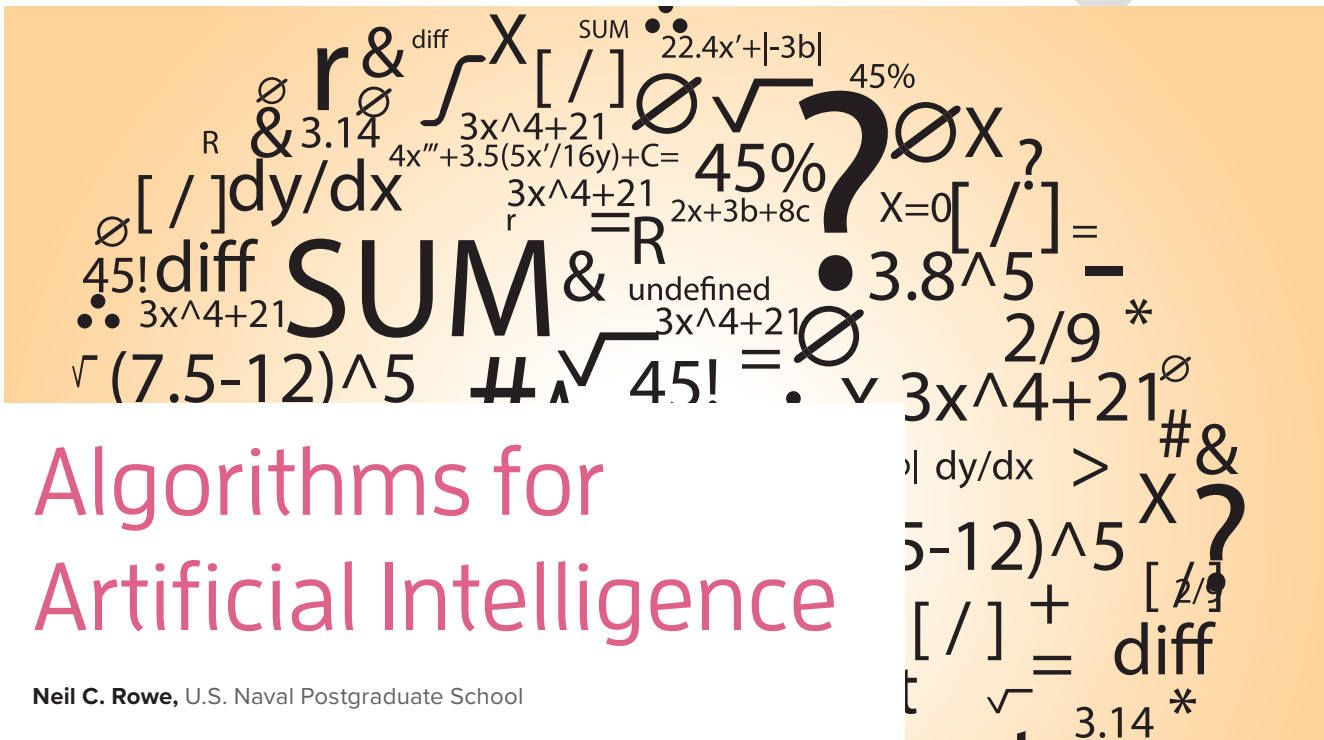


Author Notes

- ✓ The last page contains pull quotes that might be used in the article, depending on design and spacing considerations. Are these pull quotes appropriate? If not, please provide several alternatives. Pull quotes should be concise and taken verbatim from article text.



IEEE PROOF



Algorithms for Artificial Intelligence

Neil C. Rowe, U.S. Naval Postgraduate School

Artificial intelligence and machine learning use a wide variety of algorithms. More than 60 years of research have shown that intelligent behavior requires a variety of methods. We discuss the major classes of algorithms and their uses.

