



Teaching intelligence:

The secret to useful, production-ready AI



| Contents

Introduction	3
Useful AI recognizes AI as an evolution of automated control and optimization systems	4
Useful AI creates control and optimization systems that learn	6
Expert teachers for expert learnings: Helping real AI succeed beyond simplified learning scenarios	8
The next age of useful AI requires teaching intelligence	9
Solving real-world problems with AI	10
Conclusion	11

Authored by:

Kence Anderson

Principal Program Manager, Machine Teaching Innovation
for Autonomous Systems

Microsoft

I Introduction

In 2016, the documentary film *AlphaGo* chronicled how Alphabet subsidiary DeepMind's AlphaGo AI defeated then-champion Lee Sedol in a five-game series of Go. Long considered one of the most strategically complex games ever created (along with games like shogi and chess), AlphaGo was able to pick up the game not as a collection of steps and procedures, but by learning and self-discovering a system of fuzzy rules and strategies. [With this understanding, AlphaGo competed with, and defeated, human champions.](#)

While these feats of AI strength encourage companies like DeepMind to continue their pursuit of human-like “strong AI,” it drives companies like Microsoft with their Project Bonsai and Tesla with their Autopilot AI to innovate new “useful” AI through cutting-edge machine teaching approaches.



In late 2020 Microsoft and Neal Analytics built and certified an AI neural net “brain” to serve as a medium expert operator in a PepsiCo snack foods R&D plant.

[This brain controls the extruders](#) that make Cheetos snack foods, ingesting feedback from a computer vision system to measure quality attributes like length, diameter, density, and curl in real time while using that feedback to learn how to control the extruder across a huge variety of scenarios. The AI additionally learned how to adapt its controls to compensate for changing conditions, such as wetter or drier corn. Most importantly, this brain was designed by chemical and mechanical engineers who imparted actual on-the-ground expertise to the system, rather than AI experts without industry experience.

It's hard to overstate how important this kind of innovation has been in the field of AI. Specifically, we are seeing a transformation in AI around the interrelationship between intelligent AI, neural network brains, and expert machine teaching that connects general-purpose AI and application-specific “useful” AI.

Unlike the rule-based algorithms that drive AI in games of chess on your phone, AlphaGo **learns** strategies and creative responses to each board position with an impressive measure of forward thinking and planning based on its experience playing many games against itself. In this way, it resembles human skill acquisition more than systems that calculate or look up their next move.

Indeed, there are concrete differences between how researchers develop AI in labs to test the limits of what AI can learn on its own, and the ways that companies like Microsoft, Tesla, and SpaceX teach “useful” AI what humans already know about how to perform a particular high-value task in the real world using techniques that have been proven in research labs.

This white paper will outline a path to production-ready, decision-making AI for engineering managers who operate complex, high-value processes. Instead of debating the bar that machines need to cross to demonstrate human intelligence, we draw on key concepts from the history of AI and combine them with state-of-the-art automation technology in pursuit of systems better suited for optimization and control—useful AI. We assert that useful AI provides the next evolution in industrial control, automation, and optimization and that teaching is the primary mechanism for enabling useful AI.

Useful AI can be defined as “intelligent systems that can control or optimize an asset better than existing methods can.”

