AAAI Presidential Address:

Steps Toward Robust Artificial Intelligence

Thomas G. Dietterich

■ *Recent advances in artificial intelligence are encouraging governments and corporations to deploy AI in high-stakes settings including driving cars autonomously, managing the power grid, trading on stock exchanges, and controlling autonomous weapons systems. Such applications require AI methods to be robust to both the known unknowns (those uncertain aspects of the world about which the computer can reason explicitly) and the unknown unknowns (those aspects of the world that are not captured by the system's models). This article discusses recent progress in AI and then describes eight ideas related to robustness that are being pursued within the AI research community. While these ideas are a start, we need to devote more attention to the challenges of dealing with the known and unknown unknowns. These issues are fascinating, because they touch on the fundamental question of how finite systems can survive and thrive in a complex and dangerous world.*

Let me begin by acknowledging the recent death of Marvin Minsky. Professor Minsky was, of course, one of the four authors of the original Dartmouth Summer School proposal to develop artificial intelligence (McCarthy et al. 1955). In addition to his many contributions to the intellectual foundations of artificial intelligence, I remember him most for his iconoclastic and playful attitude to research ideas. No established idea could long withstand his critical assaults, and up to his death, he continually urged us all to be more ambitious, to think more deeply, and to keep our eyes focused on the fundamental questions.

In 1959, Minsky wrote an influential essay titled Steps Toward Artificial Intelligence (Minsky 1961), in which he summarized the state of AI research and sketched a path forward. In his honor, I have extended his title to incorporate the topic that I want to discuss today: How can we make artificial intelligence systems that are robust in the face of lack of knowledge about the world?

Minsky shared this concern. In his book, *Society of Mind* (Minsky 1988) and in many interviews, he often

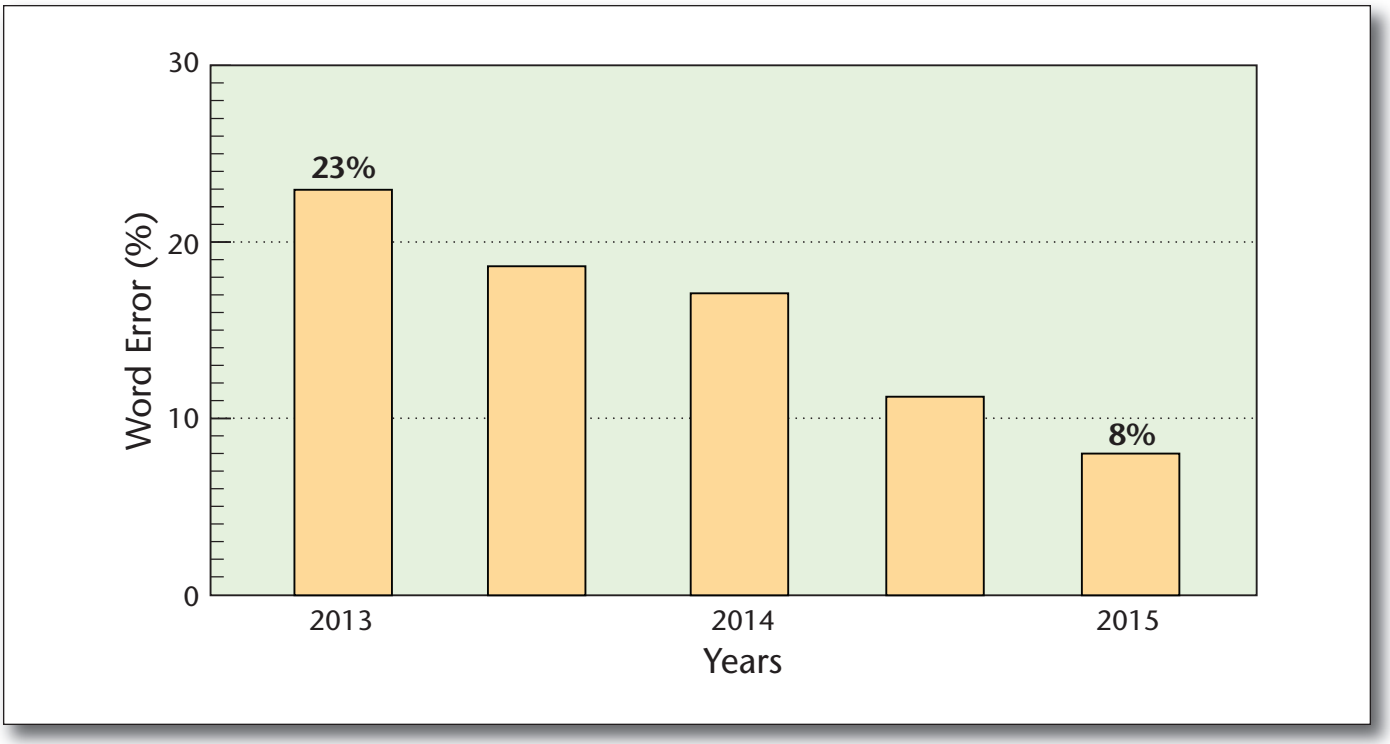

Figure 1. Google Speech Word Error Rate.

(Reproduced with permission from Fernando Pereira and Matthew Firestone, Google)

pointed out the contrast between the robustness of the human intellect and the brittleness of existing AI systems. In an interview with John Brockman, he said

> almost any error will completely paralyze a typical computer program, whereas a person whose brain has failed at some attempt will find some other way to proceed. We rarely depend upon any one method. We usually know several different ways to do something, so that if one of them fails, there's always another. (Brockman [1996] p. 156)

In this article, I wish to address this question: As a field, what are our current ideas about how to achieve robustness in AI systems? I will begin by arguing that we need methods for robust AI because of emerging applications of AI in high-stakes applications where human lives are at risk. I will then discuss the two main settings in which to consider robustness: Robustness to the *known unknowns* (that is, robustness to aspects of the world for which we have models) and robustness to the *unknown unknowns* (that is, robustness to unmodeled aspects of the world). I will then describe four approaches to robustness within each of these settings.

Why We Need Robust AI

Recent advances in AI are encouraging and enabling new high-stakes AI applications. My argument is that AI technology is not yet sufficiently robust to support these applications. Let me first review some of the recent progress and then discuss these emerging applications.

The past decade has witnessed exciting advances in artificial intelligence research and applications. Figure 1 shows that the word error rate of the Google speech engine has declined dramatically from 23 percent in 2013 to 8 percent in 2015 (Fernando Pereira and Matthew Firestone, personal communication). Figure 2 shows similar progress in computer vision for the task of determining whether an image contains an instance of an object class (for 1000 possible classes). The top-5 error rate has dropped from 28.2 percent in 2010 to 6.7 percent in 2014 (Russakovsky et al. 2015). There have been similar advances in other computer vision tasks such as object localization, recognizing text in images (for example, signage), and image captioning. Turning to language processing, the progress on natural language translation has been substantial. A standard automated metric for translation quality is the bilingual evaluation understudy (BLEU) score. A typical BLEU score in 2007 was 14.6. In 2014, scores in the range of 23.6–24.7 were attained by several groups (Sennrich 2016). Another way to assess translation accuracy is to have bilingual human judges assign a score from 0 (nonsense) to 6 (perfect translation) to a pair of sentences. Figure 3 from Google compares such scores for various language pairs and translation methods: (a) phrase-based translation, (b) neural machine translation, and (c)

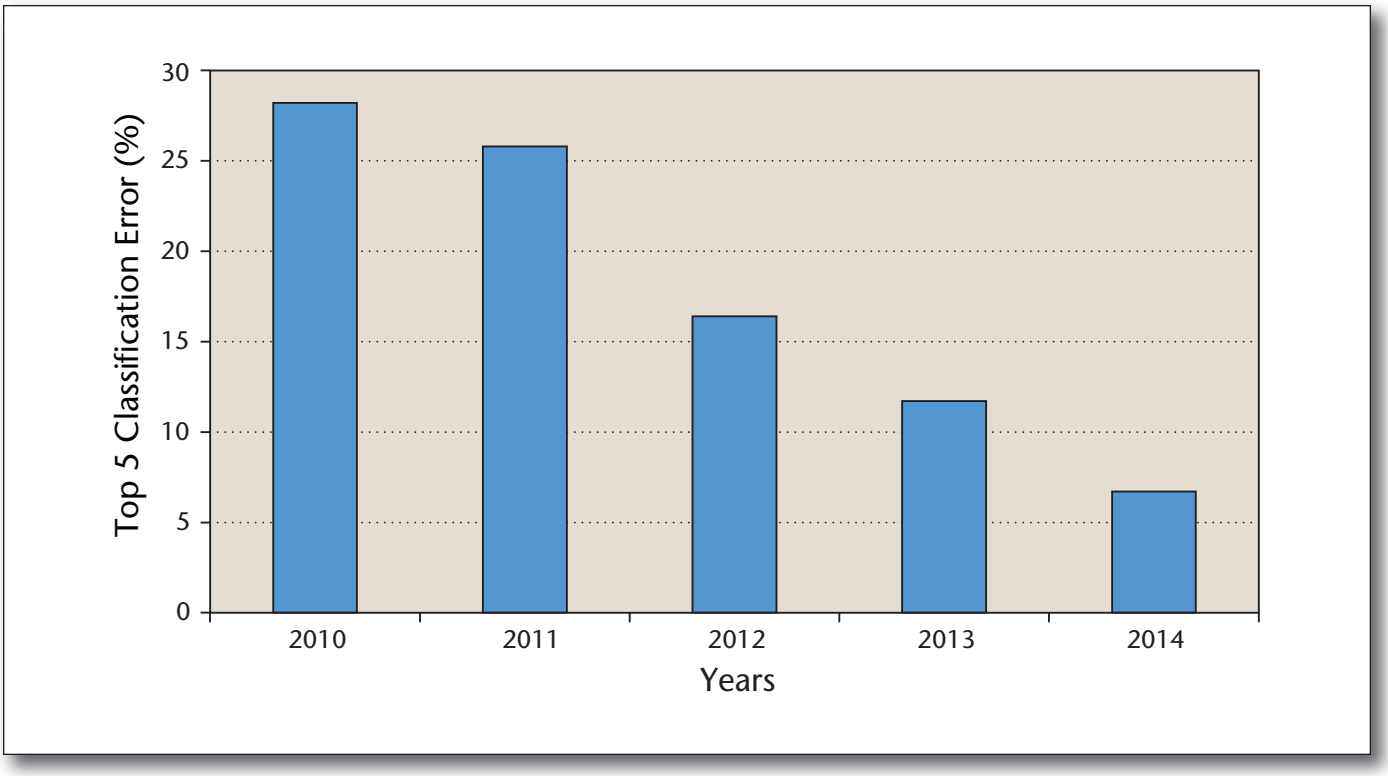

Figure 2. ImageNet Top 5 Classification Error Rate.

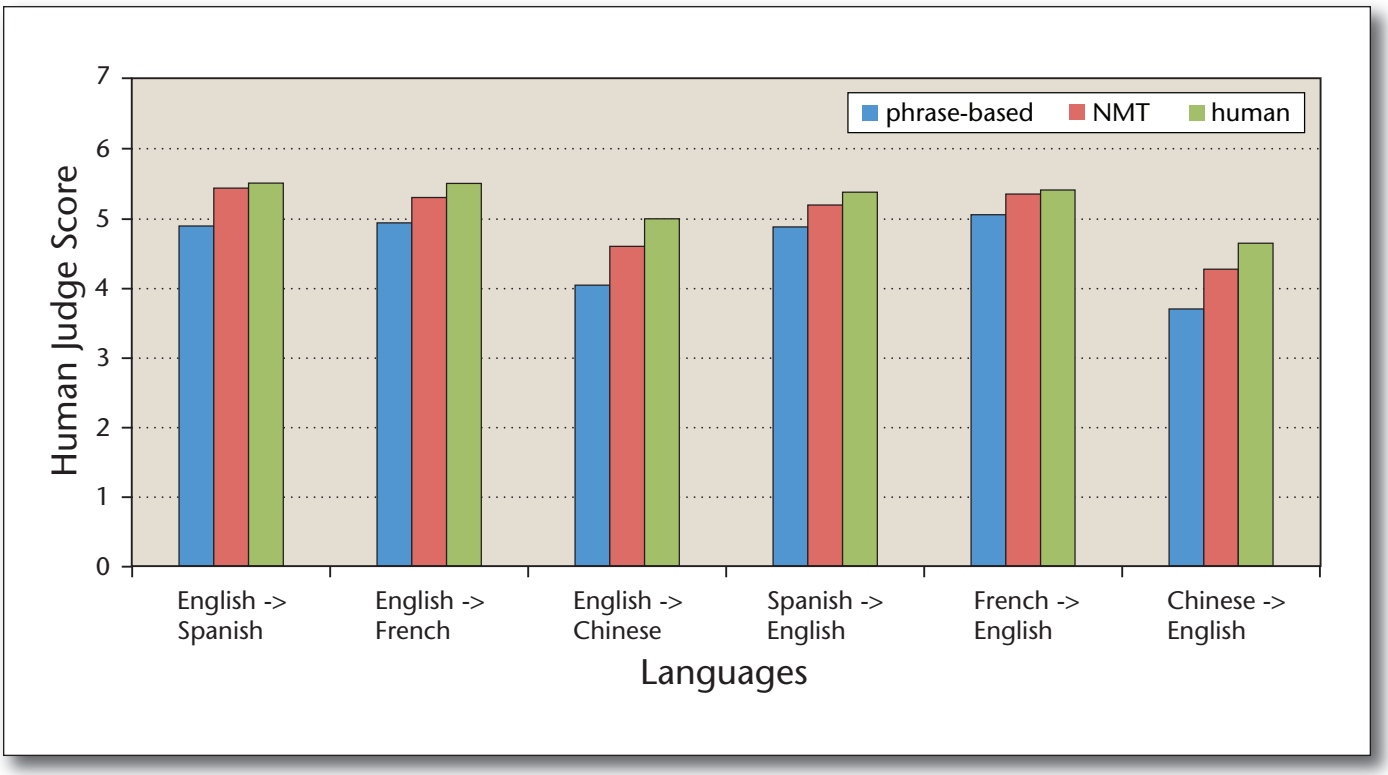

Figure 3. Human Evaluation of Google Translate Quality.