<AU: Please note that the abstract has been trimmed in accordance to magazine style. Please check that the included details are correct.>

Arartificial intelligence (AI) is usually defined as behavior by machines that would appear intelligent if it were done by a human. That does not necessarily require reasoning as brains do. AI is a large subarea of computer science with a long history starting in 1956. Over the years, it has borrowed many algorithms from other areas of computer science, statistics,

Digital Object Identifier 10.1109/MC.2022.3169360
Date of current version: xxxxxx

and operations research toward the goals of creating many kinds of intelligent behavior. It has also invented some algorithms of its own.

BACKGROUND ON AI ALGORITHMS <AU: Please check that the edited section heading is appropriate.>

Many AI algorithms are described in introductory textbooks such as the one by Russell and Norvig.¹ Algorithms can be distinguished as to whether they implement intelligent behavior (AI modeling) or improve it (machine learning). Modeling methods have been around a long time, and machine learning methods have become primary only with the increases in computation speeds in the last 20 years. However, machine learning is unnecessary for AI. You can still program intelligent behavior directly if a reasonably good simple solution for a task will suffice—something possible, for instance, for many help desk tasks. Both AI modeling and machine learning can be distinguished further on whether they focus on logical reasoning, numerical reasoning, or some combination of



both. Early work focused on logical reasoning, but current work is focused on numerical reasoning, though both are necessary to cover the full range of human reasoning abilities.

Some researchers in AI and allied areas of psychology use AI algorithms to model brains (human or animal)² and other aspects of biology such as “artificial life” and immune systems.³ We touch only briefly on these approaches here. This work has important differences from most of AI as biological phenomena are limited in speed and space in ways like semiconductor technology. For instance, brains have short-term and long-term memory of a limited size, and brains have a distinctive top-level process called “consciousness.” Modeling of brains and biology is, however, important in simulating the behavior of people and living creatures as in computer games, as well as suggesting good ideas for AI. Some have argued with a religious-like fervor that brain modeling is the only way to achieve true AI, but this claim is disputed by the generally more impressive reasoning abilities of algorithms without any biological or psychological analogs.

Most problems addressed by AI are NP-hard, so their computations are exponential in problem size. That is because most tasks have features of nondeterminism in which considerable trial and error in the testing of combinations is necessary. However, AI practitioners have been quite clever in finding criteria to considerably reduce the rate of the “combinatorial explosion” with problem size for many practical problems so that implementations of many AI algorithms appear polynomial in average performance.⁴ A good example is the graph-subgraph isomorphism problem (determining whether one graph is a subgraph of another), which arises in many AI areas such as finding objects in pictures and

matching natural-language questions to natural-language documents. It can often be done in polynomial average time when items have rich features to exploit in the matching.⁵ A related problem is that because they are NP-hard, many AI implementations consume large amounts of energy, and “deep learning” methods with artificial neural networks on large data sets, such as picture libraries, can be especially energy intensive.⁶

IMPLEMENTING BEHAVIOR

Logical methods

Logical models are necessary whenever there are absolute constraints on the solutions to a task, as often happens in modeling policies and laws. Computers and digital devices perform logical operations in their machine language, so they have an inherent ability for logical reasoning. Most logical AI algorithms represent knowledge in the form of if-then rules and apply them to Boolean values representing facts to make inferences. Inference can go either forward from facts to conclusions (“forward chaining”) or backward from conclusions to facts (“backward chaining”). Backward chaining is analogous to conventional program execution. The logic used can be either Boolean algebra (propositional calculus), limited forms of predicate calculus, or full predicate calculus. A limited predicate calculus used in the Prolog programming language requires the conclusions of rules to be single unnegated universally quantified facts and can run much faster than full predicate calculus by reducing the combinatorial explosion.⁷ Reasoning with full predicate calculus can be done with the classic method of resolution theorem proving, but other logical inference methods like modus ponens are used, too.

Some AI tasks require the knowledge of many facts, such as answering unrestricted human questions about a subject. Algorithms are then necessary to store the facts, index them, and efficiently retrieve them. Logical methods can then provide encyclopedic capabilities to answer many kinds of questions from such facts. Ontologies (representations of interconnected facts) have been produced as input data for such purposes for many common applications.