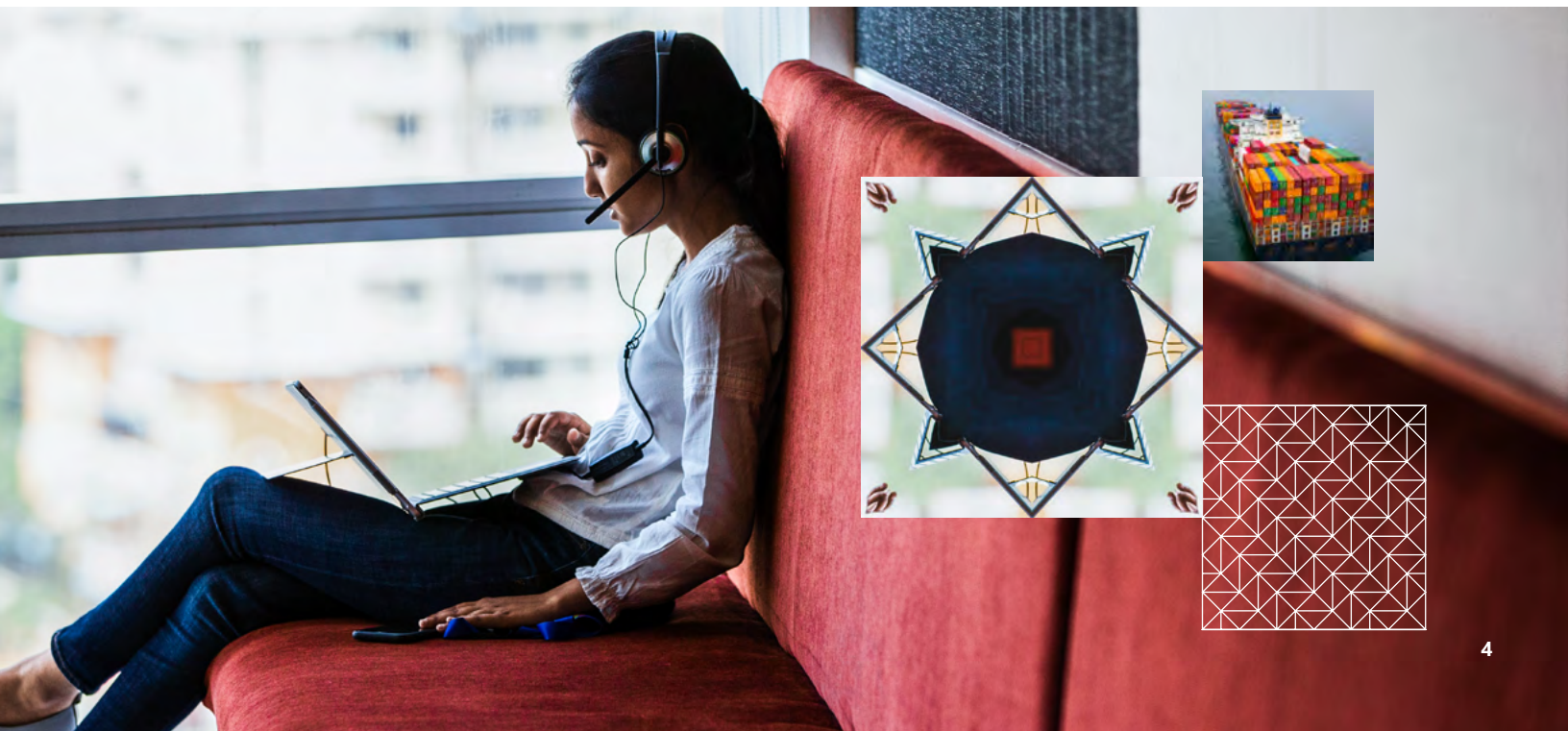
This definition distinguishes AI as a practical solution to real-world problems while still leveraging insights into machine learning gained from gaming examples. Useful AI limits the scope of expectation to complex but well-defined control and optimization problems usually predicated on a specific asset, system, or process.

Useful AI recognizes AI as an evolution of automated control and optimization systems

There are three primary automation methods used in real-world systems and processes.

- ① **Control theory** uses mathematics and science to calculate its next action, usually based on feedback.
- ② **Optimization** algorithms search options, then select a course of action from those options based on objective criteria.
- ③ **Expert systems** codify human expertise as rules and look up methods to make decisions. Each methodology exhibits strengths and weaknesses that we should consider while trying to devise superior control using AI.

Method	How are decisions made?	Strengths	Weaknesses
Control Theory	Mathematical equations calculate the next actions based on sensor inputs.	Reliable and predictable.	Not very flexible; can be susceptible to interference.
Optimization Algorithms	Algorithms search a huge number of options, then select the next action using objective criteria.	Thorough and exploratory; useful when there is limited expert knowledge.	Don't handle uncertainty or fuzzy real-world cases well. Time- and compute-intensive.
Expert Systems	Look up actions from a large database of rules and heuristics that navigate clearly defined logical paths.	Work well when a lot of expert knowledge is available.	Can be difficult for actual experts in the field to provide rules. Don't capture the nuances of a field of expertise.





Automated systems have evolved significantly since control theory was born in 1897. Known as PID control, these early systems were built to steer the rudders of navy ships. In 2016, around the same time that DeepMind released AlphaGo, [Boston Dynamics built a robot called Atlas](#), which began walking. It performed leaps and flips in 2018, demonstrated robust walking in 2020, and was recently filmed dancing to famous pop tunes. However, Atlas doesn't use the same kinds of learning AI that DeepMind trained to beat grandmasters in chess and Go. The methods that power Atlas to these amazing feats are more closely related to optimization algorithms and control systems that are used to control processes in factories. Reinforcement learning, the predecessor of the AI behind AlphaGo, has roots in control theory and uses optimization algorithms to help it learn. But we seem to be approaching the end of the runway of what control theory-based automation can do autonomously. Systems like Atlas require a huge amount of time, money, and specialized expertise to build.