Figure 4. Google Real-Time Image Translation.

(Courtesy, BBC.com)

human translation (Wu et al. 2016). We see that neural machine translation has greatly reduced the gap between AI methods and human translators, although a substantial gap in quality remains. In all of these cases, the advances are due to improvements in deep neural networks for machine learning. The separate advances in vision, speech, and translation can be combined in exciting ways. For example, by combining speech recognition and translation, Microsoft now offers Skype Translator, which provides real-time speech-to-speech translation in Skype. By integrating computer vision (for recognizing text) with translation, Google offers live image translation, as shown in figure 4.

In addition to progress enabled by machine learning, many improvements in AI systems have resulted from advances in reasoning methods. Figure 5 shows how the size of satisfiability problems that can be solved in a fixed amount of time improved 1000-fold from 1998 to 2010. Note in particular that these advances resulted from many different algorithm innovations. Satisfiability solvers (especially, solvers for satisfiability modulo theories; SMT) are now being widely deployed for model checking in hardware and software verification. In addition, areas of AI such as automated planning and scheduling have made progress by formulating their tasks as satisfiability problems and solving them with these improved algorithms.

Game playing is another area where AI continues to make progress. Figure 6 plots the performance of computer chess and Go programs as a function of time (Schaeffer, Müller, and Kishimoto 2014). Performance on chess has continued to improve beyond the level achieved by IBM's Deep Blue when it defeated Gary Kasparov in 1997. Progress in the game of Go was very slow until the development of the UCT algorithm for Monte Carlo tree search (Kocsis and Szepesvári 2006). This ushered in a period of rapid progress, but it still did not lead beyond master-level play. In 2015–16, the researchers at Google DeepMind developed AlphaGo, which combined deep neural network methods with Monte Carlo tree search. The neural networks were apparently able to solve the pattern perception problem, which was long believed to be a key to human skill in the game. By combining this with search, AlphaGo was able to defeat Lee Sedol, one of the top players in the world.

Beyond games of perfect information, such as Go and chess, AI has made steady progress in games of imperfect information, such as poker. In such games, one measure of difficulty is the number of possible

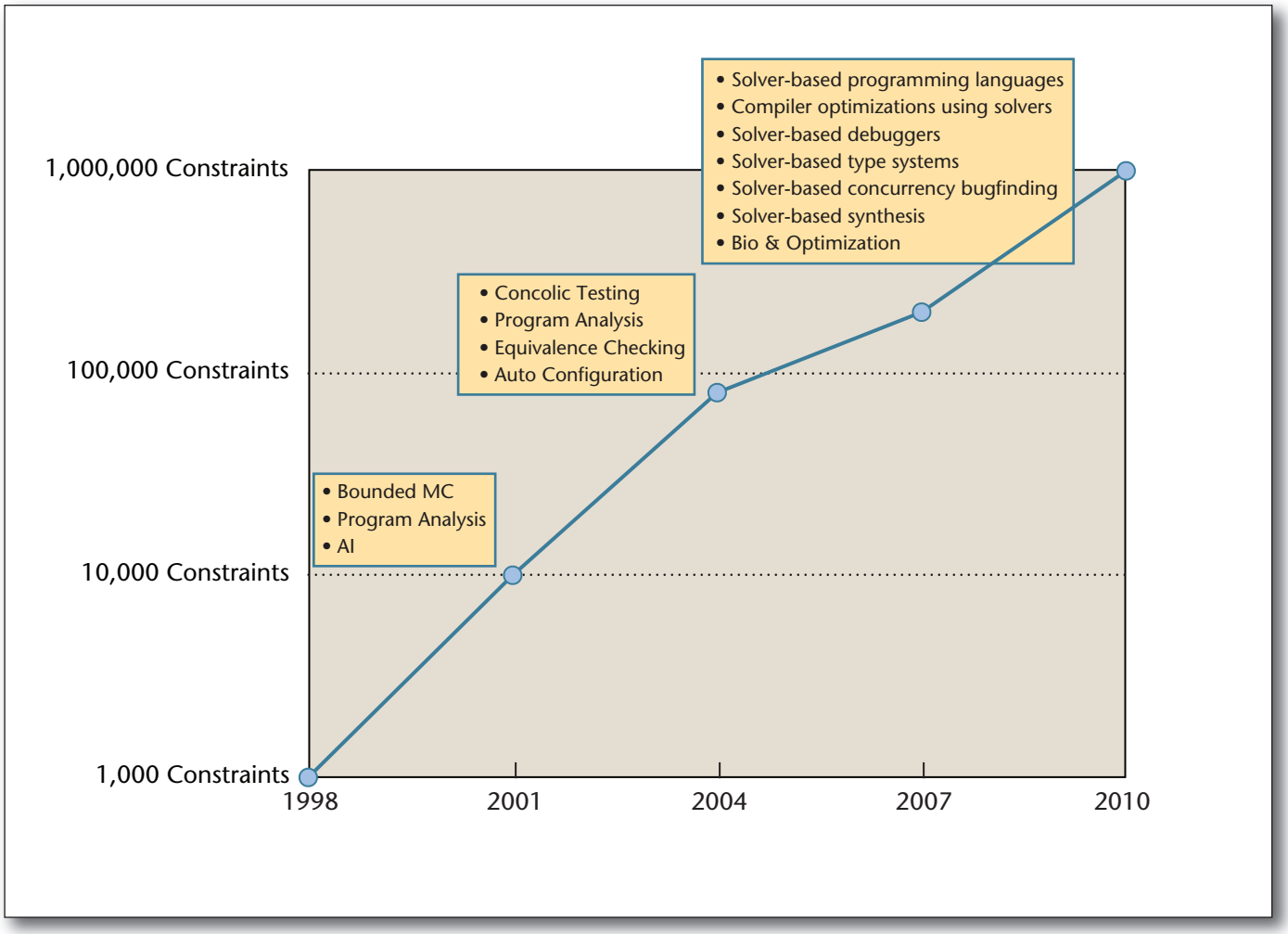

Figure 5. Progress on Satisfiability Algorithms.

(Courtesy, Vijay Ganesh.)

information sets. In poker, an *information set* is the information about the cards that are face up. Figure 7 reports the number of information sets that can be correctly processed as a function of time. There has been an increase of eight orders of magnitude between 2003 and 2014, although three of these can be credited to improvements in computer hardware (Bowling et al. 2015). In 2017, a program named Libratus from Tuomas Sandholm's lab at Carnegie Mellon University convincingly defeated four of the world's top 10 players in Heads-Up No Limit Texas Hold'em. This version of poker has 10^{161} information sets. Libratus combines deep neural networks with many search techniques. Unlike the work reported in figure 7, it does not provide a guaranteed solution to the full game. Nonetheless, Libratus clearly demonstrates the power of combining automated reasoning and machine learning to solve difficult problems of reasoning with imperfect information.

All of these advances in perception, learning, and reasoning have led to a huge increase in applications of AI. Many of these, such as personal assistants (Siri, GoogleNow, Alexa, and others), advertisement placement, and recommendation engines, operate in very low-stakes settings where errors are easily managed. However, several emerging applications involve potential loss of human lives or catastrophic financial and infrastructure disruption. Let us list some of these applications.

The application that has received the most attention in the media is the development of self-driving cars. These are enabled by advances in computer vision and sensor fusion as well as significant cost reductions in sensing (for example, LIDAR, RADAR). Autonomous cars have the potential to greatly reduce the loss of human lives due to human error. But we know that computer vision systems based on machine learning sometimes make errors that no human would make. How many such errors will society be willing to tolerate?

A second application area is robotic surgical assistants. For several years, teleoperated robots have

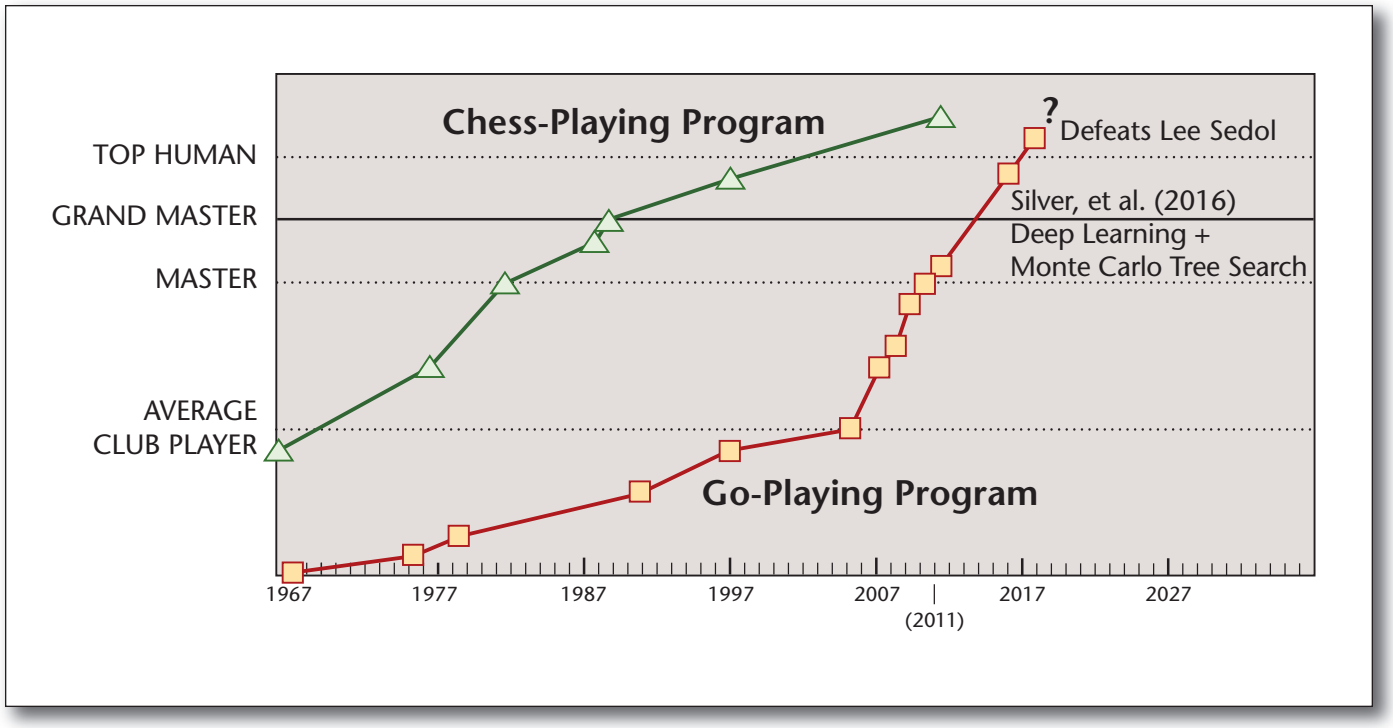

Figure 6. Progress on Chess and Go.