If a set of if-then rules must be run frequently, the methods of “compiling” the rules can improve their speed. This can involve creating an equivalent of a customized gate array or semiconductor chip or creating a decision tree of yes-no questions. Special methods called “constraint programming” have been developed for tasks with many logical conditions on a solution, which focus on efficient constraint selection and application; especially difficult scheduling problems can be solved with such methods. Constraint-based tasks appear to be well suited for quantum computing, though implementations are proceeding quite slowly.

Planning methods

A special class of logical reasoning is needed for reasoning about actions in time, the subarea of “planning.” For instance, repairing a vehicle or doing surgery often requires a precise sequence of steps done in a precise order. Tree traversal algorithms are helpful for these problems because the space of possible states often resembles a tree. Rules can be used to guide choices, what is known as a “heuristic search.” Numbers can also be used to guide choices, as in best-first search or branch-and-bound search. A classic and versatile search algorithm is the A* algorithm, which combines best-first and branch-and-bound ideas by adding the costs incurred to a state

to a prediction of the future costs to a goal state. Good tasks for A* search are route planning and resource allocation. Alternatives are cross-entropy optimization⁸ and random searches such as Monte Carlo tree search and simulated annealing.

A subclass of planning addresses extended interactions between two opposing agents, also the subject of game theory in operations research. This has become especially important recently for its applications to cybersecurity planning. Traditional game modeling involves adversaries who alternate turns in planning, each trying to thwart the other's goals or increase the other's costs. To plan in adversarial situations, it is important to look ahead several moves; a classic heuristic called alpha-beta pruning can be used to prove that certain subtrees can be ignored if the adversary always makes their best choice. Some degree of randomness in exploration can also be helpful in problems such as planning business strategies.

Numerical methods

The field of statistics has many models with integer and real variables, and many of these can be used for AI. A simple linear weighted sum of numeric input variables (a "linear model") is widely useful in modeling human decision making. In fact, humans often unconsciously follow linear models in making many decisions, an observation that marketing strategies have efficiently exploited. However, a fundamental weakness of linear models is that they cannot be stacked; a weighted sum of linear models is itself a linear model. This means that linear models cannot explain the levels of computing in brains, a realization that only became clear in the 1960s.

The solution was to insert nonlinear functions between linear models in a graph structure, creating artificial neural networks; the models ("neurons") at the same distance from the input are called "layers." Traditionally, the nonlinear function was the logistic

$f(x) = 1/(1+e^{k(x-c)})$, though other functions are used now. Despite their name, artificial neural networks generally do not try to model biological neurons. However, neurons have similar nonlinearities in the rate at which they fire, their primary quantitative encoding, since they will not fire unless they have more than a threshold of excitation, and with increasing input, they asymptotically approach a maximum due to chemical limitations on the rate of molecular diffusion. Many variant algorithms for neural networks have been proposed over the years, but no one method has been shown to provide consistently better performance than the others.

Linear and many mostly linear numerical models are easy to implement with vector and matrix operations in digital technology, or they can be implemented with analog (voltage-based) computational elements. They can run very fast because they primarily add and multiply without iteration. They can also be combined with logical models by comparing their outputs to thresholds and interpreting a value over the threshold as a True and under as a False. Or the thresholds can define boundaries that partition hyperspace in subregions defining a class of data ("support vector machines"). This is useful for many kinds of classification, such as that of sounds and signals.

Two particular kinds of artificial neural networks have become popular recently. For computer vision, the intelligent processing of images, convolutional neural networks have supplanted traditional methods and have been shown to solve many classic problems much better. They are neural networks where the initial ("convolutional") layers take inputs from a local area of the image, and then the later layers combine information over the whole image. The early convolutional layers generally identify small objects in the images, and then the later layers identify relationships between them; an example would be an aerial photo where the convolutional layers

identify vehicles, and the later layers identify traffic jams. The other popular kind of neural network is the recurrent neural network, which explores locality in time analogously to the spatial dimensional of convolutional networks. They have additional inputs to layers that are previous values of their inputs or summaries of previous values. Recurrent neural networks are a good way to recognize and classify sequences of events from simple sensors, as in the monitoring of vehicle or human traffic for accidents.