Because of these limitations on what automated systems can further accomplish autonomously, humans frequently make supervisory decisions

and then hand them to automated systems to carry out. Humans also often step in to make real-time decisions when automated systems falter in very dynamic conditions or fuzzy scenarios. For example, a mining company may use an expert system to produce aluminum in a smelting plant. The expert system was programmed to alternate between two time-tested strategies that were effective individually but required specific insights about what conditions necessitated moving from one to the other—a task that proves difficult for an expert system locked into a codified rule set. When the expert system causes adverse process conditions by switching strategies too early or too late, machine alerts call humans to step in and use their expert intuition to get the process back on track.

The innovation challenge is to move from purely linear intelligence models that we see in expert systems to add depth, flexibility, and growth to the more flexible systems we see today. Instead of compiling rules and instructions in lookup tables, these flexible systems need to adopt different learning methods that support executive decision making and creative thinking and meet the demands of production engineering and logistics environments.

Useful AI creates control and optimization systems that learn

A neural network is a system of interconnected “nodes,” synthetic imitations of neurons in our brains, that accept different weights (or values of importance) based on the effect that the weight of each node has on the output of the entire network. The network learns how to represent complex relationships between network input and output as the weights of each node evolve. This approach is called deep learning. So, a neural network will associate success in a task with certain conditions and assign more weight to relevant nodes, thus building strategies based on the consequences of its actions.

Imagine how complex the function is that describes the relationship between an arbitrary set of pixels in an image and whether those pixels form a specific image, like a human face, a cat, or a dog. Neural networks build these correlations and correctly identify cats and dogs in images, for example, because they can approximate any function. Before neural networks, engineers and scientists relied exclusively on simpler mathematical relationships to describe how variables related to each other. Because they can approximate any function, neural networks help correlate complex relationships and serve as the foundation for learning in complex, realistic, fuzzy, nuanced environments.




This means that no matter how complex, a neural network can describe the operations you need to perform on any inputs to get any output ([as proven by the Universal Approximation Theorem](#)). So, a neural network can represent the complex functions needed to recognize patterns in pixels without needing step-by-step algorithmic instructions.

Deep learning, when paired with a form of machine teaching called “reinforcement learning,” trains neural networks to output sequential prescriptive decisions like an automated control system. In reinforcement learning, which rose out of control theory in the 1960s, AI learns to make decisions by practicing in simulation. Algorithms train agents (which are just instances of a program that does something in a program, like a machine playing a

game in a multiplayer environment) that [learn from trial and error based on rewards](#). For example, a game-playing agent may learn strategies in a game by completing tasks and earning points. [By earning points based on specific actions, the agent develops an approach to the game.](#)

Researchers combined the flexibility of deep learning with agent-based reinforcement learning to create deep reinforcement learning (DRL) as an approach that [brings deep learning structures to active agents in a reward or weighted reinforcement-learning environment.](#)

While some view DRL through the lens of the heights or hype that AI research can achieve, DRL possesses a quality that sets it far apart if we view it from the lens of a feedback-based control system: **it learns**. This quality of learning by practicing across a variety of scenarios endows DRL with a set of qualities that present as an evolution of control and optimization capabilities.

Learning	
 Traditional systems leverage static expertise	Useful AI learns non-linear relationships across fuzzy conditions
Delayed Gratification	
 Traditional systems often make shortsighted decisions	Useful AI makes decisions with forward thinking
Sensory Perception	
 Traditional systems cannot consider some sensory information	Useful AI considers visual, sound, and categorical features as it makes decisions

DRL exhibits other unique qualities among decision-making techniques, even qualities so compelling that they merit consideration as higher-level executive reasoning. The first unique quality is that it can learn. The second unique quality is that DRL uses forward thinking as it makes decisions. DRL algorithms maximize future reward, which presents as delayed gratification in practice. [Researchers at Microsoft Project Bonsai amusingly experienced this while teaching the Jaco robotic arm to grasp and stack one block on another.](#)

This seven-jointed robot arm folded in on itself while performing the task, so the researcher set a reward for the distance between the block and the shoulder of the robotic arm and a stiff penalty for bringing the block closer to and shoulder. After more practice at the task, the agent learned to wind up, bringing the block closer to its shoulder at great short-term penalty, to strike the block or throw the block. This machine agent learned that it could achieve much greater rewards in the long term by winding up and throwing the block. Of course, what the researcher intended was to reward the agent for keeping the robot wrist far from the shoulder. This change in reward disincentivized the throwing behavior and prompted the robot to learn to extend its arm during the movement.