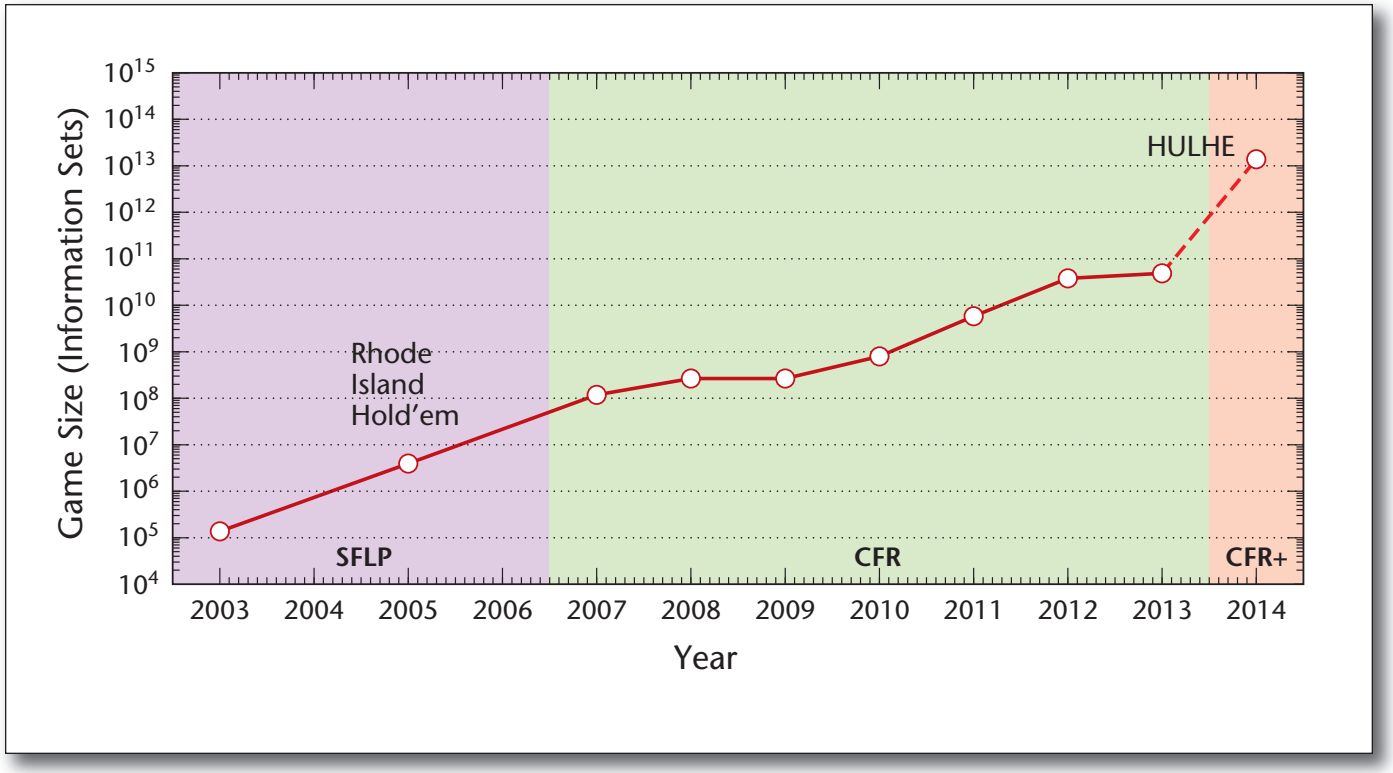

Figure 7. Progress on Computer Poker.

Courtesy Science Magazine.

helped surgeons perform delicate operations by scaling down and steadying the movements of their hands. But advances in perception and control are now encouraging the near-complete automation of certain subtasks such as LASIK eye procedures, knee surgeries, and even soft-tissue suturing (Shademan et al. 2016, Solis 2016). The potential for improved medical outcomes is great, but flaws in perception, reasoning, and execution could place human lives at risk.

Automated stock trading is a third area where AI methods are being applied. This application illustrates one of the key advantages of automation: it can operate at much greater speeds than human traders. This is also a key vulnerability. It is impossible for humans to monitor each trade and intervene to prevent errors. We have already observed cases of market instability where automated trading magnified market swings (Kirilenko et al. 2017).

A fourth high-stakes application is AI control of the electrical power grid. AI methods have the potential to manage the increasingly complex task of integrating renewable energy sources, such as wind and solar, with steadier generating methods, such as hydropower and nuclear power (Gopakumar, Reddy, and Mohanta 2014). Reinforcement learning algorithms have been developed that can better manage hydroelectric systems to improve the health of fish stocks (Grinberg, Precup, and Gendreau 2014). Monte Carlo tree search methods are being developed to respond rapidly to equipment failures and prevent large-scale blackouts (Eduardo Cotilla-Sanchez, personal communication). The potential benefits of these AI applications are large, but the risks, if these systems make mistakes, are also large.

The final high-stakes application that I wish to discuss is autonomous weaponry. Unlike all of the other applications I've discussed, offensive autonomous weapons are designed to inflict damage and kill people. These systems have the potential to transform military tactics by greatly speeding up the tempo of the battlefield. Like high-speed trading systems, this in itself poses grave risks unless human commanders can keep up with the faster pace. Without meaningful human control, some people argue that such systems will violate the laws of war (Human Rights Watch 2016). Notwithstanding that concern, advocates argue that battlefield robots will be better able to obey the Geneva conventions on the laws of war, because they will not be overcome with emotions and stress (Arkin 2009). Finally, some analysts are concerned that any flaw in the AI systems could lead a battlefield robot to attack the wrong targets. If that flaw is replicated across an entire robot force, the results could be devastating (Scharre 2016). My view is that until we can provide strong robustness guarantees for the combined human-robot system, we should not deploy autonomous weapons systems on the battlefield. I believe that a treaty banning such weapons would be a safer course for humanity to follow.

All of these high-stakes applications require robust artificial intelligence technology. There are at least five aspects of robustness that require attention. First, systems need to be robust to errors committed by their human operators. In robotic surgery, weapons systems, and, possibly, in self-driving cars, the human is "in the loop" and the system is therefore a joint human-computer agent. Second, high-stakes systems must be robust to misspecified goals. This is a particularly serious form of human error in which the human gives a command, for example, "Get to the airport as soon as possible" (given to a self-driving car), that if interpreted literally could involve breaking laws, injuring pedestrians, and even killing the occupants of the car. Third, high-stakes systems need to be robust to cyberattack. In universities, cyber security is studied separately from AI, but when AI systems wield control of highway networks, power grids, financial markets, and weapons systems, they become attractive targets for cyberattacks. Hence, cyber security must become an integral part of the design of AI systems. Fourth, AI systems need to be robust to errors in their models of the world—that is, to places where their models are explicitly incorrect. Finally, AI systems need to be robust to unmodeled aspects of the world. I am particularly interested in this last form of robustness, because even if we address all of the others, we will still be confronted with the problem of the unknown unknowns.

Although all five aspects are important, in this article, I will focus on only the last two. Let me motivate these a bit more. It is easy to understand why an AI system must be robust to errors in its models. We all have experience debugging models, and we know that it is practically impossible to eliminate all of the errors, particularly if we consider errors of precision (for example, positional uncertainty in robots).

The importance of unmodeled phenomena for robustness is less obvious. Why can't we just build complete models? There are two reasons. First, it is impossible to model everything. AI researchers long ago expressed this in terms of two named problems: the qualification problem and the ramification problem. The qualification problem formalizes the realization that it is impossible to enumerate all of the preconditions of an action. For example, we can write down many conditions that must hold in order for a car engine to start: sufficient fuel, sufficient battery power, correct functioning of each of the components, and so on. But we must also consider such famous preconditions as "There is no potato in the tail pipe." Symmetrically, the ramification problem considers the opposite direction: it is impossible to enumerate all of the consequences of an action.

Even if it were not impossible to model everything, it would not be desirable. Consider the theory of machine learning, roughly summarized by the equation

$$\text{error rate} \propto \frac{\text{model complexity}}{\text{training data size}}.$$

Because a model of every aspect of the world would be extremely complex, this equation tells us that to learn the parameters of such a model, we would need an extremely large set of training data.

These arguments drive us to the conclusion that every AI system will need to act without having a complete and correct model of the world.

Digression One: Uncertainty and the History of AI

Before we explore methods for achieving robust AI systems, let's pause for a moment to consider the role of uncertainty in the history of AI. We can divide this history into three periods. The period from 1958 to 1984 can be called the period of the known knowns. AI research focused on reasoning and search: methods for theorem proving, planning in deterministic, fully observed worlds (the blocks world), and games of perfect information (checkers and chess). Such fully known worlds are not devoid of uncertainty, but the uncertainty can be resolved by deeper search and additional reasoning. The uncertainty is a consequence of incomplete computation rather than lack of knowledge (Dietterich 1986).

Beginning around 1980, AI researchers started attacking applications, such as medical diagnosis, in which observations (for example, symptoms, lab tests) are processed to make uncertain inferences about hidden variables (such as diseases). The field of uncertainty in AI was founded, and Judea Pearl and colleagues developed practical ways to deploy probability theory to represent uncertainty about the values of large sets of variables (Pearl 1988). This period, from 1980 to the present, could be called the period of the known unknowns. The dominant methodology is to identify those variables whose values are uncertain, define a joint probability distribution over them, and then make inferences by conditioning on observations. A wave of textbooks have been published with titles such as *Probabilistic Graphical Models* (Koller and Friedman 2009), *Probabilistic Robotics* (Thrun, Burgard, and Fox 2005), *Machine Learning: A Probabilistic Perspective* (Murphy 2012), and, of course, *Artificial Intelligence: A Modern Approach* (Russell and Norvig 2009). Recently, probabilistic programming languages have been developed to make it easy to define and reason with highly complex probabilistic models (Gordon et al. 2014, Pfeffer 2016).

I believe we have now entered a third period of AI—the period of the Unknown Unknowns. In this period, we must develop algorithms and methodologies that enable AI systems to act robustly in the presence of unmodeled phenomena.

Digression Two: Robustness Lessons from Biology

In computer science, an important paradigm for analyzing and solving problems is to formulate them in terms of optimization. By stating the optimization objective, we gain clarity about what counts as a solution, and we can prove guarantees on the correctness of our systems. Examples abound. In machine learning, we often seek maximum likelihood estimates for the parameters in our probabilistic models. In perception, we wish to estimate the depth of each pixel in an image or the most likely sequence of words spoken by a person. In planning, we seek the optimal plan or the plan that maximizes the expected cumulative discounted reward.

However, the optimization paradigm is not robust: it assumes that the optimization objective is correct. The optimum is often attained on the boundary of the feasible region (such as in linear programming) — precisely where the model is most likely to be incorrect. In machine learning, for example, maximizing the likelihood is well known to cause overfitting and result in poor predictive performance.

In biological evolution — in contrast — natural selection can be seen to select organisms that survived threats from a complex and uncertain environment. The internal models that those organisms might possess are certainly not complete and may not even be particularly accurate, but the organisms are robust. Evolution does not optimize an objective; it does not necessarily lead to increases in complexity or intelligence. Instead, it can be viewed as optimizing robustness (Kitano 2004, Whitacre 2012, Félix and Barkoulas 2015).

Biology also relies on maintaining diverse populations of individuals. This can be viewed as a "portfolio" strategy for robustness, much along the lines that Minsky suggested in my opening quotation. Even within individuals, we often find redundancy. Many organisms have multiple metabolic pathways for producing critical molecules. Each of us has two copies of our genes, because (with the exception of the sex chromosomes of males), we carry two of each chromosome. This allows recessive alleles to be passed on to future generations even though they are not expressed in the current one.

Finally, biological organisms disperse spatially, which confers robustness to spatially localized disturbances such as droughts, fires, landslides, and diseases.

Perhaps biology has lessons for us as we seek to create robust AI systems?

Approaches to Robust AI: The Known Unknowns

The goal for the remainder of the article is to make an inventory of ideas within the AI community for improving the robustness of our systems. I will begin

by discussing four ideas for improving the robustness of AI systems when dealing with the known unknowns. These four ideas can all be viewed as incorporating robustness into the optimization paradigm. Then I will discuss four ideas for addressing robustness in the face of the unknown unknowns.

Idea 1: Robust Optimization

Consider the standard problem of linear programming. Suppose our goal is to maximize a linear objective function $J(x_1, x_2)$ over two variables x_1 and x_2 subject linear inequality constraints:

$$\max_{x_1,x_2} J(x_1,x_2)$$

subject to

$$ax_1 + bx_2 \le r \text{ and } cx_1 + dx_2 \le s$$

As I mentioned previously, we know that the optimum will be located on the boundary of the feasible region. Figure 8 shows a typical example; the objective function is increasing in the direction of the arrow, the constraints are shown as lines, and the feasible region is shaded. The optimum is located at the vertex where the two constraint lines intersect.

Suppose we are uncertain about the parameters in the constraint equations (a, b, c, d, r, and s). One approach to formulating a robust optimization problem is to define uncertainty regions for each parameter such that $a \in U_a$, $b \in U_b$, …, $s \in U_s$. We can then formulate the minimax optimization problem

$$\max_{x_1,x_2} \min_{a,b,c,d,r,s} J(x_1,x_2;a,b,c,d,r,s)$$

subject to

$$ax_1 + bx_2 \le r \text{ and } cx_1 + dx_2 \le s$$

and $a \in U_a$, $b \in U_b$, …, $s \in U_s$

We can view this as a game in which an adversary chooses the values of the parameters (within the constraint regions) in order to minimize the quality of the optimal solution. While this does confer robustness, it often results in very poor solutions, because the adversary can create a devastatingly bad worst case.