An alternative to a linear model is a multiplicative model where the factors for a conclusion are multiplied rather than added. Many of these methods multiply probabilities and derive from Bayes' rule in mathematics by imposing various additional assumptions on the independence of evidence. Multiplicative models can be considered the exponentiation of linear models, where the weights in linear models become powers of the exponentiated values. However, when used with probabilities, they have the advantage that one high probability does not overwhelm the other factors, unlike one high factor in a linear model. Human expertise is often used to connect Bayesian values in a network along likely causal relationships, rather than trained as with artificial neural networks.

"Fuzzy" models and networks are popular to address evidence uncertainty, and they use special mathematics to combine evidence, especially linguistic evidence. An older alternative to recurrent neural networks for reasoning about uncertainty for events in time is the hidden Markov model where probabilities are inferred for successive states from probabilistic observations and known transition probabilities; a classic use is for speech understanding.

Specialized AI methods

Brains have specialized capabilities for sensing and motion that allow an animal to interact with its environment,

and these are necessary for intelligent behavior as well. The AI algorithms for these tasks follow more closely to biological models than other algorithms. For instance, computer vision must recognize the boundaries of objects in an image before it can recognize the objects, and this is accomplished in the retina in an eye and the subsequent visual preprocessing at the back of the brain; audio processing needs to recognize the distinctive patterns in sets of frequencies of sounds, and this is accomplished in the cochlea of the ear and the subsequent neural auditory preprocessing. Many of the algorithms use artificial neural networks since anatomical investigations have documented real neural networks for these tasks.

AI algorithms are often good candidates for distributed processing, and pipelined architectures originally designed for fast graphics (“graphical processing units”) have greatly contributed to the recent success of artificial neural networks. However, distributed processing plays a more intrinsic role in algorithms that model organizations or societies—“social AI”; they model intelligent communications between “agents” (intelligences). Social AI differs from protocols for digital networks in that it can negotiate in human-like ways rather than issue requests and reports.³ A synergistic effect can occur where the network of communicating AI agents is smarter than any one of them alone. One simple form of this is “swarm intelligence,”⁹ which can model coordinated herding and flocking actions by animals as well as analogous phenomena in military operations.

MACHINE LEARNING

Recent attention to AI has been focused on the subarea of machine learning and the related area of data mining.¹⁰ With machine learning, a carefully designed starting model for intelligent behavior is unnecessary; instead, an adequate model is optimized automatically. Most algorithms for machine

learning have been around for a long time but have not been sufficiently fast to be useful until recently; my AI textbook, which came out in 1988,¹¹ said little about machine learning.

Machine learning can be supervised (training on examples), unsupervised (using some less direct feedback), or some combination of both. Training examples in supervised learning must specify what should be concluded from the case; for instance, a training set for medical AI could be patient records with their diagnoses. Unsupervised learning often uses numeric feedback, such as the degree of quantitative success or distance to a goal state. Most machine learning is supervised because that is much faster.

Logical learning

The caching of previous results is a simple and often effective method of supervised learning. A new case can be assumed to have similar conclusions to similar, previously seen cases; this is “case-based” or “instance-based” reasoning. For instance, a medical system may remember the pattern of symptoms in rare cases and what the eventual diagnoses were. However, a distance metric between the current situation and stored situations is needed, and this is not always easy to define; if more than one situation is similar, an average or consensus of conclusions of the “nearest neighbors” should be taken. Stored cases should be indexed to enable quickly finding the nearest matches.