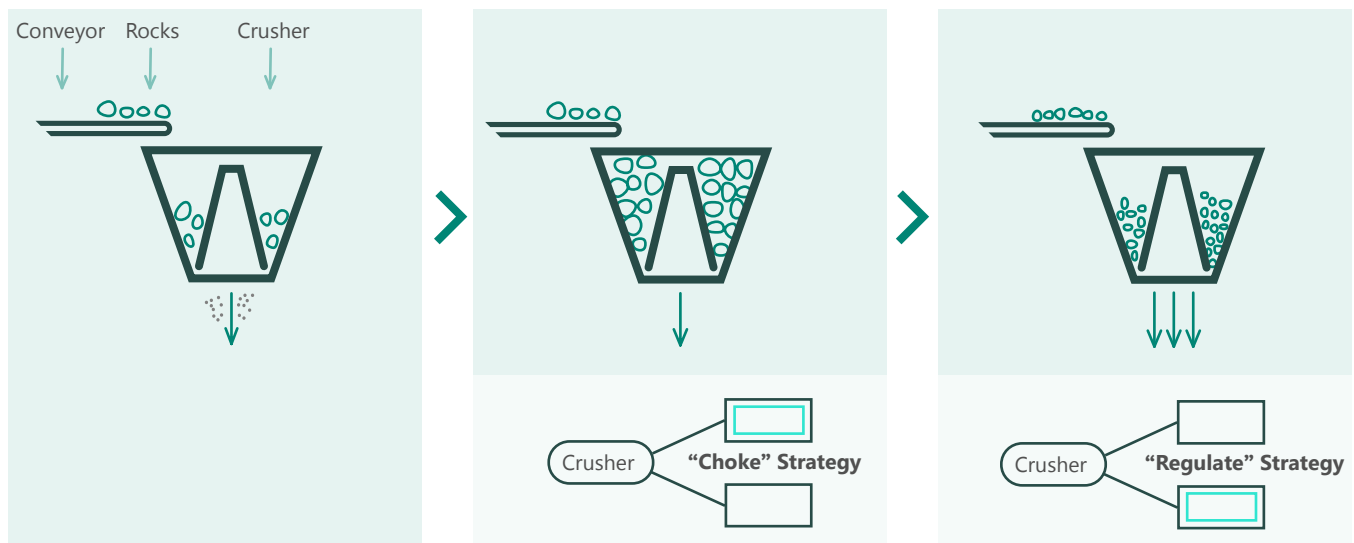
The third unique quality of useful AI that sets it apart from automated systems is the ability to learn from "what it sees and hears" and make supervisory decisions. For example, when controlling kilns that cook limestone as the first step in the cement-making process, operators provide supervisory control based on the visual appearance of the kiln flame. The operators supervise the process differently depending on the shape of the flame, the color of the flame, and the haziness of the air inside the kiln. This kind of "extra-sensory perception" moves AI toward higher-level executive functioning in the context of machine and process control. While automated systems cannot make supervisory decisions based on visual and auditory perception, useful AI can.

DRL is also the only technology of any kind that has demonstrated the ability to **learn strategy**.

Early chess-playing machines utilized programmed strategy, but DRL agents learn strategy as they gain experience. For example, [AlphaChess learned and regularly used the 12 most common opening move sequences in chess on its own without being taught.](#)

If useful AI can learn strategies from the games of Chess and Go, it can also learn strategies that are used to control high-value equipment and processes. For example, a piece of equipment called a gyratory crusher is commonly used to crush rocks as the first step in many mining processes. The goal is to crush as much rock as possible (measured in tons per hour) to the particle size that will fit through the holes in a large shaking sieve. A gyrating steel arm rotates inside the cone and crushes the rocks against the steel cone. The rocks then fall through the bottom of the funnel. If you stuff the crusher chock-full, the resulting compression forces crush even the largest, hardest rocks, but it takes more time for the rocks to move through the crusher. If you fill the crusher $\frac{2}{3}$ to $\frac{3}{4}$ full, you can move more material through the crusher per hour, but the crusher doesn't generate as much compressive force. The first control strategy efficiently crushes large, hard rocks. This second control strategy is perfect for increasing throughput when the rocks are softer and smaller.