An important idea in robust optimization is to impose a budget on the adversary. Let us reformulate the uncertainty regions so that they define perturbations. For example, let $a + \delta_a$ be the perturbed value of the a parameter, $b + \delta_b$ be the perturbed value of the b parameter, and so on. Then require that $\delta \in U_a$, $\delta_b \in U_b$, …, $\delta_s \in U_s$. We can define the budgeted adversary minimax problem as

$$\max_{x_1,x_2} \min_{\delta_a,\ldots,\delta_s} J(x_1,x_2;\delta_a,\ldots,\delta_s)$$

subject to

$$(a+\delta_a)x_1 + (b+\delta_b)x_2 \le (r+\delta_r)$$

$$(c+\delta_c)x_1 + (d+\delta_d)x_2 \le (s+\delta_s)$$

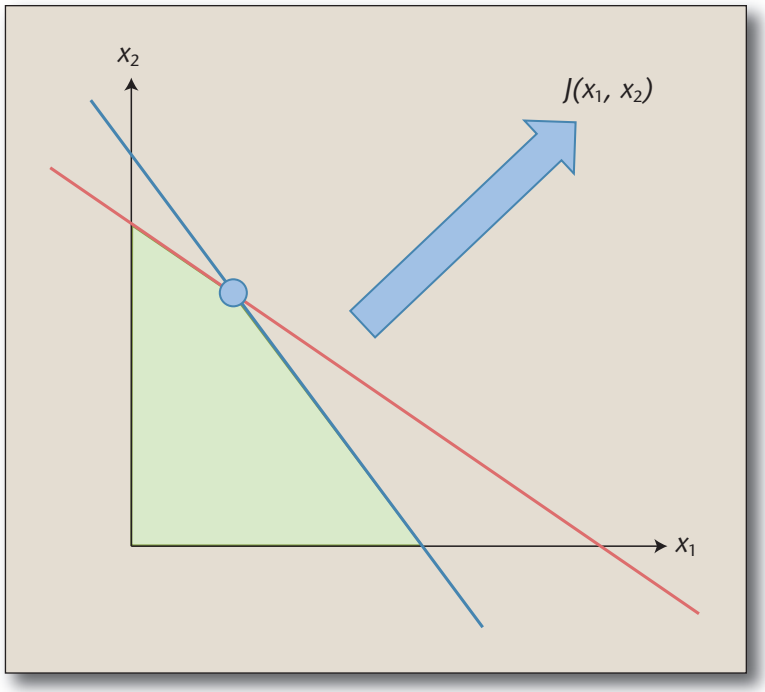

Figure 8. A Simple Linear Programming Problem.

$$\sum_i |\delta_i| \le B$$

$$\delta_a \in U_a,\ \delta_b \in U_b, \ldots, \delta_s \in U_s.$$

The constant B is the total perturbation budget given to the adversary. By solving the problem for various values of B, we can map out the trade-off between the value of the objective J and the robustness of the solution. If the uncertainty regions are convex and defined by linear constraints, then this problem is still a linear program, so it can be easily solved (Bertsimas and Thiele 2006). The idea of robust optimization — or more generally, the idea of optimizing against an adversary — is broadly applicable and will connect all four ideas of this section.

Idea 2: Regularization in Machine Learning

Consider the standard supervised learning problem. We are given a collection of training examples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$, where each x_i is an input feature vector and y_i is the corresponding desired output (usually a class label or real-valued response). We also specify a hypothesis class $\mathcal{H}$ such as the class of linear separators (for support vector machines) or more complex classes such as decision trees or neural networks. Our goal is to find a hypothesis $h \in \mathcal{H}$ such that the predicted value $\hat{y}_i = h(x_i)$ is close to the observed value y_i. We measure "closeness" through a loss function $L(\hat{y}, y)$ that quantifies how bad it is to predict $\hat{y}$ when the true value is y.

With these preliminaries, the problem of empirical risk minimization can be expressed as

$$\text{find } h \in \mathcal{H} \text{ to minimize } \sum_i L\left(h(x_i), y_i\right).$$

When the hypotheses are conditional probability distributions $h(y|x) = \Pr(y|x)$ and the loss function is $-\log h(y_i|x_i)$, then empirical risk minimization becomes maximum likelihood estimation.

The main weakness of empirical risk minimization is that if the hypothesis space $\mathcal{H}$ is highly expressive, then the function h that works best on the training data will often work poorly on new data points. The problem is that in an expressive hypothesis space there are usually functions that can essentially memorize the training data without generalizing well.

A widely adopted solution to this problem is to define a measure $\|h\|$ of the "complexity" of h. For example, if h is defined by a set of parameters (for example, coefficients of a linear function, weights of a neural network), then $\|h\|$ might be defined as the sum of the squares of these coefficients. We then define the following complexity-regularized optimization problem

$$\text{find } h \in \mathcal{H} \text{ to minimize } \sum_i L\left(h(x_i), y_i\right) + \lambda \|h\|,$$

where λ is the regularization parameter. If we set $\lambda = 0$, then we recover the maximum likelihood solution. As λ increases, the optimal h will be forced to become progressively less complex. In the limit $\lambda \to \infty$, all of the coefficients are forced to be zero. The value of λ is usually determined using a separate set of validation data (or through the technique of cross-validation).

Regularization is the key to preventing overfitting, and virtually all applications of machine learning employ some form of regularization. We can view the regularization penalty as a force that "pulls the solution back" from the unpenalized optimum.

Interestingly, one intuitive definition of overfitting is that a hypothesis h has overfit the training data if the loss sharply increases when we perturb the training examples: $L(h(x_i + \delta_i), y_i) \gg L(h(x_i), y_i)$. Equivalently, we can measure the capability of a hypothesis h to generalize to new points in terms of how stable its predictions are in the presence of perturbations. Recent research by Shie Mannor and his collaborators (Xu, Caramanis, and Mannor 2009) has shown that this intuition can be formalized. Specifically, for the case of the linear support vector machine (where L is the so-called hinge loss), the regularized optimization problem is equivalent to the following robust optimization problem

$$\min_{h \in \mathcal{H}} \max_{\delta_1, \ldots \delta_N} \sum_i L\left(h(x_i + \delta_i), y_i\right) \text{ subject to } \sum_i \|\delta_i\| \le \lambda$$

in which the parameter λ is equal to a perturbation budget given to the adversary and $\|\delta_i\|$ is the Euclidean distance of the perturbation δ_i.

In summary, regularization is an important technique for helping machine-learning algorithms to generalize well. For the case of the linear support vector machine, regularization is equivalent to robust optimization, and the parameter λ, instead of being an arbitrary parameter, turns out to be the perturbation budget given to an adversary.

Idea 3: Risk-Sensitive Objectives

Let us now turn to the problem of planning in Markov decision processes (MDPs). In an MDP, an agent is interacting with a fully observable world. At time t, the world is in state s_t. The agent observes that state and then chooses an action a_t to perform according to a policy function π: $a_t = \pi(s_t)$. When the agent performs action a_t, the world makes a stochastic state transition to a new state s_{t+1} according to the probability distribution $P(s_{t+1} \mid s_t, a_t)$. The agent receives a reward $R(s_t, a_t)$. This reward is a random variable, because it depends on all of the stochastic transitions up to time t. Let us consider the problem of finding a policy that optimizes the expected T-step reward starting from an initial world state s_0:

$$\text{find } \pi \text{ to maximize } J(\pi) = \mathbb{E}\left[\sum_{t=1}^{T} R\left(s_t, \pi(s_t)\right) | s_0\right].$$

As I have indicated here, the standard objective is to optimize the expected total reward, and the vast majority of work in MDP planning and reinforcement learning optimizes this (or closely related) objectives.

However, in some situations, one might be concerned about the down-side risk — that is, the risk that the actual reward received in a particular T-step trial will be very small. This is natural, for example, when investing for retirement. It is also a concern in problems involving endangered species where the expected total reward might be good even though the species goes extinct (a down-side risk) 25 percent of the time. In such cases, we seek a more conservative policy that may forgo large up-side outcomes in order to avoid down-side risk.

Many modifications of $J(\pi)$ have been studied that include some notion of risk or down-side risk. One of the best of these is the conditional value at risk (CVaR). To understand CVaR, imagine we have adopted a fixed policy π and that we can perform repeated trials in which we start in state s_0 and follow the actions of π for T steps. The T-step return V_T that we receive is a random variable:

$$V_T = \sum_{t=1}^{T} R\left(s_t, \pi(s_t)\right).$$

This random variable will exhibit some probability distribution resulting from the interaction of the policy π and the transition probabilities. Suppose this distribution is the black curve shown in figure 9a. Note that while the great majority of outcomes have large values, the distribution has a long tail of low returns. The conditional value at risk is controlled by

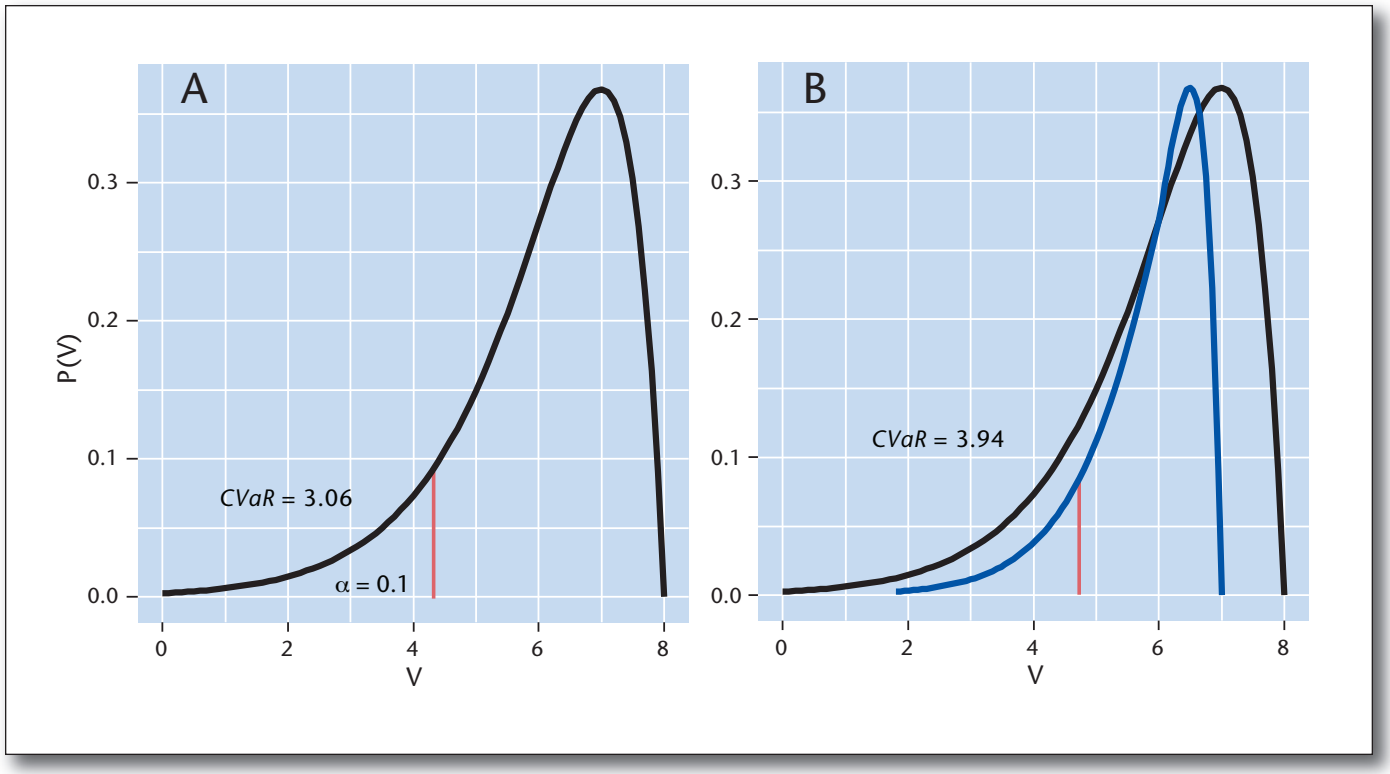

Figure 9. Conditional Value at Risk.

a parameter α that specifies a quantile in the distribution of returns. For α = 0.1, the red vertical line indicates this quantile. The CVaR is the expected value of all of the outcomes to the left of the red line — in this example, the 10 percent worst outcomes. The expected value of those outcomes for this distribution is 3.06. The CVaR objective seeks to maximize the expected value of these 10 percent worst outcomes. We search the space of policies to find the policy that maximizes this expectation.

A typical result is shown by the red curve in figure 9b. This is the distribution of V_T under the CVaR optimal policy. Again the red line marks the 10 percent quantile. The CVaR has improved to 3.94. Note that to achieve this we have sacrificed a significant amount of up-side reward.

It is interesting to ask the following question: Does acting conservatively (in the sense of CVaR) improve robustness to model error? Recent work also by Shie Mannor and his colleagues shows that the answer is yes. Optimizing CVaR is equivalent to solving a robust optimization problem in which an adversary is allowed to modify the transition probabilities.

Consider an adversary who at each time step t can choose a multiplicative perturbation δ_t and modify the MDP transition probabilities so that instead of making a transition from s_t to s_{t+1} with probability $P(s_{t+1} \mid s_t, a_t)$, the probability is changed to be $P(s_{(t+1)} \mid s_t, a_t) \cdot \delta_t$. To be more precise, let δ be a vector that specifies a multiplier, $\delta(s)$, for each possible state s. Then $P(s_{(t+1)} \mid s_t, a_t)$ is perturbed to be

$$\tilde{P}(s_{t+1}|s_t,a_t) := P(s_{t+1}|s_t,a_t)\cdot\delta_t(s_{t+1}).$$

We will place two constraints on the possible values of δ. First, the perturbed values $\tilde{P}$ must still be valid probability distributions. Second, the product of the perturbations along any possible trajectory $\langle s_1,\ldots, s_t, \ldots, s_T\rangle$ must be less than η:

$$\prod_{t=1}^{T}\delta_t(s_t) \le \eta \quad \forall\langle s_1,\ldots,s_t,\ldots,s_T\rangle$$

This is the "perturbation budget" given to the adversary. These two constraints interact to limit the extent to which δ values can become small (or even zero). This is because if $\delta_t(s) = 0$ for several states s, then $\delta_t(s')$ will be forced to become large for some other states s′, which will violate the η budget.

