the form of an estimated probability that the hypothesis is correct and then adjusting the probability as new evidence (i.e., data) becomes available. For example, if we flip a true coin a sufficiently large number of times the number of heads will be approximately equal to the number of tails. Therefore, if our hypothesis is that the coin is true then our initial estimated probability of the flipped coin coming down heads would be 50%. As we continue to flip the coin each heads will increase and each tails will decrease the probability of the coin being true. The initial probability, which is essentially a subjective belief, is referred to as the *prior probability* and the adjusted probability based on new data is referred to as the *posterior probability*. The *posterior probability* is determined with Bayes' theorem as follows, where P stands for probability:

$$P(\text{cause} \mid \text{effect}) = P(\text{cause}) \times P(\text{effect} \mid \text{cause}) / P(\text{effect})$$

Domingos (2015, 147) provides the following medical example where influenza is the cause and fever is the effect. If out of 100 patients, 14 had influenza, 20 had a fever that was not necessarily associated with influenza, and 11 had both influenza and a fever. Therefore:

$$\begin{aligned} P(\text{cause-influenza}) &= 14 / 100 \\ P(\text{effect-fever}) \mid \text{cause-influenza}) &= 11 / 14 \\ P(\text{effect-fever}) &= 20 / 100 \\ P(\text{cause-influenza} \mid \text{effect-fever}) &= (14/100) \times (11/14) / (20/100)^{5} \\ \textit{posterior probability} &= 0.55 \text{ or } 55\% \end{aligned}$$

We usually know the probability of the effects given the cause. For example, the probability of a fever if the patient has influenza. However, what we would like to know is the probability of the cause given the effect, such as the probability that the patient has influenza if the patient has a fever. If in the above example the physician had started with an intuitive estimate of the *prior probability* of 70% that a patient with a fever has influenza, then this probability would now be reduced to 55%. Additional data would either increase or decrease the 55% that has now become the *prior probability*.

In reality, the application of Bayes' theorem to this machine-learning example would require multiple effects such as sore throat, fatigue, prevalence of influenza in proximity of the patient, and so on, to be taken into account. The computational burden increases exponentially. If there are *n* effects and each carries the Boolean value of *yes* or *no*, then

[5] Posterior probability = P(cause | effect) = (14/100) x (11/14) / (20/100) = 0.14 x 0.7857 / 0.2 = 0.5499 or 55%

there are 2^n combinations that have to be calculated for each data set. In the case of only 10,000 data sets and only ten effects there are already over 10 million calculations (10000 x 2^{10} = 10000 x 1024 = 10,024,000).

However, we are faced with two further problems. First, there are dependencies that require combinations of symptoms to be considered. A patient who has a fever and also a sore throat is more likely to have influenza than a patient with only one of those symptoms. Second, we need an enormous amount of data to have some confidence that the available data covers the combinatorial range of different cases and that there are a sufficient number of instances of each case[6]. To deal with the combinatorial explosion problem we are forced to make concessions such as, we assume that the effects are independent of each other given the cause. This is referred to as the Naïve Bayes Classifier. Naïve Bayes is widely applied to e-mail spam filters, search engines, text classifiers, and in many other domains. In fact, at this time (2019) it may still be the most widely used machine-learning algorithm (Russell and Norvig 2012; Domingos 2015, 152).

Analogist Approach: Global connectivity has resulted in a deluge of data. The emergence of the Internet of Things (IoT), where almost everything we use on a daily basis will be tracked, promises to increase the amount of data by orders of magnitude. However, the data is often unevenly distributed so that we may have more data than we can deal with in most domains and at the same time insufficient data in some smaller subdomains. The connectionist, evolutionist and Bayesian approaches to machine-learning all construct an explicit model of the phenomenon under consideration and therefore depend on the availability of ample data. They cannot learn if there are serious data gaps. This is where analogizers come to the rescue, because they do not construct a model and can therefore learn from as little as one example (Domingos 2015, 175).

The analogist approach learns by finding an example that is sufficiently similar to the case in the data under consideration. At the most primitive level it uses the *nearest-neighbor* algorithm[7]. For example, when a Facebook user uploads a photograph of a person how does Facebook recognize the image as being the face of a person? It looks through its database of images to find another image that contains enough similar features to make it highly probable that it is of the same kind. If this image is known by Facebook to be the face of a person then the just uploaded photograph is also classified as containing the face of a person.

Domingos (2015, 180-2) provides an excellent example of how the apparently trivial capabilities of a *lazy learner* can be applied in a surprisingly sophisticated manner. Let us assume that we want to determine the approximate border between two states in the US. The lazy learner might start with the hypothesis that the border is a straight line half-way between the capital cities of the two states (Figure 4). By taking into consideration a large number of towns on either side of the border it is then able to construct an intricate border based on just the location of each town and the state that the town belongs to. K-nearest-neighbor is a refinement of the nearest-neighbor algorithm by taking into account several nearest neighbors instead of just one nearest neighbor. For example, if the first nearest neighbor of the case being analyzed is sufficiently like the test case but the next two nearest neighbors

[6] While the computing power to process the data may be available, the corpus of data may be insufficient. We would like to have at least a few tens of each combinatorial instance (i.e., each case), which will require millions of patients.