A responsive, useful AI can learn to move between these two strategies to maximize throughput. Importantly, however, this AI can be taught these strategies from an expert who knows the strategies and when to use them.



Expert teachers for expert learnings: Helping real AI succeed beyond simplified learning scenarios

Hands-on teaching has always been the most effective technique to bootstrap skill acquisition in learners. If you wanted your child to learn basketball, simply placing them on a court with a ball would not be the most productive way to teach them. You would want your child to know that the skill of dribbling exists and that there are multiple time-tested ways to increase their chances of getting the ball into the net: **the layup, the jump shot, and the hook shot**. Likewise, AI doesn't simply learn a task by learning rules. It learns best with a combination of experience through doing and expert teaching.

Machine teaching attempts to bridge the gap between the algorithm, the data, and the experts by providing experts in different fields with ways to break down problems into steps, which they could then [translate into a system that machine learning algorithms can understand and learn from](#). Programmers and AI researchers usually don't have expertise in the areas for which they develop AI, and subject matter experts often don't have any advanced algorithm or neural network experience. Machine teaching, therefore, serves to break down a programmatic barrier between machine learning expertise and process experts.

Machine teaching relies on the fundamental tendency of teachers to break down tasks into concepts, skills, and strategies, then orchestrate the practice and execution of these skills into a meaningful sequence. In this way, we merge attributes of expert systems with DRL to provide some of the explainability of the expert system with the adaptability and creativity of learning AI.

We borrow the top-level rules of expert systems to provide the structure of the system or process's decision tree, but we replace the second layer of rules, which provide the instructions about how to navigate the decision tree, with DRL agents that learn strategies. The decision tree is the teaching layer that describes what skills and strategies the AI will learn, and the neural network neurons comprise the machine learning layer. In this way, the structure provides some of the explainability and predictability of expert systems with the creativity and flexibility of DRL agents.

Let's return to the example of the gyratory crusher above. The structure of the expert system, which reflects the two operating modes of the machine, outlines three skills that should be taught and learned. The first skill is the strategy of choking the crusher when the mine produces larger, harder rocks. The second skill is the strategy of regulating the crusher when the mine produces smaller, softer rocks. The third skill decides when to choke the crusher and when to regulate the crusher. This act of using subject matter expertise to define these three skills is itself teaching. Then, if we train each of three separate DRL agents on one of the three skills above, the combined brain will not only tell the engineers which next action to take to control the crusher but also which skill it is using at each decision point to make that decision.

Learning Agent vs. Expert System

Modular AI Captures the Best of Both Worlds



Savant, self-learning agents master creative solutions to control problems but are a **black box**.

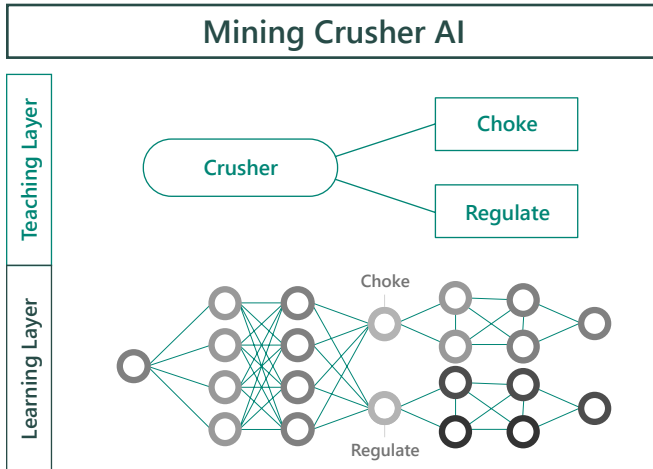


Modular AI provides some of the **explainability** of rule systems *and* the **creativity** of the learning agent.



Rules and expert systems are predictable, maintainable, and explainable but **completely static**.

The next age of useful AI requires teaching intelligence



We assert that teaching is required for and will usher in the next age of useful AI. Alan Turing opened the door to the second age of machine intelligence by suggesting that instead of building a new machine to provide intelligence for each task, [a master machine could be programmed with different algorithms](#). Before that, each example of machine intelligence from the steam regulator to automatons to player pianos to the IBM voting machine required an entirely different machine.