Let Δ be the space of all perturbations $\langle\delta_i, \ldots, \delta_T\rangle$ that satisfy these constraints. Then the robust optimization problem becomes

$$\text{find } \pi \text{ to maximize} \min_{\delta_1,\ldots,\delta_T \in \Delta} \mathbb{E}_{\tilde{P}}\left[\sum_{t=1}^{T} R(s_t,\pi(s_t))|s_0\right].$$

Chow et al. prove that this π is exactly the policy that maximizes the CVaR with quantile $\alpha = 1/\eta$.

In summary, the optimal risk-averse CVaR policy is also a policy that is robust to errors in the transition

probability model up to a total multiplicative perturbation of η. Acting conservatively confers robustness!

Idea 4: Robust Inference

In addition to robust optimization, robust learning, and robust decision making, several researchers have studied methods for robust inference.

One line of research is based on hierarchical Bayesian models. The central idea underlying the vast majority of contemporary work on the known unknowns is to represent our uncertainty in terms of a joint probability distribution. This can include treating the parameters of the joint distribution as hidden random variables and employing probability distributions to represent uncertainty about their values. These hierarchical models can be represented as standard probabilistic graphical models, although exact inference is rarely feasible. Fortunately, advances in Markov chain Monte Carlo methods now provide practical ways of sampling from the posterior distribution that results from conditioning on observations (Neal 1993; Gilks, Richardson, and Spiegelhalter 1995; Betancourt 2017). Such samples can be easily applied to make robust decisions (for example, based on conditional value at risk and other quantile-related measures).

A second line of research has studied extensions of probabilistic graphical models to capture sets of probability distributions. For example, credal networks (Cozman 1997; Cozman 2000) provide a compact method of representing convex sets of probability measures and then performing inference on them. Exact inference is generally intractable, but for restricted classes of credal networks, it is possible to define an efficient variable elimination algorithm (Antonucci and Zaffalon 2007).

One important application of probabilistic reasoning is in diagnosis, where the diagnostic system must iteratively decide which tests to perform in order to arrive at a diagnosis as quickly and cheaply as possible. One standard heuristic is to compute the expected value of the information (VOI) that will be gained through each candidate test and perform the test that maximizes the VOI. Adnan Darwiche and his collaborators have studied a robust version of this problem where they perform the test that is most likely to result in a diagnosis that is robust in the sense that further tests will not change the diagnosis (Chen, Choi, and Darwiche 2014, 2015).

Robustness to the Unknown Unknowns

What ideas does the AI research community have for creating AI systems that are robust to unmodeled aspects of the world? In this section, I will discuss four ideas that I am aware of. I expect there are others, and I hope we can extend this list as we do more research in this direction.

Idea 5: Detecting Model Failures

When an AI system's model is inadequate, are there ways to detect this prior to taking an action that could result in a serious error?

In machine learning, the model can fail when the distribution of training objects P_{train} and the distribution of test objects P_{test} (on which the learned model will be applied to make predictions) are different. Learning theory only provides guarantees when $P_{train} = P_{test.}$ There are many ways that the training and testing distributions can be different. Perhaps the setting that best illustrates this problem is open category classification. Let me describe it with an example from my own work.

To monitor the health of freshwater streams, scientists monitor the population of insects that live in these streams. In the United States, the Environmental Protection Agency conducts an annual randomized survey of freshwater macroinvertebrates belonging to the families of stoneflies, caddisflies, and mayflies. The specimens are collected using a kicknet and brought back to a laboratory where each insect must be identified to at least the level of genus. This is a time-consuming and tedious process that requires substantial expertise. Our research group at Oregon State trained a computer vision system to recognize the genus of a specimen from an image. We created a training set of images of 54 taxonomic groups covering most of the stoneflies, caddisflies, and mayflies found in the US Pacific Northwest. Figure 10 shows images of some of these taxa as captured by our photographic apparatus.

Evaluations on a separate test set showed good predictive accuracy. However, when we considered deploying this to process real kicknet samples, we realized that those samples would contain lots of other things beyond the 54 categories that our vision system had learned to recognize. There are often leaves and twigs, and there are also other species of bugs and worms. Following standard machine-learning practice, we had trained our system using discriminative training, which has been repeatedly shown to produce higher recognition accuracy than the alternative method of generative training. Discriminative training divides the image space into 54 partitions separated by decision boundaries. The result is that any possible image will fall into one of the 54 partitions and be assigned to one of the 54 insect categories that the system was trained to recognize. Hence, any image containing a leaf, twig, or bug belonging to a "novel" species would be guaranteed to be misclassified. One might hope that these novel items would fall near to the decision boundaries and, hence, result in lower-confidence predictions. This is sometimes true, but when we attempted to define a rejection rule — a rule for abstaining when the predictions have low confidence — the result was an equal error rate of more than 20 percent. That is, 20 percent of images from the 54 taxa were misclassified as novel, and 20

Figure 10. Images of Some Freshwater Macroinvertebrates.

percent of the images of novel objects were misclassified as belonging to one of the 54 taxa. This is unacceptably high.

Several research groups have been studying the problem of open category learning (Scheirer et al. 2013; Da, Yu, and Zhou 2014; Bendale and Boult 2015; Rudd et al. 2016; Steinhardt and Liang 2016). At Oregon State we have been experimenting with the architecture shown in figure 11 in which each input query x is first analyzed by an anomaly detector to compute an anomaly score $A(x)$. If $A(x)$ is greater than a specified threshold τ, the query is judged to be anomalous relative to the training examples and rejected. If the anomaly score is smaller than the threshold, then the trained classifier makes its prediction. We evaluated this method on the Letter Recognition task from the University of California, Irvine (UCI) machine-learning repository (Lichman 2013). We trained an isolation forest anomaly detector (Liu, Ting, and Zhou 2012) on the classes corresponding to the letters 'A' and 'B' and then measured how well it could detect that new examples belonged to these classes versus novel classes (the letters 'C' through 'Z'). Figure 12 plots an ROC curve for this problem. The dot corresponds to applying the method of conformal prediction (Shafer and Vovk 2008) to assess the confidence in the classifications. The ROC curve is significantly above and to the left of the dot, which indicates that the anomaly detector is able to do a better job. However, note that in order to achieve fewer than 5 percent missed alarms (novel objects incorrectly classified as known), we must suffer a false alarm rate of 50 percent (known objects rejected as being novel), so there is a lot of room for improvement.

One thing that makes open category classification

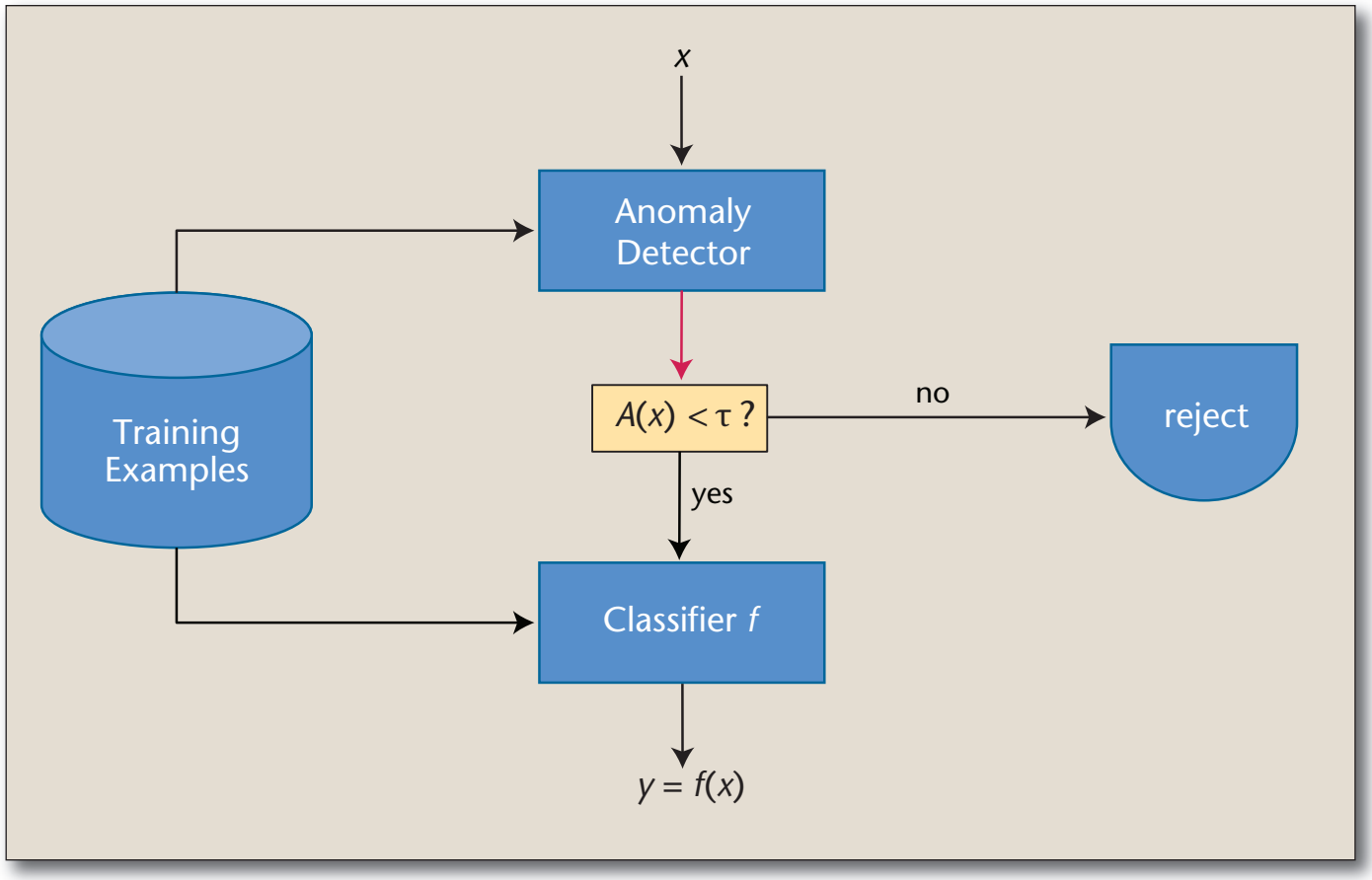

Figure 11. Screening for Novel Inputs Through Anomaly Detection.

particularly challenging is that we seek to detect individual queries x that correspond to novel classes. The problem becomes easier if we are willing to delay detection until we have accumulated more data. This is the setting of change-point detection in which it is assumed that for some period of time after training, the queries continue to come from P_{train}, but then a change point occurs, and the queries shift to a different distribution P_{test}. A standard approach to change-point detection is to collect up the k most recent points $\{x_{(t-k)}, \ldots, x_{t-1}\}$ and compare their distribution to the previous k points $\{x_{t-2k}, \ldots, x_{t-k-1}\}$. This can be done through a two-sample test (Gretton et al. 2012). Of course a drawback of this approach is that there is a k-step lag between the time the change point occurs and the time it is detected. Change-point detection has a long history in engineering and statistics (Page 1955, Barry and Hartigan 1993, Adams and MacKay 2007). Most methods can detect multiple change points over time.

When a change has been detected — and if the change does not involve novel classes — then there are several methods for adapting to the change. Methods for covariate shift (Huang et al. 2007; Sugiyama, Krauledat, and Müller 2007; Cortes et al. 2008; Tsuboi et al. 2009; Sugiyama, Suzuki, and Kanamori 2012) assume that the conditional probability $P(y|x)$ of the outputs is invariant and only the distribution $P(x)$ of the inputs has changed. Methods for domain adaptation (Blitzer, McDonald, and Pereira 2006; Ben-David et al. 2007; Ben-David et al. 2010) are designed to handle arbitrary changes in the distributions. They seek to find an intermediate representation that captures the shared aspects of multiple domains (that is, the shared aspects of the joint distributions $P_{train}(x, y)$ and $P_{test}(x, y)$).