[7] The *nearest-neighbor* algorithm is also known as the *lazy learner*.

are not then the combined vote would be that the test case classification is not appropriate for the case under consideration.

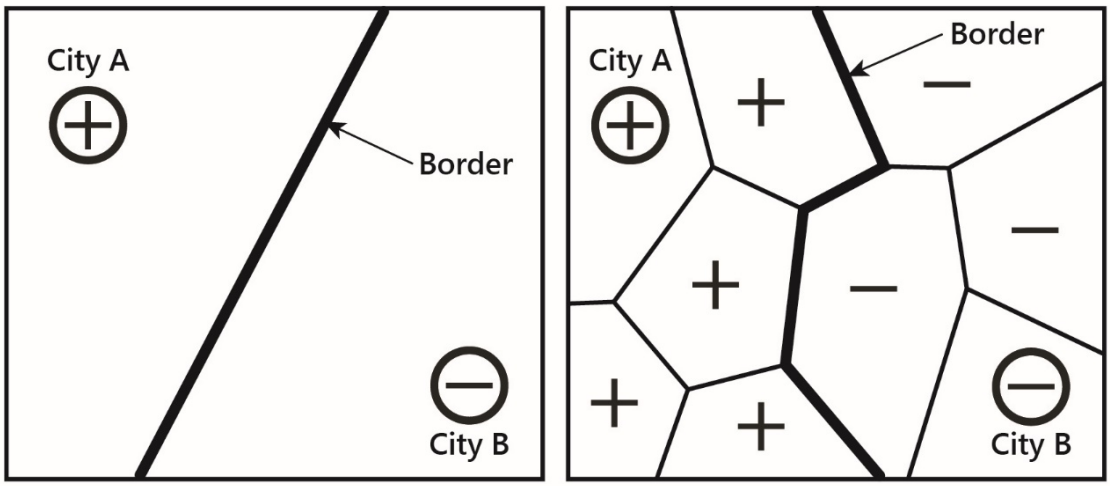

Figure 4: Nearest-neighbor algorithm example

In the early 1990s the nearest-neighbor algorithm was essentially supplanted by the Support Vector Machine (SVM) concept developed by the Russian Vladimir Vapnik at Bell Laboratories after he immigrated to the US from Russia in 1990 (Ben-Hur et al. 2001). SVM is similar to a weighted k-nearest-neighbor. The division between the positive and negative cases[8] is defined by a set of weighted examples together with a similarity measure. A test example is considered to belong to the positive class if, on average, it looks more like the positive examples than the negative examples. In the border example above, SVM will remember only the key examples that define the location of the border line.

The two most difficult problems encountered by analogical learning is how to measure similarity and what to infer from the similarity. A typical example is an automated Help Desk. Chances are that the problem that a customer is encountering with a product has occurred previously and that a solution has been successfully implemented. As Domingos (2015, 199) points out Wall Street hires many physicists because scientific and financial problems have a similar mathematical structure. Similarly, persons with a degree in architecture often end up in professional careers that are unrelated to the design and construction of buildings. Since a typical five-year architecture degree program is heavily oriented toward design, it prepares students for solving problems through the assembly and careful analysis of requirements. Those skills are applicable in many other professional domains.

Symbolistic Approach: The symbolistic approach to machine-learning is based on the representation of objects with symbols that can be manipulated logically, very much like we deal with variables in mathematical equations. In this respect, symbolists are particularly interested in how infants learn to recognize and classify objects during the first 18 months of their life. During this time we learn that we are situated in an environment that is made up of objects that persist over time. We appear to spontaneously establish object categories (i.e., clusters) and gain some understanding of what different categories of object can and cannot

[8] The technical term is classes (i.e., instead of cases) because the nearest-neighbor, k-nearest-neighbor and SVM algorithms are categorized as classifiers.

do. For example, dolls and teddy bears cannot fly while birds, bees and balloons can fly.

The ability to cluster is fundamental to human intelligence (i.e., biological intelligence) and is often the first step in acquiring knowledge. A cluster is a group of physical objects or non-physical entities that have similar characteristics, or are at least more similar to each other than members of other clusters. The quest for an algorithm that can automatically group together entities into clusters is an intensely pursued area of research in machine-learning. A cluster typically has a prototypical set of attributes such as an average height or weight of a person in a cluster of people, even though none of its members may be of that exact height or weight. Or, the desirable product specifications of an abstract cluster such as a market segment or potential consumer group. Without any preexisting knowledge a clustering algorithm will need to start off by assuming that each new entity is part of a separate cluster unless it is in some ways similar to the entities in an existing cluster. With a computer being able to perform millions of computations per second such a clustering algorithm is able to fairly rapidly group thousands of entities into a hierarchical structure of clusters and sub-clusters based on some rules that define the desired degree of similarity that the members of a sub-cluster should adhere to.