A still-used supervised learning algorithm from the late 1950s learns a decision tree incrementally from Boolean data. Decision trees are often a good model for tasks governed by policies, such as filling out tax forms. Starting with an empty tree, it considers each training case in order, following the tree according to the features of the case until it gets to a leaf node. If this process does not lead to a correct conclusion according to the training set, the leaf node is changed to query a previously unasked feature that will

distinguish this case from the previous ones. Cycling through the cases until no further changes are made to the tree ensures that the tree handles the cases in its training set correctly if they are not contradictory.

However, the tree built may be quite unbalanced; entropy calculations can help balance it by considering the features whose querying reduces the entropy the most. The accuracy of the tree on new cases can still be a problem, however, since the tree does not generalize from its experience. A popular improvement called a “random forest” builds a set of trees with random orders of questions and takes a majority vote on their predictions; this seems to perform well on many practical problems such as classifying objects in images. The multiple trees may require more energy to run than alternatives like neural networks, which, however, require more energy to train.

An alternative model for logical data is a set of if-then rules interpreted as an implicit conjunction. For an incremental approach, rules can be modified as necessary to conform to each new case. This may require adding terms to the premises (often negated terms representing exception conditions), deleting terms, or joining two rules together. New rules should be created for cases sufficiently different from any seen before. As with decision trees, cases should be cycled through until the rules no longer change. An alternative is to learn a set of broader but imperfect (“heuristic”) rules since humans use many such rules. The “set covering” approach, a “greedy” algorithm, uses conditional probabilities of the conclusions given the premises to rank rules. It first finds the best one-premise rules, then combines them by taking conjunctions or disjunctions of the “if” parts to get new rules, and repeats to build larger and larger rules of a minimum conditional probability. For instance, a person may think they get sick after eating seafood, but they may discover after further tries that the combination of

seafood with garlic and high amounts of fat is more likely to cause the symptoms.

Numerical learning

The classic way to fit numeric models to data is through regression methods from statistics, and these are widely used for the simpler numeric models in AI. However, artificial neural networks have their own specialized learning algorithm called “backpropagation,” which adjusts the weights in networks incrementally to training data. It uses the chain rule for finding derivatives and optimizes the weights by working backward from the final layer to the input layer. Its idea is that weights should be increased for a case proportional to the correct output minus that actual output (which can be negative). The degree of blame or credit should also be proportional to the variable value multiplied by the weight on that case, but since a nonlinear function was imposed after the weight was applied, it should also be proportional to the slope of the nonlinear function then. These three factors should be multiplied.

Backpropagation ran into accuracy limits in deeper networks during the 1980s until it was realized that neurons at deeper layers were frequently getting stuck on extremes of the ranges of their nonlinear functions. The solution was to normalize the set of weights for a neuron periodically to get them back closer to the middle of their ranges, where adjustments could work faster. This enabled neural networks to have more layers, hence the current vogue for “deep learning.” Note that learning for neural networks fits known cases to a complicated mathematical function and cannot be said to be “creative” (in the sense of forming something new).

Numerical learning of plans

Planning methods can be trained using recurrent neural networks where successive outputs represent successive steps in a plan. Increasingly

popular, however, is to use reinforcement learning to improve planning. One simple approach is inspired by ant trails as marked by chemical signals (pheromones) and modifies the costs of branches taken in a traditional search tree with backpropagation of the degree of the eventual relative success or failure of plans using that branch. Relatively successful routes to a solution decrease the costs of their steps, whereas relatively unsuccessful routes increase them. By running enough examples, consensus costs are obtained on the choices of plan steps. An example is businesses learning good production and marketing strategies. Reinforcement learning only works, however, when there are few enough branches to obtain good feedback on them, and many important planning tasks are too complex for this to work.

Adversarial and game-search tasks may use an adversary’s actions to learn better than they could on their own. That is because adversaries deliberately seek weaknesses in their opponents, and that helps opponents to focus on the features of their strategies that most need improving. Recent examples have been the surprising successes of AI programs against people in games like poker¹² and Go that were previously thought too difficult for AI methods.