Idea 6: Use Causal Models

Causal models (Pearl 2009) account for the effects of interventions (actions). In so doing, they tend to be more compact and capture more conditional independence relationships than models based on statistical correlations (Schachter and Heckerman 1987). This also means that they can be easier to learn (Meek and Heckerman 1997). A fascinating aspect of causal models is that they are more transportable than correlative models. Indeed, it is precisely their transportability that motivates scientists to seek causal

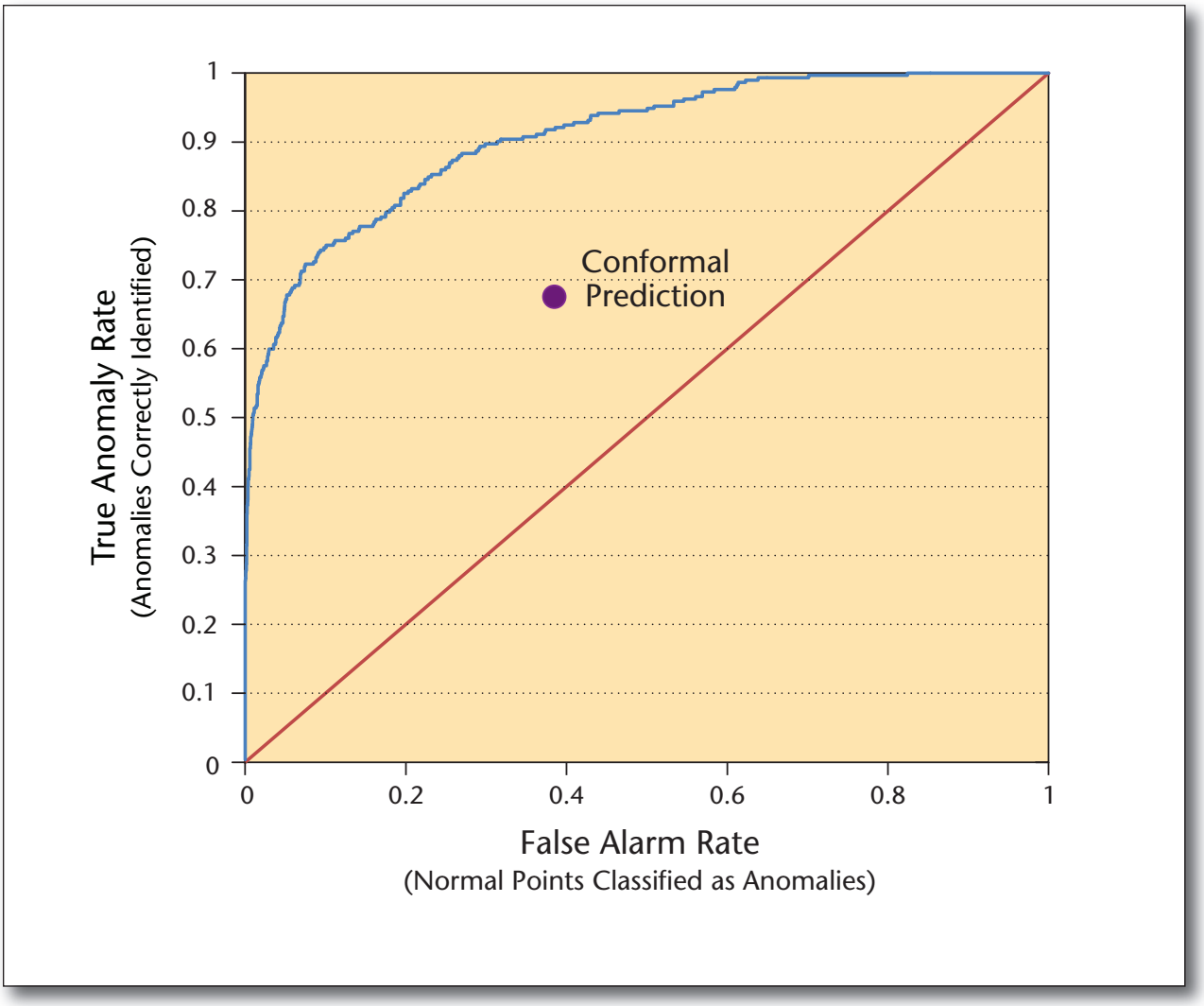

Figure 12. Effectiveness of Anomaly Detection.

ROC curve (blue) is shifted upwards and two the left.

models. Once we understand which variables are causally connected and which are only correlated, we can make successful predictions in novel situations as long as the causal variables are the same. Recent work has formalized the conditions under which causal models are transportable (Pearl and Bareinboim 2011, Bareinboim and Pearl 2012, Lee and Honavar 2013).

Idea 7: Portfolio Methods

A third approach to making AI systems robust to model incompleteness is to adopt portfolio (or ensemble) methods. As Minsky said, "We usually know several different ways to do something, so that if one of them fails, there's always another." Ensemble methods are applied universally in machine learning when the computational cost can be managed, and even deep networks benefit from being combined into ensembles (He et al. 2016).

A line of research that relates closely to Minsky's point is the work on portfolio methods in satisfiability solvers. One of the first such systems was SATzilla (Xu et al. 2008). A key aspect that is exploited by SATzilla and other SAT solver portfolios is that they can detect when they have found a solution to a SAT problem. This is a very powerful form of metaknowledge that is not available to machine-learning ensembles.

SATzilla was optimized for a benchmarking competition in which a collection of SAT problem instances is designed, and the system is given at most 1200 seconds to solve each instance. SATzilla has been tested on several different benchmark collections. Here, I report the results on the HANDMADE benchmark, which contains 1490 problem instances.

Figure 13 shows the pipeline of SATzilla. Given a SAT problem instance, SATzilla first applies two SAT solvers (presolver1 and presolver2) in sequence with

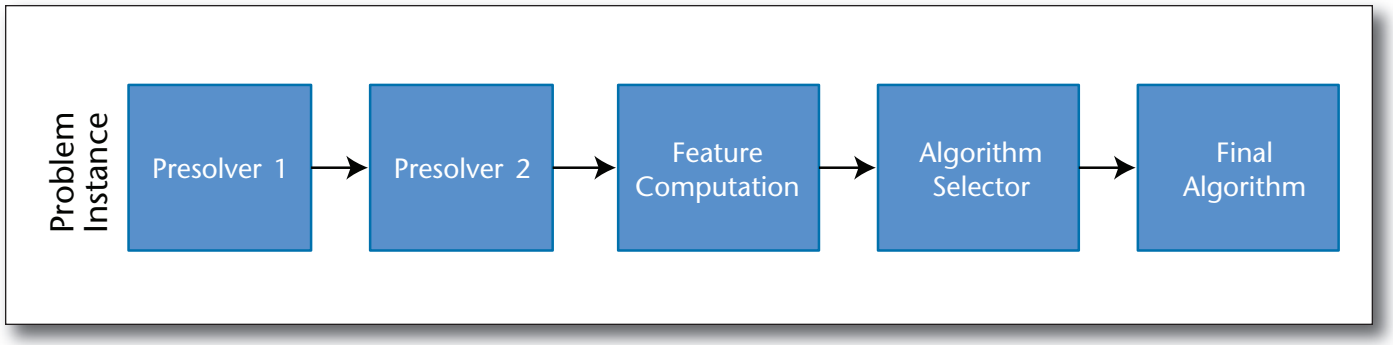

Figure 13. SATzilla Processing Pipeline.

very small time budgets. In the configuration that I discuss here, presolver1 is the systematic solver March_d104 (Heule et al. 2004) and presolver2 is the stochastic solver SAPS (Hutter, Tompkins, and Hoos 2002). March_d104 is typically able to solve 40–45 percent of the problem instances within this 5-second budget, and SAPS is able to solve an additional 5–14 percent of the instances within its 2-second budget. If these two solvers are not able to find a satisfying assignment, SATzilla spends some time computing 48 features describing the instance. These features include properties of the problem size such as the number of clauses, number of variables, and the ratio of clauses to variables and properties of the variable-clause graph such as the min, max, mean, and entropy of the degree distribution of the variables and clauses. Additional features are computed by analyzing the results of applying local search and DPLL (Davis and Putnam 1960; Davis, Logemann, and Loveland 1962) each to the problem for 1 second of CPU time. Examples of these features include the number of unit propagations (for DPLL) and statistics on the number of steps to the best local optimum (for SAPS). These features are then fed to a machine-learning classifier that selects one of seven different solvers to use for the time remaining.

Figure 14 plots the percentage of instances solved as a function of time. The heavy black line is the performance of an oracle that knows the best SAT solver to apply to each instance (computed offline, of course). It solves 100 percent of the instances in less than 1200 seconds (each). The three dashed lines show the performance (from bottom to top) of three solvers, Minisat2.0, Valist, and March_d104, when these methods are applied to solve all instances. The best of these, March_d104, is only able to solve 80 percent of the instances within the 1200 second time limit. Finally, the red curve (underneath the heavy black line) is the SATzilla portfolio method, which can solve 92 percent of the problem instances within the time budget.

This figure illustrates how a portfolio of methods — without explicitly representing or reasoning about its uncertainty concerning the optimal solver — can achieve more robust performance than any single method alone.

The idea of portfolio methods is broadly applicable. For example, if multiple computers (or cores) are available, SAT solvers can be applied in parallel, and as soon as one solver has found a solution, the others can be terminated (Yun and Epstein 2012). We can also view IBM's Watson system as a portfolio method (Ferrucci 2012). Watson combines more than 100 different techniques for analyzing natural language, identifying relevant information sources, finding and generating hypotheses, finding and scoring evidence, and merging and ranking hypotheses.

In addition to developing algorithm portfolios, I urge the AI community to consider what we might call knowledge-level portfolios. Another one of Marvin Minsky's aphorisms was this: "You don't really understand something if you only understand it one way." In his 1988 book *Society of Mind,* Minsky devotes a chapter to what he calls *learning meaning*. He explores the problem of learning the definition of a blocks world arch. Patrick Winston's doctoral dissertation (Winston 1970) applied an early form of relational learning to learn that (in the toy blocks world) an arch consists of two upright blocks and a third horizontal block resting on top of them. This is a structural understanding. But Minsky pointed out that there is a functional notion of an arch too: When you are pushing a toy car through it, you must change hands to complete the action. I call this *multifaceted understanding*. More recent instances of this idea include multiview learning (Blum and Mitchell 1998) and the work on learning to recognize handwritten characters by combining appearance with a model of how to draw each character (Lake, Salakhutdinov, and Tenenbaum 2015).

There are many benefits to having a multifaceted understanding of a concept. First, the multiple views reinforce each other. Work in machine learning and computer vision shows that learning is more successful and requires less data when we have multiple independent *views* of an object. Second, the multiple views give us multiple ways of recognizing the object. To decide whether something is an arch, we can ask

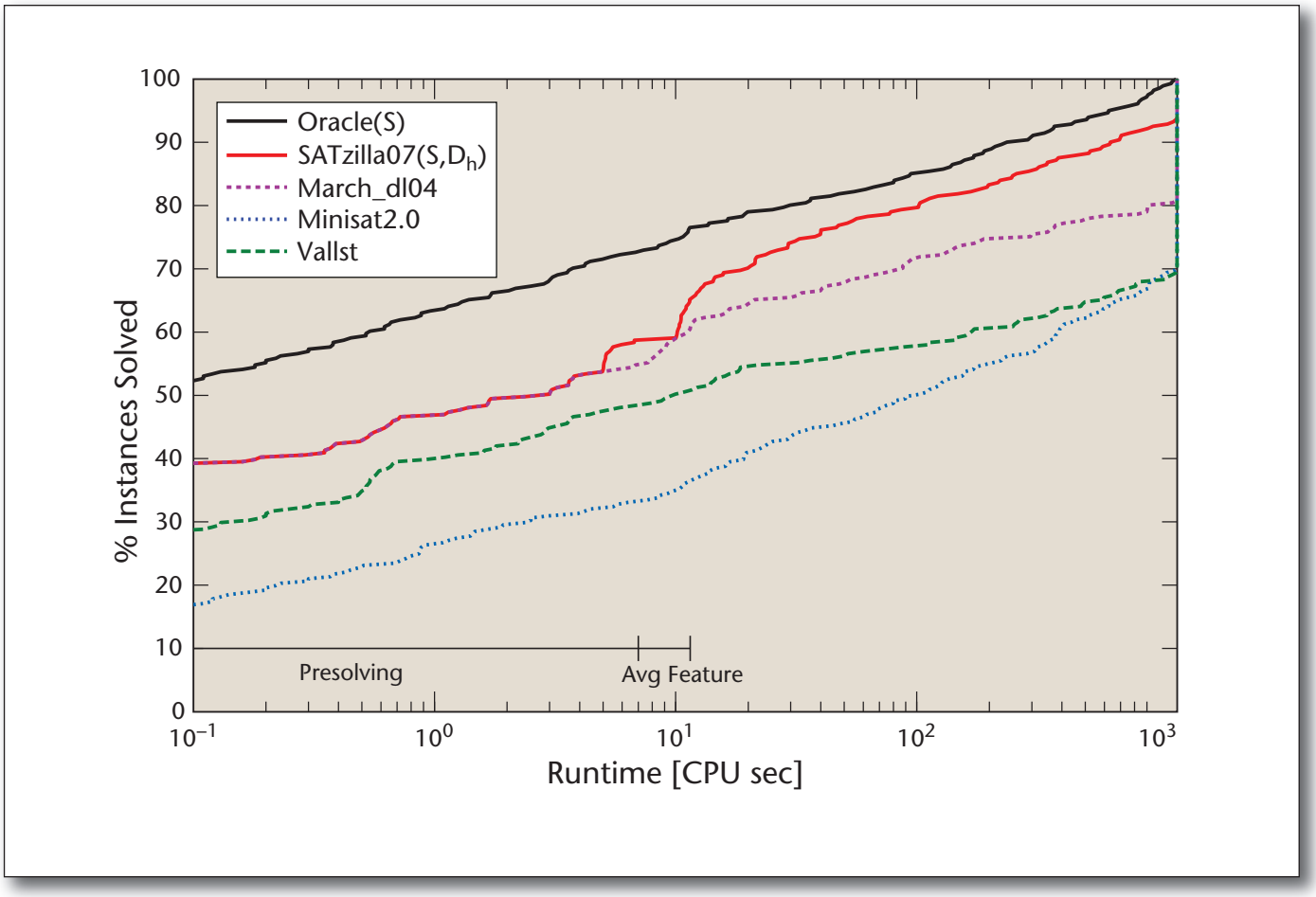

Figure 14. Performance of SATzilla on HANDMADE Problem Set.