This is essentially the way the *k-means* algorithm that was first proposed by Stuart Lloyd at Bell Laboratories in 1957 works (Lloyd 1982). While it is simple and popular, it has at least one serious hurdle to overcome. The *k-means* algorithm works only if the clusters are well differentiated. We can overcome this hurdle to some extent by providing external assistance. For example, by training *k-means* with the attributes of members of existing clusters and the probability associated with each attribute. Another approach is to reduce the number of possible similarity dimensions through a process of *dimensionality reduction*. In the case of clustering images we can reduce the number of visible differences (i.e., dimensions) at the pixel level to a much smaller number of combined features. For example, with only 10 choices for each facial feature, a law enforcement artist can draw a portrait of a suspect that is often good enough to recognize that suspect.

In the late 1970s Newell and Rosenbloom (1980) proposed the concept of *chunking* to explain why the rate of performance improvement in learning a skill is not constant. The rate falls off after an initial learning period, with the performance varying with time raised to some negative power (Figure 5); - referred to as the *power law*.

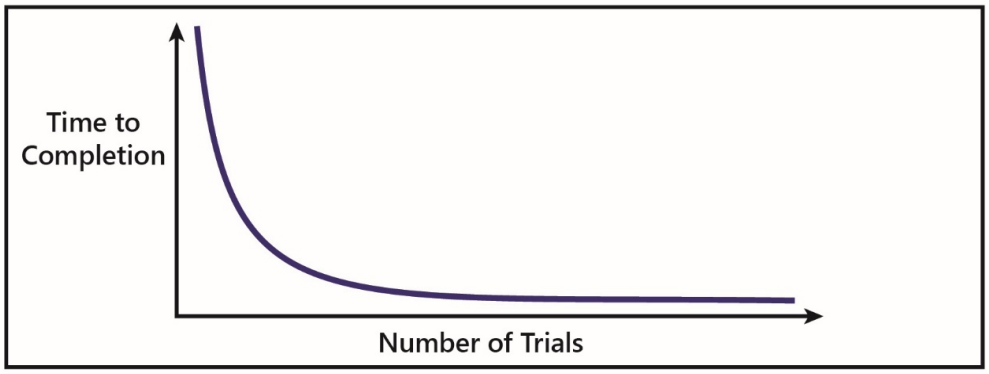

Figure 5: The Power Law

They argued that our ability to perceive and remember in chunks allows us to process much more information than we could if we had to deal with each information item individually.

We can remember the telephone number 805-697-0034 much more easily than without separating hyphens (i.e., 8056970034). In learning a skill we progressively build chunks that consist of chunks. For example, a major difference between a novice and an experienced chess player is that the experienced player judges a game by the positional pattern, while the novice player sees individual chess pieces. Protocol studies have shown that we solve problems by decomposing them into multiple levels of sub-problems depending on the complexity of the problem. The solutions of these chunked sub-problems are essentially symbols that are then combined into a solution of the problem as a whole.

It was initially thought that since chunking is a fundamental component of learning that a chunking-based algorithm may be sufficient for machine-learning. This led to the proposal by Newell, Laird and Rosenbloom (Laird et al. 1986; Rosenbloom 2006) of a general theory of cognition and the Soar general problem-solving program (Laird 2012). While Soar worked well within a predefined hierarchy of goals and was even able to define and solve new sub-problems, it ultimately failed to live up to the initial expectations. In particular, as Soar learned more complicated chunks the program slowed down instead of becoming faster. Even though chunking is not as prevalent in the business applications of machine-learning as clustering, supervised learning and some other learning algorithms, it is recognized as an important contribution from psychology that seems assured of playing a role in the continued evolution of AI.

Tribal Differences

In summary: symbolists believe that intelligence can be reduced to the manipulation of symbols; connectionists simulate the operation of the biological brain, which learns by changing the strengths of the connections (i.e., synapses) between neurons; evolutionaries simulate natural selection using genetic algorithms; Bayesians rely on probabilistic inference to overcome uncertainty; and, analogizers look for similarities between disparate situations to discover other similarities. Each of these five approaches to machine-learning has strengths and weaknesses.

All of the algorithms used by the five approaches are guided by instant gratification. There is a subfield in machine-learning entitled *reinforcement learning* that focuses on algorithms that are designed to explore future opportunities that may sacrifice taking advantage of an immediate reward, just like a chess player may sacrifice a piece for a positional advantage that may eventually win the game. The core concept of *reinforcement learning* is that every state has a value, even though it may not have a reward (Sutton and Barto 1998; Hutter 2004).

While the connectionist and evolutionary approaches to machine-learning both attempt to simulate nature, they take different directions. The connectionist approach uses many copies of a relatively simple mathematically formulated component (i.e., the neurode) that are interconnected as nodes within a network. The flow of signals between nodes is controlled by weighting factors that are incrementally adjusted during thousands of cycles based on specific training criteria. In the evolutionary approach the focus is on creating many alternatives, combining parts of the best alternatives based on a fitness function, and repeating this process over many generations to produce an optimized learning algorithm. Domingos (2015, 138) argues that neither approach is sufficient by itself, but that there is a need to combine the structure learning approach (i.e., nurture) of the evolutionaries with the weight learning approach

(i.e., nature) of the connectionists[9]. He points to natural evolution where structure (the brain) evolves in tandem with the need to process environmental stimuli, citing as an example the growth of the cortex in response to the learning associated with the sensory areas.