Unsupervised learning

Supervised learning methods need training cases, and for many important AI tasks, such as computer vision, huge training sets are required because of the wide variety of possible inputs that must be handled. Big data methods have developed considerably in recent years, so obtaining and working with millions or billions of cases may not be unreasonable, as with the task of assigning names to people from their online photos, which only some are captioned. Nonetheless, many important data collections do not come with preassigned conclusions, such as collections of new Internet traffic for the identification

of suspicious activity. Unsupervised learning can find patterns and hidden classes in unlabeled data.¹³ However, there must be at least some feedback to the learning algorithm to guide it. The feedback can be a rating of the apparent success (as with planning), consistency of the model with new data (as with classification problems), or some measure of interestingness of what the algorithm has done.

Clustering algorithms are a classic form of unsupervised learning that group related cases in data; the groups they find can be used as labels for supervised learning. Clustering has a long history in many areas of science, most notably in the social sciences in identifying types of behavior. Evolutionary algorithms are unsupervised methods that use semirandom heuristic searches to rate states by “interestingness” and use heuristics to create new interesting states by combining and modifying the most interesting known ones. They were originally inspired by genetics and are good at discovering surprises that other methods may overlook. An example of usage is finding technological surprises that could affect a military organization.

Neural networks can be adapted for unsupervised learning by using two of them together. A first idea was an “autoencoder,” where one neural network created a random compressed encoding of its input, and another neural network tried to decode it back to the original input. Encodings that could decode closest to the original input caused weights to be increased in both networks. This idea has developed into “generative adversarial networks,” which model learning as a game where one network (a “generator”) tries to create input to fool another network (a “discriminator”). The weights on the generator neural network are increased when it fools the discriminator well, and the weights on the discriminator are increased when it is not fooled much. A much-discussed application is creating fake faces of people. Despite notable successes,

unsupervised learning methods are hard to control and inconsistent, and many never discover anything useful. However, they may be worth a try for important problems since they are truly creative compared to supervised learning.


COMPARING AND COMBINING ALGORITHMS

With so many AI algorithms, a frequent question is which one is best for a particular task. This question is usually answered experimentally by trying the algorithms on a sample of the data and measuring their performance with prechosen metrics. In supervised learning with a training set, part of the training set can be set aside for testing. Recall and precision are standard measures of accuracy for logical models, and root-mean-squared error is standard for numerical models; average processing speed, storage required, energy required, and explainability of the reasoning are other important metrics. However, practically speaking, little difference in performance is observed for many algorithms on many tasks, as is observed in running AI packages, such as Weka and Scikit-Learn, that offer a variety of algorithms to use on the same data. That is because it appears that most models provide representational (or “epistemological”) adequacy for their tasks, meaning that most models are usable for most AI tasks with proper adjustments.


However, there are exceptions. Logical models cannot adequately represent, without significant complexity, the adding of effects from many weak factors in a linear model. Similarly, some numerical models cannot adequately represent some logical constraints, as for instance, a linear weighted sum of numeric factors cannot represent an exclusive-or relationship. The latter example raises questions about the epistemological adequacy of artificial neural networks, which are based on weighted sums though they include nonlinearity. However,

epistemological adequacy is not always necessary. A good example is natural-language processing. Despite the fact that natural languages have been shown to require at least context-sensitive grammars, many useful neural networks for speech understanding provide adequate performance from models equivalent to regular grammars (finite-state machines) since the occasional context-sensitive features can be approximated adequately.

The choice of a single AI algorithm may not be appropriate for some tasks anyway. Many successes have been achieved with “ensemble learning,” also called “bagging,” where multiple learning algorithms or their variants are run simultaneously on data; random forests are an example previously mentioned.¹⁴ If the conclusions are categorical, a majority vote can be taken of the methods; if the conclusions are numeric, a weighted average can be taken. Ensemble learning appears to be the best countermeasure to the threat of an adversary manipulating training sets to force learning to reach incorrect conclusions, as could occur with learning of new kinds of cyberattacks.