Originally published in Xu et al (2008) (figure 8, p. 594). Reprinted with permission.

"Does it look like an arch?" and also "Would I need to change hands to push something through it?" (which, if I'm evaluating the St. Louis arch, would require me to imagine I am a giant). This redundancy helps us be more robust to unusual arches.

Unfortunately, virtually all of our current AI systems understand things only one way. Consider, for example, the recent work on image captioning. Figure 15 shows the output of the Berkeley image-captioning system. The result seems impressive until you realize that the computer vision system has a very narrow understanding of cats, chairs, and sitting. It has developed a good model of the kinds of images that people will label with keywords such as *cat* and *chair*. This is an impressive accomplishment, because there is a high degree of variability on the appearance of these objects. However, this vision system has not learned to localize these objects within the image, so it knows nothing about the typical size of cats versus chairs, for example. It chooses to include the word *sitting* based on word co-occurrence statistics: when people write captions for images that contain both cats and chairs, they often use the word *sitting*.

Beyond the task of linguistic description, the system doesn't know anything about the typical context in which a cat is sitting on a chair. It doesn't know that there is a human who owns the cat and the chair. It doesn't know that the cat is preventing the human from sitting on the chair and that the human is often annoyed by this because the cat also leaves hair on the chair. It therefore can't predict that the cat will soon not be sitting on the chair.

In my view, an important priority for AI research is to find ways to give our computers multifaceted understanding of the world. To do this with machine learning, we need to give our computers experience performing tasks, achieving goals through natural language dialogue, and interacting with other agents. The greater the variety of tasks that the computer learns to perform, the larger the number of different facets it will acquire, and the more robust its knowledge will become.

"a black and white cat is sitting on a chair."

Figure 15. Example Output from the Berkeley Image-Captioning System.

Idea 8: Expand the Model

The final method for improving the robustness of AI systems to the unknown unknowns is to expand the model. We can all point to examples of ways in which our AI systems fail because they know so little. While it is impossible to create a "model of everything," our existing systems fail primarily because they have a model of almost nothing.

There have been some notable efforts. Doug Lenat's decades-long effort to create a large commonsense knowledge base, CYC, led to some interesting applications and insights (Lenat et al. 1990) and has been licensed to Lucid (Knight 2016). Recent work has seen the development of systems that can extract concepts and properties from the World Wide Web to grow and populate a knowledge base (Mitchell et al. 2015). NIST has been operating a knowledge base population competition to evaluate such systems (Surdeanu and Ji 2014). Google employs a knowledge graph that contains millions of objects and relationships, and other companies including Microsoft, Yandex, LinkedIn, and Baidu have built similar semantic networks.

There are some risks to expanding our models. Every time we add something to the model, we may introduce an error. Inference can then propagate that error. The result may be that the expanded model is less accurate and less useful than the original model. It is important to test our models continually to prevent this from happening. A beautiful aspect of the application of knowledge bases in web search is that because millions of queries are processed every day, errors in the models can be identified and removed.

Summary

AI has been making exciting progress. The last two decades have seen huge improvements in perception (for example, computer vision, speech recognition), reasoning (for example, SAT solving, Monte Carlo Tree Search), and integrated systems (for example, IBM's Watson, Google's AlphaGo, and personal digital assistants). These advances are encouraging us to apply AI to difficult, high-stakes applications including self-driving cars, robotic surgery, finance, real-time control of the power grid, and autonomous

weapons systems. These applications require AI systems that are highly robust, and yet our current systems fall far short. To create the level of robustness required for such high-risk applications, we need systems that are robust both to known unknowns (the uncertainty they represent explicitly) and to unknown unknowns (unmodeled aspects of the world).

In this article, I've made an (incomplete) catalogue of the ideas and methods that the AI community has developed for achieving robustness. To manage the known unknowns, we can build on our existing methods for representing uncertainty using probability distributions or uncertainty intervals. We can then define robust optimization problems in which we search for the optimal solution when competing against an adversary that is given a fixed budget. An important idea for achieving robustness in machine learning is regularization. We saw that it is possible to reformulate regularization in terms of finding a robust optimum against an adversary who can perturb the data points. A second important idea for achieving robustness is to optimize a risk-sensitive objective, such as the conditional value at risk (CVaR). Again we saw that it is possible to formulate CVaR optimization as finding a robust optimum against an adversary who can perturb the transition probabilities of our dynamical model of the world. This tells us that acting conservatively can confer robustness. Finally, we explored how to make inference itself robust and discussed work on robust probabilistic and diagnostic reasoning.

I then turned to cataloguing our ideas about the unknown unknowns. The first idea is to develop methods for detecting when our model is inadequate before our AI system makes a mistake. I discussed work on anomaly detection and change-point detection that can protect a system against changes in the data distribution. A second idea is to learn causal models, because they have been proven to be more transportable and therefore more robust to changes in the context of decision making. I spent a long time discussing the third idea, which is to employ ensembles or portfolios of methods. I looked at algorithm portfolios for SAT solving as well as knowledge-level portfolios in which the AI system models multiple facets (such as structure, function, and appearance) of objects in the world. Finally, I discussed the idea of continually expanding the knowledge that our systems possess. While it is impossible to know everything, we can hope that "on the average, and in the long run, more knowledge is better than less" (Herbert Simon, Harry Camp Lectures at Stanford University, 1982).

Acknowledgments

I wish to thank the many people who provided suggestions and pointers to relevant research: David Ackley, Stefano Albrecht, Juan Augusto, Randall Davis, Pedro Domingos, Alan Fern, Boi Faltings, Stephanie Forrest, Helen Gigley, Barbara Grosz, Vasant Honavar, Holgar Hoos, Eric Horvitz, Michael Huhns, Rebecca Hutchinson, Mykel Kochenderfer, Pat Langley, Sridhar Mahadevan, Shie Mannor, Melanie Mitchell, Dana Nau, Takayuki Osogami, Don Perlis, Jeff Rosenschein, Dan Roth, Stuart Russell, Tuomas Sandholm, Rob Schapire, Scott Sanner, Prasad Tadepalli, Milind Tambe, Brian Williams, Zhi-hua Zhou. I apologize that I was not able to weave in all of the great work that folks described. I also thank the *AI Magazine* editor Ashok Goel for his suggestions and editorial improvements.

This work was partially supported by the Future of Life Institute (futureoflife.org) FLI-RFP-AI1 program, grant number 2015-145014, and by NSF grants 0705765 and 0832804.

References

Adams, R. P., and MacKay, D. J. 2007. Bayesian Online Changepoint Detection. arXiv Preprint. arXiv:0710.3742 [stat.ML]. Ithaca, NY: Cornell University Library.

Antonucci, A., and Zaffalon, M. 2007. Fast Algorithms for Robust Classification with Bayesian Nets. *International Journal of Approximate Reasoning* 44(3): 200–223. doi.org/10.1016/j.ijar.2006.07.011

Arkin, R. C. 2009. *Governing Lethal Behavior in Autonomous Robots*. London: Chapman and Hall/CRC. doi.org/10.1201/9781420085952

Bareinboim, E., and Pearl, J. 2012. Transportability of Causal Effects: Completeness Results. In *Proceedings of the 26th National Conference on Artificial Intelligence* (AAAI-2012), 698–704. Menlo Park, CA.: AAAI Press.

Barry, D., and Hartigan, J. A. 1993. A Bayesian Analysis for Change Point Problems. *Journal of the American Statistical Association* 88(421): 309–319. doi.org/10.1080/01621459.1993.10594323

Ben-David, S.; Blitzer, J.; Crammer, K.; Kulesza, A.; Pereira, F.; and Vaughan, J. W. 2010. A Theory of Learning from Different Domains. *Machine Learning* 79(1): 151–175.

Ben-David, S.; Blitzer, J.; Crammer, K.; and Pereira, F. 2007. Analysis of Representations for Domain Adaptation. In *Advances in Neural Information Processing Systems 19*, 137. Cambridge, MA: The MIT Press. doi.org/10.1007/s10994-009-5152-4

Bendale, A., and Boult, T. 2015. Towards Open World Recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition*, 1893–1902. Piscataway, N.J.: Institute for Electrical and Electronics Engineers. doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298799

Bertsimas, D., and Thiele, A. 2006. Robust and Data-Driven Optimization: Modern Decision Making Under Uncertainty. In *Tutorials in Operations Research: Models, Methods, and Applications for Innovative Decision Making*, 95–122. Catonsville, MD: The Institute for Operations Research and the Management Sciences (Informs).

Betancourt, M. 2017. A Conceptual Introduction to Hamiltonian Monte Carlo. arXiv Preprint. arXiv:1701.02434 [stat.ME]. Ithaca, NY: Cornell University Library.

Blitzer, J.; McDonald, R.; and Pereira, F. 2006. Domain Adaptation with Structural Correspondence Learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural*

Language Processing, (EMNLP 2007), 120–128. Stroudsberg, PA: Association for Computational Linguistics. doi.org/10.3115/1610075.1610094

Blum, A., and Mitchell, T. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory,* 92–100. New York: Association for Computing Machinery. doi.org/10.1145/279943.279962

Bowling, M.; Burch, N.; Johanson, M.; and Tammelin, O. 2015. Heads-Up Limit Hold'em Poker Is Solved. *Science* 347(6218): 145–149. doi.org/10.1126/science.1259433

Brockman, J. 1996. *Third Culture: Beyond the Scientific Revolution.* New York: Simon and Schuster.

Chen, S. J.; Choi, A.; and Darwiche, A. 2014. Algorithms and Applications for the Same-Decision Probability. *Journal of Artificial Intelligence Research* 49: 601–633.

Chen, S. J., Choi, A.; and Darwiche, A. 2015. Value of Information Based on Decision Robustness. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence,* 3503–3510. Palo Alto, CA: AAAI Press.

Chow, Y.; Tamar, A.; Mannor, S.; and Pavone, M. 2015. Risk-Sensitive and Robust Decision-Making: A CVaR Optimization Approach. In Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, (NIPS 2015), 1522–1530. December 7–12, 2015, Montréal, Québec, Canada.

Cortes, C.; Mohri, M.; Riley M.; and Rostamizadeh, A. 2008. Sample Selection Bias Correction Theory. *Algorithmic Learning Theory,* Lecture Notes in Computer Science volume 5254, 38–53. Berlin: Springer. doi.org/10.1007/978-3-540-87987-9_8

Cozman, F. 1997. Robustness Analysis of Bayesian Networks with Local Convex Sets of Distributions. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence,* 108–115. San Francisco: Morgan Kaufmann Publishers Inc.

Cozman, F. G. 2000. Credal Networks. *Artificial intelligence* 120(2): 199–233. doi.org/10.1016/S0004-3702(00)00029-1

Da, Q.; Yu, Y.; and Zhou, Z.-H. 2014. Learning with Augmented Class by Exploiting Unlabeled Data. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence* (AAAI 2014), 1760–1766. Palo Alto, CA: AAAI Press.

Davis, M.; Logemann, G.; and Loveland, D. 1962. A Machine Program for Theorem-Proving. *Communications of the ACM* 5(7): 394–397. doi.org/10.1145/368273.368557

Davis, M., and Putnam, H. 1960. A Computing Procedure for Quantification Theory. *Journal of the ACM* (JACM) 7(3): 201–215. doi.org/10.1145/321033.321034

Dietterich, T. G. 1986. Learning at the Knowledge Level. *Machine Learning* 1(3): 287–315. doi.org/10.1007/BF00116894

Félix, M.-A., and Barkoulas, M. 2015. Pervasive Robustness in Biological Systems. *Nature Reviews Genetics* 16: 483–496. doi.org/10.1038/nrg3949

Ferrucci, D. A. 2012. Introduction to This Is Watson. *IBM Journal of Research and Development* 56(3.4): 1:1–1:15.

Gilks, W. R.; Richardson, S.; and Spiegelhalter, D.1995. *Markov Chain Monte Carlo in Practice.* Boca Raton, FL: CRC Press.