There are two principal differences between the connectionist and symbolist approaches. First, in symbolism there is a one-to-one correspondence between symbols and what they represent, while connectionist representations are distributed among many neurodes. Second, symbolist learning is sequential, while connectionist learning is a parallel operation with neurodes learning simultaneously. One might ask: How successful can these connectionist simulations be in the light of advances in technology? The number of transistors in computers is catching up with the number of neurons in the brain, but the brain has orders of magnitude more connections (i.e., synapses). Each transistor in a computer is connected to only a few other transistors, while a single neuron may have thousands of synapses. The computer can to a modest extent make up for some of the lack of connectivity by being much faster than the brain. While a computer can make millions of calculations per second, neurons will typically fire less than a thousand times per second. However, there may be billions of neurons firing during a single second.

Symbolists dominated the first few decades of cognitive psychology (1940s to 1970s). Connectionists came to the foreground in the 1980s and 1990s, but now Bayesians are on the rise. Thomas Bayes, an 18th Century English clergyman, proposed the concept of considering an a priori probability. However, it was the Frenchman, Pierre-Simon de Laplace, who 50 years later formulated Bayes theorem. He argued that every morning that the sun rises should increase our confidence the sun will also rise tomorrow. After hundreds of years the probability that the sun will rise tomorrow will be very close to 1 (but not quite 1), because we can never be completely certain that the sun will rise. Based on Laplace's Rule of Succession the probability that the sun will rise again tomorrow morning after it has risen on n successive previous mornings is (n+1)/(n+2). If n is 0 then the probability is 50% and if n is 100 then the probability is 99%.

Frequentists, like Ronald Fisher, have been historically at odds with Bayesians. While frequentists believe that the only acceptable way to estimate a probability is to count the number of times that the particular event occurred and then divide it by the total number of observations, the Bayesians start off with an initial subjective estimate that is then improved or disproved by new information. According to frequentists, probability is based on frequency of occurrence and according to Bayesians probability is a subjective degree of belief. For frequentists to estimate the probability of an event, the event must have occurred at least more than once. However, Bayesians are able to subjectively estimate the probability of an event occurring that has never occurred previously.

Domingos (2015, 174-5) argues that in machine-learning there is a need for both probability and logic. A Bayesian network can model one aspect of how cells function, but logic is needed to build a comprehensive model of cell operations. On the other hand, logic cannot deal with incomplete information or noisy data, while Bayesian networks excel in this area. While we can combine the evolutionist and connectionist approaches by building a network structure and determining the weights between nodes with backpropagation, unifying probability and logic is a much more difficult undertaking.

Analogy continues to play a prominent role in machine-learning. According to Hofstadter

[9] Domingos (2015, 140) further argues that in the context of information systems, nature is the computer program and nurture is the data that it processes, and that neither of them is more important than the other.

(Hofstadter and Sander 2013, Hofstadter 1999) all intelligent behavior reduces to analogy. He argues that everything that we learn, from the meaning of everyday words such as mother, father, house, and car to our understanding of concepts and acquisition of knowledge is the result of analogous thinking. Throughout the recent history of cognitive science there has been a debate between analogizers and symbolists in respect to which of the two approaches is more comprehensive in respect to its ability to model a phenomenon. While the symbolist's approach to modeling is to devise rules (i.e., rule-based), the analogical approach is instance-based learning (Domingos 2015, 200-2).

Conclusion

While clustering and dimensionality reduction are certainly powerful tools in the symbolistic approach, they like the algorithms used in the other four machine-learning approaches are mathematically predefined. They cannot change their mathematical formulation based on the results of their data analysis. An important part of the learning process of an infant is that the infant is actively engaged in the environment by being able to touch objects, play with them, experiment, use them as tools to pursue objectives, and so on. Children are not taught to crawl, walk and run. They acquire those skills on their own through trial and error driven by a strong desire to be mobile, because mobility facilitates the exploration of the environment in which they are situated. Emotions play a role in as much as children, like adults, seek pleasure and avoid pain. According to Thorndike's *law of effect* (1898) actions that result in pleasure are more likely to be repeated than those resulting in pain. Humans are able to associate both pleasure and pain not only with that actual experience but also with the sequence of actions or events leading up to that experience.

What the infant achieves during the first few months does not involve much teaching. The infant essentially learns through physical interaction, trial and error experimentation, and curiosity. None of the existing machine-learning algorithms are capable of or even intended to emulate this experience-based approach to learning. What is probably highly significant is that the infant's brain, which is the core of its learning capability, forms in parallel with the learning process. The infant does not learn by applying an existing capability, instead the capability evolves driven by the infant's interaction with the environment. In other words, the infant's learning capabilities develop as an integral part of the learning experience. This is a very different paradigm in comparison with a machine-learning algorithm, whose operational mechanism and capabilities are fixed from the start with the objective of using these capabilities to extract knowledge from data.