AI has adopted and modified algorithms from many areas of computer science. Much of this variety is unnecessary for practitioners to examine since many tasks are addressed equally well by most algorithms. Nonetheless, there are important tasks for which one algorithm is better, such as neural networks for computer vision and speech processing. Despite the current vogue for numerical algorithms, logical algorithms remain a better fit for problems of finding combinations and imposing of logical constraints. Though most algorithms are NP-hard, the parameters of the complexity with input size can vary considerably, and in many cases, the tasks are polynomial on the average. So the performance of AI algorithms generally needs to be determined by experiment. 

REFERENCES

1. S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. New York, NY, USA: Pearson Education, 2020.
2. J. Crowder, J. Carbone, and S. Fries, *Artificial Psychology: Psychological Modeling and Testing of AI Systems*. Cham, Switzerland: Springer Nature, 2020.
3. D. Floreano and C. Mattiussi, *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. Cambridge, MA, USA: MIT Press, 2008.
4. T. Roughgarden, *Algorithms Illuminated (Part IV): Algorithms for NP-Hard Problems*. New York, NY, USA: Soundlikeyourself Publishing, 2020.
5. J. Choi, Y. Yoon, and B.-R. Moon, “An efficient genetic algorithm for subgraph isomorphism,” in *Proc. 14th Annu. Conf. Genetic Evol. Comput.*, Jul. 2012, pp. 361–368, doi: 10.1145/2330163.2330216.
6. J. Finks. “Solving Big AI’s big energy problem.” *The Next Web*. <https://thenextweb.com/news/solving-big-ai-big-energy-problem> **<AU: Please provide the date of access.>** 
7. M. Bramer, *Logic Programming with Prolog*. Cham, Switzerland: Springer Nature, 2013.
8. D. Drusinsky and J. B. Michael, “Multiagent pathfinding under rigid, optimization, and uncertainty constraints,” *Computer*, vol. 54, no. 7, pp. 111–118, Jul. 2021, doi: 10.1109/MC.2021.3074264.
9. P. Tarasewich and P. R. McMullen, “Swarm intelligence: Power in numbers,” *Commun. ACM*, vol. 45, no. 8, pp. 62–67, Aug. 2002, doi: 10.1145/545151.545152.
10. I. Witten, E. Frank, M. Hall, and C. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*, 4th ed. Cambridge, MA, USA: Morgan Kaufmann, 2017.

DISCLAIMER

The views expressed are those of the author and do not necessarily represent those of the U.S. Government.

ALGORITHMS

11. N. Rowe, *Artificial Intelligence through Prolog*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1988.
12. "Libratus Poker AI Beats Humans for \$1.76m; Is End Near?" PokerListings. <https://www.pokerlistings.com/libratus-poker-ai-smokes-humans-for-1-76m-is-this-the-end-42839>**<AU: Please provide the date of access.>**
13. M. Celebi and K. Aydin, Eds. *Unsupervised Learning Algorithms*. Cham, Switzerland: Springer Nature, 2016.
14. H. M. Gomes, J. P. Barddal, F. Embrecht, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 1-36, Mar. 2017, doi: 10.1145/3054925.



NEIL C. ROWE is a professor of computer science at the U.S. Naval Postgraduate School, Monterey, California, USA. **<AU: Please provide postal code of current affiliation.>** Contact him at ncrowe@nps.edu.



Algorithms can be distinguished as to whether they implement intelligent behavior (AI modeling) or improve it (machine learning).

Most logical AI algorithms represent knowledge in the form of if–then rules and apply them to Boolean values representing facts to make inferences.

A simple linear weighted sum of numeric input variables (a “linear model”) is widely useful in modeling human decision making.

Many successes have been achieved with “ensemble learning,” also called “bagging,” where multiple learning algorithms or their variants are run simultaneously on data.

There are important tasks for which one algorithm is better, such as neural networks for computer vision and speech processing.

Despite the current vogue for numerical algorithms, logical algorithms remain a better fit for problems of finding combinations and imposing of logical constraints.