We have learned well from Turing, but now that learning algorithms exist, do we need to write a new learning algorithm for each new task? One insight offered from Microsoft Project Bonsai is that most of the AI that we have taught to successfully make useful decisions in real-world applications was built from the same small handful of learning algorithms. A new algorithm wasn't required for each new feat of intelligence, but a teacher was. This is why we believe that teaching, not programming, will usher in the next age of useful AI.

Era of Intelligence	Scope of Intelligence	Examples
Machine Intelligence	Build a new machine to provide intelligence for each task.	Automaton, IBM Voting Machine
Algorithm Intelligence	Write a new algorithm to provide intelligence for each new task.	Turing Computer
Teaching Intelligence	Teach a learner each new task.	AlphaGo



Solving real-world problems with AI

The Microsoft Project Bonsai platform presents many of the innovations discussed in this white paper, including DRL and an emphasis on machine teaching. With it, experts in various fields can utilize low-code approaches to teach a neural network brain to make decisions that control processes and equipment through complex scenarios.

Many Microsoft Partners are already working with Project Bonsai to improve their day-to-day operations:

1 **PepsiCo** makes several kinds of food and drink products, including delicious Cheetos. Because food production relies heavily on precise manufacturing systems, PepsiCo is always interested in optimizing those systems for efficiency and cost savings.

PepsiCo recently partnered with Microsoft for a pilot plant utilizing Project Bonsai autonomous solutions. This solution reads data from the system gathered by sensors and makes decisions about optimizing consistency, length, and other specifications for Cheetos. This brain, which was certified as a medium expert operator, adjusts to changing conditions that cannot be measured (such as variations in cornmeal moisture) to make more accurate and uniform Cheetos from that production line.

2 **Petrochemical company SCG** also utilizes a Project Bonsai solution to connect their engineers to optimizing systems. The author interviewed expert engineers to design an AI and build a curriculum to teach that AI the same two skills that each operator at the plant is trained in. The AI practiced using these two skills across plant variations such as changing the composition of input reagents and different catalysts. With open machine teaching and low-code solutions, Project Bonsai allowed engineers at SCG and the AI to work together to learn about the system and make changes to optimize production. The result was an expert operator AI that could control the reactor and reduce the calibration time of the reactor simulation from six months to two weeks.

3 **Sberbank**, one of the largest Russian and Eastern European banks, implemented a robotic arm to handle heavy coin bags that were a common part of their operations. Each coin bag weighed over four pounds, and it was difficult to clear coin bags from deep carts. Sberbank worked with Microsoft to use the Project Bonsai platform to act as the brain for a mechanical arm that could automatically remove the coins without the need for a human operator. The brain succeeded in this difficult grasping task 97% of the time.

Additionally, Tesla has been utilizing AI and machine learning for its fleet of self-driving electric cars. The Autopilot AI is one of the key components of Tesla's cars. While not much is known about the AI itself, it is common knowledge that the deep learning algorithms are also informed by crowdsourced data collected from all Tesla vehicles and then stored in the cloud. This data informs the underlying neural nets of the machine learning brains to essentially create their spatial location and response strategies.

| Conclusion

AI and machine learning are an evolution of thought, practice, and implementation not only of machines but also of how we envision what learning and thinking are. The invention of the neural network, DRL methodologies, and the rise of big data and powerful cloud computers have ushered in a new AI boom where useful AI is innovating several major industries.

Perhaps more importantly, this new age of AI has introduced researchers and professionals to a new way to teach machines. With innovative no-code or low-code solutions, experts in any field can feasibly teach a machine brain how to perform specific functions.

Bridging the gap between experts and AI is a critical next step in the evolution of machine learning and AI as a useful solution to real-world problems.

Finally, as experts find new ways to teach and deploy AI, the possibilities of what machines can think, learn, and do become almost limitless.

[Learn more about Microsoft Autonomous Systems →](#)



About the Author

Kence Anderson is Principal Program Manager, Machine Teaching Innovation for Autonomous Systems at Microsoft. Kence has designed over 100 autonomous decision-making AI systems for commercial uses, including at PepsiCo, Bell Flight, and Shell. Kence is the son of a master teacher and is trained in mechanical engineering, and he utilizes both aspects by researching how the principles of teaching combined with human expertise can be used to design and build useful AI.