Notwithstanding the above, a great deal of progress has been made in machine-learning, particularly during the past decade. However, we are still very much at the beginning of truly intelligent computer-based capabilities. What is significant is that some of the approaches described in this paper have learning capabilities. While these capabilities are still in their infancy and therefore primitive in comparison with human learning capabilities, they nevertheless suggest that more sophisticated computer-based learning is feasible in the not too distant future. As Domingos (2015) points out a master algorithm that would bring machine-learning to an Artificial General Intelligence (AGI) level will require the seamless integration of the distinctly different approaches pursued by the connectionists, evolutionists, Bayesians, analogists, and symbolists.

It is easily argued that without automated learning capabilities Artificial Intelligence (AI) cannot be more than a set of tools for human decision makers. Even though such tools have already

proven themselves to be powerful decision-assistance aids, they serve in a supportive rather than primary role that is on par with human intelligence. Husain (2017, 22-24) points to the distinct difference between the computer's ability to analyze a large number of alternative moves in a chess game and mathematically assess the positional outcome in each case to considerable depth, and the operation of our brain. This brute force approach, which may appear to be a sign of intelligence when judged on the basis of the outcome, is very different from human intelligence. Operating at a much slower, energy conserving pace and massively parallel mode our brain will quickly prune the large number of possible moves to a much smaller number of more promising alternatives. The fact that at times the computer will find an alternative sequence of superior moves that may have been overlooked by the human chess master is a consequence of its computational capabilities rather than a sign of chess playing expertise. The acquisition of the latter expertise requires a learning capability and this is why the recent advances in machine-learning are highly significant. Automated learning algorithms combined with enormous computational speed potentially allow the computer to learn in seconds and minutes what would take the human brain weeks and months. Therein lies the promise that AGI and even Artificial Superintelligence (ASI) are indeed eventually achievable.

References

Ackley D, G. Hinton and T. Sejnowski (1985); 'A Learning Algorithm for Boltzmann Machines'; Cognitive Science, 9 (pp. 147-169).

Ben-Hur A., D. Horn, H. Siegelmann and V. Vapnik (2001); 'Support vector clustering'; Journal of Machine Learning Research, 2 (pp. 125–137).

Carnett J. and E. Heinz (2006); 'John Koza Has Built an Invention Machine'; Popular Science, 18 April.

Cohen B. and A. Whitman (1999); 'Isaac Newton, The Principia: Mathematical Principles of Natural Philosophy (The Authoritative Translation)'; University of California Press, Oakland, California.

Domingos P. (2015); 'The Master Algorithm: How the Quest for the Ultimate Learning Machine will Remake the World'; Basic Books, New York, New York.

Drosnin M. (1997); 'The Bible Code'; Touchstone, New York, New York.

Eisenberg J. (2018); 'John Stuart Mill on History: Human Nature, Progress, and the Stationary State'; Lexington Books, Lanham, Maryland.

Hebb D. (1949); 'The Organization of Behavior'; John Wiley & Sons, New York, New York.

Hofstadter D. and E. Sander (2013); 'Surfaces and Essences: Analogy as the Fuel; and Fire of Thinking'; Basic Books, New York, New York.

Hofstadter D. (1999); 'Gödel, Escher, Bach: An Eternal Golden Braid'; Basic Books, New York, New York.

Holland J. (1992); ‚Genetic Algorithms: Computer Programs that evolve in ways that resemble natural selection can solve complex problems even their creators do not fully understand'; Scientific American, July (pp.66-72).

Husain A. (2017); 'The Sentient Machine: The Coming Age of Artificial Intelligence'; Simon &

Schuster, New York, New York.

Hutter M. (2004); 'Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability'; Springer, Heidelberg, Germany.

Koza J. (1992); 'Genetic Programming'; MIT Press, Cambridge, Massachusetts.

Koza J., F. Bennett, D. Andre and M. Keane (1999); 'Genetic Programming III'; Morgan Kaufmann, San Francisco, California.

Laird J. (2012); 'The Soar Cognitive Architecture'; MIT Press, Cambridge, Massachusetts.

Laird J., P. Rosenbloom and A. Newell (1986); 'Chunking in Soar: The anatomy of a general learning mechanism'; Machine Learning, 1(1) (pp. 11-46).

Lloyd S. (1982); 'Least squares quantization in pcm'; IEEE Transactions on Information Theory, 28(129) (pp. 137).

McCulloch W. and W. Pitts (1943); 'A logical calculus of the ideas imminent in nervous activity. Bulletin of Mathematical Biophysics, vol. 5 (pp. 115-133).

McGrayne S. (2011); 'The Theory that Would Not Die: How Bayes' rule cracked the enigma code, hunted down Russian submarines, and emerged triumphant from two centuries of controversy'; Yale University Press, New Haven, Connecticut.

Michalski R. (1980); 'Knowledge Acquisition Through Conceptual Clustering: A Theoretical Framework and an Algorithm for Partitioning data into Conjunctive Concepts'; *Journal of Policy Analysis and Information Systems*'; 4(3), September (pp. 219-44).

Michalski R. (1983); 'A Theory and Methodology of Inductive Learning'; Artificial Intelligence, vol. 20 (pp. 111-61).

Minsky M. and S. Papert (1969); 'Perceptrons: An Introduction to Computational Geometry'; MIT Press, Cambridge, Massachusetts.