Gopakumar, P.; Reddy, M. J. B.; and Mohanta, D. K. 2014. Stability Control of Smart Power Grids with Artificial Intelligence and Wide-Area Synchrophasor Measurements. *Electric Power Components and Systems* 42(10): 1095–1106. doi.org/10.1080/15325008.2014.913745

Gordon, A. D.; Henzinger, T. A.; Nori, A. V.; and Rajamani, S. K. 2014. Probabilistic Programming. In *FOSE 2014: Proceedings on the Future of Software Engineering*. Hyderabad, India, 167–181. New York: Association for Computing Machinery. doi.org/10.1145/2593882.2593900

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A Kernel Two-Sample Test. *Journal of Machine Learning Research* 13: 723–773.

Grinberg, Y.; Precup, D.; and Gendreau, M. 2014. Optimizing Energy Production Using Policy Search and Predictive State Representations. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems* 2014, 3657–3665. La Jolla, CA: Neural Information Processing Systems Foundation, Inc.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 770–778. Piscataway, NJ: Institute for Electrical and Electronics Engineers. doi.org/10.1109/cvpr.2016.90

Heule, M.; Dufour, M.; van Zwieten, J.; and van Maaren, H. 2004. March_eq: Implementing Additional Reasoning into an Efficient Look-Ahead SAT Solver. In *Theory and Applications of Satisfiability Testing. SAT 2004* Lecture Notes in Computer science volume 3542, 345–359. Berlin: Springer. doi.org/10.1007/11527695_26

Human Rights Watch. 2016. Killer Robots and the Concept of Meaningful Human Control: Memorandum to Convention on Conventional Weapons (CCW) Delegates. 11 April. New York: Humans Rights Watch, Inc. (www.hrw.org/news/2016/04/11/killer-robots-and-concept-meaningful-human-control).

Huang, J.; Smola, A. J.; Gretton, A.; Borgwardt, K. M.; and Schölkopf, B. 2007. Correcting Sample Selection Bias by Unlabeled Data. *Advances in Neural Information Processing Systems 19,* 601–608. Cambridge, MA: The MIT Press.

Hutter, F.; Tompkins, D. A. D.; and Hoos, H. H.. 2002 Scaling and Probabilistic Smoothing: Efficient Dynamic Local Search for SAT. In *Principles and Practice of Constraint Programming — CP 2002*. Lecture Notes in Computer Science Volume 2470, ed. P. Van Hentenryck, 233–248. Berlin: Springer. doi.org/10.1007/3-540-46135-3_16

Kirilenko, A.; Kyle, A. S.; Samadi, M.; and Tuzun, T. 2017. The Flash Crash: High Frequency Trading in an Electronic Market. *The Journal of Finance* 72(3): 967–998. doi.org/10.1111/jofi.12498

Kitano, H. 2004. Biological Robustness. *Nature Reviews Genetics* 5(11): 826–837. doi.org/10.1038/nrg1471

Knight, W. 2016. An AI with 30 Years' Worth of Knowledge Finally Goes to Work. *MIT Technology Review* March 14.

Kocsis, L., and Szepesvári, C. 2006. Bandit Based Monte-Carlo Planning. In *Machine Learning: ECML 2006, Proceedings of the 17th European Conference on Machine Learning,* Berlin, Germany. Lecture Notes in Computer Science 4212, 282–203. Berlin: Springer. doi.org/10.1007/11871842_29

Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA: The MIT Press.

Lake, B. M.; Salakhutdinov, R.; and Tenenbaum, J. B. 2015. Human-Level Concept Learning Through Probabilistic Program Induction. *Science* 350(6266): 1332–1338. doi.org/10.1126/science.aab3050

Lee, S., and Honavar, V. 2013. Causal Transportability of Experiments on Controllable Subsets of Variables: Z-Trans-

portability. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence,* 361–370. Corvallis, OR: Association for Uncertainty in Artificial Intelligence Inc.

Lenat, D. B.; Guha, R. V.; Pittman, K.; Pratt, D.; and Shepherd, M. 1990. Cyc: Toward Programs with Common Sense. *Communications of the ACM* 33(8): 30–49. doi.org/ 10.1145/ 79173.79176

Lichman, M. 2013. UCI Machine Learning Respository, School of Information and Computer Sciences. Irvine, CA: University of California, Irvine.

Liu, F. T.; Ting, K. M.; and Zhou, Z.-H. 2012. Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data* 6(1): 1–39.

McCarthy, J.; Minsky, M.; Rochester, N.; and Shannon, C. E. 1955. A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence. *AI Magazine* 27(4): 12–14. doi.org/10.1609/aimag.v27i4.1904

Meek, C., and Heckerman, D. 1997. Structure and Parameter Learning for Causal Independence and Causal Interaction Models. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence,* 366–375. San Francisco: Morgan Kaufmann Publishers Inc.

Minsky, M. 1961. Steps Toward Artificial Intelligence. *Proceedings of the IRE* 49(1): 8–30. doi.org/10.1109/JRPROC.1961.28777

Minsky, M. 1988. *Society of Mind*. New York: Simon and Schuster.

Mitchell, T. M.; Cohen, W.; Hruschka, E.; Talukdar, P.; Betteridge, J.; Carlson, A.; Mishra, B. D.; Gardner, M.; Kisiel, B.; Krishnamurthy, J.; Lao, N.; Mazaitis, K.; Mohamed, T.; Nakashole, N.; Platanios, E. A.; Ritter, A.; Samadi, M.; Settles, B.; Wang, R.; Wijaya, D.; Gupta, A.; Chen, X.; Saparov, A.; Greaves, M.; and Welling, J. 2015. Never-Ending Learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (AAAI-15), 2302–2310. Palo Alto, CA: AAAI Press.

Murphy, K. P. 2012. *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: The MIT Press.

Neal, R. M. 1993. Probabilistic Inference Using Markov Chain Monte Carlo Methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada. 25 September.

Page, E. 1955. A Test for a Change in a Parameter Occurring at an Unknown Point. *Biometrika* 42(3/4): 523–527. doi.org/10.2307/2333401

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems*. San Francisco: Morgan Kaufmann Publishers.

Pearl, J. 2009. *Causality.* Cambridge, UK: Cambridge University Press. doi.org/10.1017/CBO9780511803161

Pearl, J., and Bareinboim, E. 2011. Transportability of Causal and Statistical Relations: A Formal Approach. In *Proceedings of the 25th National Conference on Artificial Intelligence* (AAAI), 247–254. Menlo Park, CA; AAAI Press.

Pfeffer, A. 2016. *Practical Probabilistic Programming*. Shelter Island, NY: Manning Publications.

Rudd, E. M.; Jain, L. P.; Scheirer, W. J.; and Boult, T. E. 2016. The Extreme Value Machine. Preprint of a manuscript accepted to the *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* journal. arXiv:1506.06112 [cs.LG]. Ithaca, NY: Cornell University Library.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. Imagenet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision* 115(3): 211–252. doi.org/10.1007/s11263-015-0816-y

Russell, S. J., and Norvig, P. 2009. *Artificial Intelligence: A Modern Approach* (3rd edition). Engelwood Cliffs, NJ: Prentice Hall.

Schachter, R. D., and Heckerman, D. 1987. Thinking Backward for Knowledge Acquisition. *AI Magazine* 8(3): 55. doi.org/10.1609/aimag.v8i3.600

Schaeffer, J.; Müller, M.; and Kishimoto, A. 2014. Go-Bot, Go. *IEEE Spectrum* 51(7): 48–53. doi.org/10.1109/MSPEC.2014.6840803

Scharre, P. 2016. Autonomous Weapons and Operational Risk. Ethical Autonomy Project. February 2016. Washington, DC: Center for a New American Security.

Scheirer, W. J.; Rocha, A.; Sapkota, A.; and Boult, T. E. 2013. Towards Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(7): 1757–72. doi.org/10.1109/TPAMI.2012.256

Sennrich, R. 2016. Neural Machine Translation: Breaking the Performance Plateau. Slides of an Invited Talk Delivered at the Multilingual Europe Technology Alliance (META-FORUM 2016), 6 July 2016, Lisbon, Portugal. Institute for Language, Cognition and Computation, University of Edinburgh, Edinburgh, UK..

Shademan, A.; Decker, R. S.; Opfermann, J. D.; Leonard, S.; Krieger, A.; and Kim, P. C. W. 2016. Supervised Autonomous Robotic Soft Tissue Surgery. *Science Translational Medicine* 8(337): 337–364.

Shafer, G., and Vovk, V. 2008. A Tutorial on Conformal Prediction. *Journal of Machine Learning Research* (JMLR) 9: 371–421.

Solis, M. 2016. New Frontiers in Robotic Surgery. *IEEE Pulse* (November/December): 51–55. doi.org/10.1109/MPUL.2016.2606470

Steinhardt, J., and Liang, P. S. 2016. Unsupervised Risk Estimation Using Only Conditional Independence Structure. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016,* 3657–3665. La Jolla, CA: Neural Information Processing Systems Foundation, Inc.

Sugiyama, M.; Krauledat, M.; and Müller, K.-R. 2007. Covariate Shift Adaptation by Importance Weighted Cross Validation. *Journal of Machine Learning Research* 8(May): 985–1005.

Sugiyama, M.; Suzuki, T.; Kanamori, T. 2012. Density Ratio Estimation in Machine Learning. New York: Cambridge University Press. doi.org/10.1017/CBO9781139035613

Surdeanu, M., and Ji, H. 2014. Overview of the English Slot Filling Track at the TAC2014 Knowledge Base Population Evaluation. In *Proceedings of the Text Analysis Conference* (TAC2014). Gaithersburg, MD: National Institute of Standards and Technology, US Department of Commerce.

Thrun, S.; Burgard, W.; and Fox, D. 2005. *Probabilistic Robotics*. Cambridge, MA: The MIT Press.

Tsuboi, Y.; Kashima, H.; Hido, S.; Bickel, S.; and Sugiyama, M. 2009. Direct Density Ratio Estimation for Large-Scale Covariate Shift Adaptation. *Journal of Information Processing* 17(January-December): 138–155. doi.org/10.2197/ipsjjip.17.138

Whitacre, J. M. 2012. Biological Robustness: Paradigms, Mechanisms, Systems Principles. *Frontiers in Genetics* 3(May): 1–15. doi.org/10.3389/fgene.2012.00067

Photo courtesy iStock

Save the Date for ICWSM-18!

Please join us for the Twelfth International AAAI Conference on Web and Social Media, to be held at Stanford University, Stanford, California, USA, June 24–28, 2018.

This interdisciplinary conference is a forum for researchers in computer science and social science to come together to share knowledge, discuss ideas, exchange information, and learn about cutting-edge research in diverse fields with the common theme of online social media. This overall theme includes research in new perspectives in social theories, as well as computational algorithms for analyzing social media.

ICWSM is a singularly fitting venue for research that blends social science and computational approaches to answer important and challenging questions about human social behavior through social media while advancing computational tools for vast and unstructured data.

Full conference details will be posted at on the conference website (www.icwsm.org/2018) as they become available.

Winston, P. H. 1970. Learning Structural Descriptions from Examples. PhD dissertation, Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology. Report AITR-231, September 1, Cambridge, MA. (hdl.handle.net/1721.1/6884).

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; Klingner, J.; Shah, A.; Johnson, M.; Liu, L.; Kaiser, L.; Gouws, S.; Kato, Y.; Kudo, T.; Kazawa, H.; Stevens, K.; Kurian, G.; Patil, N.; Wang, W.; Young, C.; Smith, J.; Riesa, J.; Rudnick, A.; Vinyals, O.; Corrado, G.; Hughes, M.; and Dean, J. 2016. Google's Neural Machine Translation System: Bridging the Gap Between Human and Machine Translation. arXiv Preprint. arXiv:1609.08144 [cs.CL]. Ithaca, NY: Cornell University Library.

Xu, H.; Caramanis, D.; and Mannor, S. 2009. Robustness and Regularization of Support Vector Machines. *Journal of Machine Learning Research* 10: 1485–1510.

Xu, L.; Hutter, F.; Hoos, H. H.; and Leyton-Brown, K. 2008. SATzilla: Portfolio-Based Algorithm Selection for SAT. *Journal of Artificial Intelligence Research* 32(1): 565–606.

Yun, X., and Epstein, S. L. 2012. Learning Algorithm Portfolios for Parallel Execution. In *Learning and Intelligent Optimization,* ed. Y. Hamadi and M. Schoenauer. Lecture Notes in Computer Science, vol 7219, 323–338. Berlin: Springer-Verlag. doi.org/10.1007/978-3-642-34413-8_2

Thomas G. Dietterich (AB Oberlin College 1977; MS University of Illinois 1979; PhD Stanford University 1984) is a professor emeritus and director of intelligent systems research in the School of Electrical Engineering and Computer Science at Oregon State University, where he joined the faculty in 1985. Dietterich has devoted his career to machine learning and artificial intelligence. He has authored more than 180 publications and two books. His research is motivated by challenging real world problems with a special focus on ecological science, ecosystem management, and sustainable development. Dietterich has devoted many years of service to the research community. He is past president of AAAI, and he previously served as president of AAAI (2014-16) and as the founding president of the International Machine Learning Society (2001-08). Other major roles include executive editor of the journal *Machine Learning* (1992-98), co-founder of the *Journal for Machine Learning Research* (2000), and program chair of AAAI 1990 and NIPS 2000. Dietterich is a Fellow of the ACM, AAAS, and AAAI.