Mitchell T. (1997); 'Machine Learning', McGraw-Hill, New York, New York.

Newell J. and P. Rosenbloom (1980); 'Mechanisms of Skill Acquisition and the Law of Practice'; Technical Report, September, Computer Science Department , School of Computer Science, Carnegie Mellon University, Pittsburg, Pennsylvania.

Pearl J. (1988); 'Probabilistic Reasoning in Intelligence Systems'; Morgan Kaufmann, San Mateo, California.

Rosenblatt F. (1958); 'The Perceptron: A probabilistic model for information storage and organization in the brain'; Psychological Review, 65(6) (pp. 386-408).

Rosenbloom P. (2006); 'A Cognitive Odyssey: From the Power Law of Practice to a General Learning Mechanism and Beyond'; Tutorials in Quantitative Methods for Psychology, vol. 2.

Rumelhart D., G. Hinton and R. Williams (1986); 'Learning representations by backpropagating errors'; Nature, 323, October (pp. 533-6).

Russell S. and P. Norvig (2016); 'Artificial Intelligence: A Modern Approach'; 3rd Edition, Pearson Education Limited, Edinburgh Gate, Harlow, Essex, UK.

Saxena S., R. Raperya and N. Malik (2017); 'Machine Learning Using Chunking'; International

Journal of Advanced Research in Science and Engineering, 6(2), February (pp. 285-292).

Silver N. (2012); 'The Signal and the Noise'; Penguin Press, New York, New York.

Sutton R. and A. Barto (1998); 'Reinforcement Learning: An Introduction'; MIT Press, Cambridge, Massachusetts.

Appendix: Machine-Learning Methods

The purpose of this Appendix is to briefly define and explain where appropriate the wide range of concepts, methods and algorithms employed with varying success in the quest for machine-learning capabilities.

1. ***abduction:*** Abductive reasoning starts with an observation or set of observations and then seeks to find the simplest and most likely explanation.
2. ***A/B testing:*** Also known as randomized controlled trial (RCT); - particularly in respect to drug testing. A very simple method of testing that is however still widely used in business to test consumer preferences. For example, to determine whether a particular new feature on a company's website will increase sales it is tried out on thousands of randomly chosen customers and then compared with the results of the website without the new feature (Domingos 2015, 226-7). To apply A/B testing to machine-learning it is necessary to first form a hypothesis, work out a randomization strategy, decide on a sample size, and finally chose a measurement method.
3. ***autoencoder:*** An autoencoder is a multilayer neural network whose output is the same as its input. Its value lies in the fact that it ends up encoding the input in fewer bits within the hidden layer in such a way that it is automatically decoded back to the input form as an output. In other words, it is able to encode a million pixel image into a much smaller bit-code invented by itself and then decode that bit-code back to the complete million pixel image as an output. In *deep learning* autoencoders are stacked vertically with the output of the lowest encoder becoming the input of the one above it, and so on. The interpretation of the input from the lowest to the highest autoencoder proceeds in stages from very primitive localized features to more interconnected features to a final interpretation of the input, much the same way that we currently believe the brain interprets images.
4. ***backpropagation:*** After a neural network has processed the input to the output during its forward pass, the output is compared with the desired output. The error is propagated back through the layers of the network and the input. Each neurode adjusts its weighting based on the error and the input that it received during the forward pass. Backpropagation is the connectionist's principal machine-learning algorithm.
5. ***chunking:*** The human brain handles the retention of information in chunks and this enables us to process much more information than if we had to deal with each individual information item. In learning a skill we progressively build chunks of sub-problems that themselves consist of chunks. In a general sense chunking solves problems by reference to past experience that is also stored in chunks. Conversely, when a system has insufficient knowledge to solve a problem it forms a sub-goal. Any result that is produced in the sub-goal results in the creation of chunks. Once a chunk has been learnt the results of an impasse can be directly applied with the chunk(s) created in the sub-goal, if the same situation occurs again. In this way chunking speeds up problem solving (Saxena et al. 2017).
6. ***clustering:*** A cluster is a group of physical objects or non-physical entities that have similar characteristics, or are at least more similar to each other than members of other clusters. The ability to classify entities into categories and combine such clusters into a hierarchical structure of objects (i.e., treat a cluster or set of clusters as an object) is

fundamental to human intelligence. It allows us to acquire skills by decomposing the skill into sub-skills and then combining those sub-skills to acquire a more comprehensive skill.

7. ***Bayes theorem:*** The core concept of *Bayes theorem* is that the search for a solution should start with an initial belief (i.e., hypothesis) and a subjective probability that this hypothesis is correct. As the data analysis produces new evidence the probability of the hypothesis is adjusted up or down accordingly. The following probability equation is used by the Bayes approach:

   P(cause | effect) = P(cause) x P(effect | cause) / P(effect)

8. ***Bayesian network:*** Pearl (1988) introduced a major extension to the application of Bayes' theorem by proposing the concept of a Bayesian network to represent dependencies among variables, with the limitation that each variable depends directly on only a few other variables. If we make the hypothesis (i.e., the belief) the cause and the data the effect then the hypothesis can be as complex as a whole network or as simple as the probability that a flipped coin will come up heads.

   P(hypothesis | data) = P(hypothesis) x P(data | hypothesis) /P(data)

9. ***Boltzmann machine:*** A type of recurrent neural network in which there are sensory and hidden neurons. The weightings between neurons are adjusted statistically according to the Boltzmann distribution, which is a probability measure that a system will be in a certain state as a function of given criteria. Boltzmann machines can be strung together to make more sophisticated systems such as *deep belief networks*. Replacement of deterministic neurons in Hopfield networks with probabilistic neurons (hence the name Boltzmann machines) was first proposed by Ackley, Hinton and Sejnowski (1985).
10. ***Decision Tree:*** A hierarchical (tree-like) structure in which the properties of each node differ by at least one attribute. Therefore, each branch of a decision tree can be represented by one rule. A decision tree model is a classification system that classifies future observations based on a set of decision rules, with maximum accuracy.
11. ***deduction:*** Reasoning from one or more premises to a logically certain conclusion. Therefore, deductive reasoning moves from generalized principles that are accepted as being true to a specific conclusion.
12. ***Deep Belief Networks:*** A class of multi-layer neural networks in which the layers, but not the neurodes within the layers, are connected.
13. ***dimensionality reduction:*** Used to reduce the number of dimensions of data to a more manageable set by defining characteristic attributes. For example, in trying to automatically match the photograph of a robbery suspect with the photographs of thousands of known criminals pixel by pixel, we can establish a few dozen facial attributes to distinguish the faces of different persons.
14. ***empiricism:*** Empiricists believe that all reasoning is fallible and that true knowledge must come from observation and experimentation. Typically, journalists, doctors, and scientists are empiricists. Empiricists prefer to try things to see how they turn out; - in computer science hackers and machine learners are empiricists. Aristotle was an early empiricist followed by Locke, Berkeley, and Hume in more recent times.
15. ***Expectation Maximization (EM) algorithm:*** A statistical classification model that is

capable of clustering cases when crucial information such as the classes of the training data are unknown. EM can fractionally assign a case to two clusters and at the same time update their descriptions accordingly. Like the *k-means* algorithm it alternates between assigning cases to clusters and updating the descriptions of the clusters.

16. ***genetic algorithm:*** A heuristic search method based on the theory of natural selection (i.e., selection, mutation, inheritance, and recombination) that incorporates a fitness function (Holland 1992). The fitness function assigns a numeric score as a measure of how well the current state fits the purpose.
17. ***genetic programming:*** Is a special form of the genetic algorithm concept in which the output are programs instead of functions (Koza 1992).
18. ***gradient descent:*** Using a *backpropagation* algorithm, after thousands of repetitions the neural network's output neurodes will closely match the desired output that recognizes the particular input that it was trained to recognize. During this *gradient descent* operation there is no certainty that the neural network has reached the best (i.e., optimum) output condition even after thousands of iterations; - i.e., it may have reached a *local minimum* of the error.
19. ***Hebb's Law:*** When an axon of cell A is near enough to cell B and repeatedly takes part in firing it, some metabolic change takes place in one or both cells such that cell A's efficiency as one of the cells firing cell B is increased (Hebb 1949).
20. ***Hidden Markov Model HMM):*** A Markov chain where the states are hidden. For example, in a speech-recognition system such as Siri the observations are the sounds spoken to Siri and the hidden states are the written words that the sounds are intended to represent. The probability of the next word given the current word is a Markov chain.
21. ***Hopfield network:*** A neural network in which the neurodes are binary threshold units (i.e., they take on only two different values for their states and the value is determined by whether or not the units' input exceeds its threshold). Hopfield nets normally have units that take on values of 1 or -1.
22. ***induction:*** Inferring from a particular case to the general case. Therefore, inductive reasoning moves from specific instances to a generalized conclusion.
23. ***inverse deduction:*** Instead of the classical model of starting with a premise and looking for the conclusions, *inverse deduction* starts with a set of premises and conclusions and works backward to fill in the gaps. J.S. Mill (Eisenberg 2018) is the chief advocate of the *inverse deductive* method. It is a combination of inductive generalizations obtained by means of the comparative method (or by statistical method) with deduction from more ultimate laws. It is a way to arrive at reality through experiment, observation and conclusion.
24. ***Kalman filter:*** A *Hidden Markov Model (HMM)* in which the states and the observations are continuous variables instead of discrete variables.
25. ***k-means algorithm:*** The *k-means* learning algorithm is used to cluster unlabeled data (i.e., data without defined categories). It is an unsupervised clustering algorithm that aims to partition cases into *k* clusters with a case being assigned to the cluster with the nearest mean. If *k* is predetermined then the greater the value of *k* the smaller the value of the metric that defines the differences between clusters. One of the metrics that is commonly used to compare results across different values of *k* is the mean distance

between data points and their cluster centroid.

26. ***k-nearest-neighbor:*** A refinement of the *nearest-neighbor* algorithm in which not one but multiple nearest neighbors of the case under consideration determine (by vote) whether the case belongs to the training test case.
27. ***Markov Chain:*** A stochastic model[10] describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event (i.e., satisfies the Markov property). A process satisfies the Markov property if one can make predictions for the future of the process based solely on its present state just as well as one could knowing the full history of the process. For example, Google's PageRank algorithm is a Markov chain.
28. ***Markov Chain Monte Carlo (MCMC):*** While a *Markov Chain* involves a sequence of steps each of which depends only on the previous step, *MCMC* adds an element of chance to the *Markov Chain* method with random sampling. For example, to determine the area taken up by a circle that is inscribed within a square (of known dimensions) we could take a felt pen, close our eyes and randomly make a mark inside the square. If we repeat this many times then the proportion of dots that fall into the circle will allow us to estimate the area of the circle.
29. ***Markov Network:*** Also referred to as *Markov Random Fields*. Capable of compactly representing and visualizing a probability distribution that is based on the language of *undirected graphs*[11]. A *Markov Network* represents relationships between random variables. For example, Roger and Sue are both musicians and friends. However, while these two variables depend on each other neither is the cause of the other.
30. ***Nearest-Neighbor algorithm:*** Looks for a similar image or case and then assumes the image or case under consideration to be of the same classification as the similar one.
31. ***Newton's Third Principle:*** Whatever is true of everything we have seen is true of everything in the universe. According to Domingos (2015, 66) this is the first rule of machine-learning.
32. ***overfitting:*** Imagining or assuming patterns that are not really there. For example, the statistical correlation between the electricity consumption in one UK city with the suicide rate in another UK city that held true for several years, but was in fact a mere coincident. Machine-learning algorithms are particularly vulnerable to finding patterns in data that do not exist in the real world. The reason is that the computer has the ability to process vast amounts of data and therefore an unlimited capacity to identify potential patterns.
33. ***perceptron:*** A perceptron is essentially a simplistic mathematical simulation of a biological neuron. First proposed in 1943 (McCulloch and Pitts 1943) the original model of an artificial neuron (or neurode) was only able to represent OR, AND, and NOT gates. By adding variable weights to the connections between neurodes Rosenblatt (1958) showed that a network of neurodes (i.e., neural network) can learn. The name *perceptron* was coined due to Rosenblatt's research focus on perceptual tasks

[10] In probability theory, a stochastic model is defined as a set of random variables. It assumes that future states depend only on the current state and not on any events that occurred before the current state.

[11] In an *undirected graph* all connections between nodes are bidirectional. A graph where the connections between nodes point in a direction is sometimes referred to as a *directed graph*.

such as speech and character recognition. While the *perceptron* was a major step forward in AI it was also found to have serious shortcomings, which Minsky and Papert drew attention to in their book published in 1969 (Minsky and Papert 1969). In particular, the *perceptron* was not able to deal with an exclusive-OR function (i.e., XOR)[12].

34. ***Power Law:*** (see *S-curve*).
35. ***Principal Component Analysis (PCA):*** A method used in *dimensionality reduction* to reduce the number of variables in a corpus of data by representing the data in a different format. For example, creating a linear graph by plotting non-linear data in a logarithmic form or changing (i.e., rotating) the axis of a graph to reduce the distance of data points from the axis (Domingos 2015, 211-214).
36. ***rationalism:*** Rationalists believe that the senses can be deceiving and that logical reasoning is the only sure path to knowledge. Typically, lawyers and mathematicians are rationalists. Rationalists prefer to plan everything in advance before making the first move; - in computer science theorists and knowledge engineers are rationalists. Plato was an early rationalist, followed by Descartes, Spinoza, and Leibnitz in more recent times.
37. ***reinforcement learning:*** Replaces instant gratification with longer term reward seeking. While most machine-learning algorithms are designed to take advantage of any immediate rewards in their decision-making process, a *reinforced learning* algorithm is typically willing to forego an immediate reward in favor of a potential future more favorable reward; - even to the extent of sometimes choosing a random action (Domingos 2015, 218-223). The analogy is with an experienced chess player who may be willing to sacrifice a piece in favor of a positional advantage that may lead to a winning game.
38. ***relational learning:*** Treats data not as unrelated entities but as a complex network of related nodes that can be applied from one situation to another similar situation. For example, having learned how electricity consumption varies with time of day and season in one city it can be applied to another city with somewhat similar characteristics. However, whereas in regular learning all cases must adhere to the same number of attributes, in relational learning the networks can vary in size (Domingos 2015, 227-233).
39. ***S-curve:*** Also referred to as the *Power Law*. Reflects the phase transition of all kinds of phenomena (e.g., magnetization of iron, electron flipping its spin, ice melting, water evaporating, developments in technology, rumors, epidemics, etc.). At first the output increases very slowly with the input. The output then gradually increases more and more until it reaches a first upward knee where it increases exponentially. It then reaches a second downward knee where the output rapidly decreases, only to gradually level out again until the output decreases slowly with the input.
40. ***Support Vector Machine (SVM):*** Also known as Support Vector Network. *SVM* is a supervised learning model with a learning algorithm that classifies data. Given a training set that includes two categories of cases, the *SVM* learning algorithm builds a

[12] For example, best customers of Nike sports shoes are teenage boys and middle-aged women. Young is good and female is good, but young and female is not good.

model that classifies new cases into either of the two categories in the training data.