Chapter 9
Analyzing a Decade of Human-Competitive (“HUMIE”) Winners: What Can We Learn?

Karthik Kannappan, Lee Spector, Moshe Sipper, Thomas Helmuth, William La Cava, Jake Wisdom and Omri Bernstein

9.1 Introduction

In the field of evolutionary computation (EC) ideas from evolutionary biology—random variation and selection—are harnessed in algorithms that are applied to complex computational problems. The origins of EC can be traced back to the 1950s and 1960s but the field has come into its own over the past two decades, proving successful in solving numerous problems from highly diverse domains (Sipper 2002). EC techniques are being increasingly applied to difficult real-world problems, often yielding results that are not merely academically interesting but also competitive with the work done by creative and inventive humans. Indeed, a recent emerging theme is that of *human-competitive machine intelligence*, produced by evolutionary means (Koza 2008, 2010).

K. Kannappan (✉) · T. Helmuth
School of Computer Science, University of Massachusetts, Amherst, MA, USA
email: kkarthik@cs.umass.edu

T. Helmuth
e-mail: thelmuth@cs.umass.edu

L. Spector · J. Wisdom · O. Bernstein
Cognitive Science, Hampshire College, Amherst, MA 01002-3359, USA
e-mail: lspector@hampshire.edu

J. Wisdom
e-mail: jake.r.wisdom@gmail.com

O. Bernstein
e-mail: hippiccolo@gmail.com

M. Sipper
Department of Computer Science, Ben-Gurion University, 84105 Beer-Sheva, Israel
e-mail: sipper@cs.bgu.ac.il

W. La Cava
Department of Mechanical and Industrial Engineering, University of Massachusetts, Amherst, MA, USA
e-mail: wlacava@umass.edu

R. Riolo et al. (eds.), *Genetic Programming Theory and Practice XII,*
Genetic and Evolutionary Computation, DOI 10.1007/978-3-319-16030-6_9

A recent survey cited 28 instances in which genetic programming (GP), a form of EC, “has duplicated the functionality of a previously patented invention, infringed a previously issued patent, or created a patentable new invention” and cited “over a dozen additional known instances where genetic programming has produced a human-competitive result that is not patent related” (Koza 2008). These results come from an astonishing variety of fields, including image analysis, game playing, quantum computer programming, software repair, and the design of complex objects such as analog circuits, antennas, photonic crystals, and polymer optical fibers.

We believe this to be more than a mere novel line of research within a single research community. Surpassing humans in the ability to solve complex problems is a *grand challenge*, with potentially far-reaching, transformative implications.

In this chapter we take a close look at the 42 winners of the past decade (2004–2013) of Human-Competitive (HUMIE) competitions, seeking to draw conclusions about past and future directions of the field.

We note that two of the authors (Spector, Sipper) have extensive experience in human-competitive research, having won between them eight HUMIE awards (Koza 2010). In addition, Spector has served as a judge for the HUMIES awards for some years. Spector and his colleagues earned the competition’s top prize twice, once for the use of EC to produce quantum computing results that were published in a top physics journal (Barnum et al. 2000; Spector 2004) and once for results in pure mathematics that exceeded human performance by several orders of magnitude (Spector et al. 2008). Sipper, who has six wins, tackled a string of hard games and puzzles, evolving game-playing strategies that held their own in competition against humans (Sipper 2006; Hauptman et al. 2009; Benbassat et al. 2012). In collaborative work with a partner from the semiconductors industry Sipper attained marked improvement over humans in developing automatic defect classifiers for patterned wafers (Glazer and Sipper 2008).

9.2 The HUMIES

As of 2004, one of the major annual events in the field of evolutionary computation — the Genetic and Evolutionary Computation Conference[1]—boasts a competition that awards prizes to human-competitive results: The HUMIES. As noted at the competition site (Koza 2010): “Techniques of genetic and evolutionary computation are being increasingly applied to difficult real-world problems—often yielding results that are not merely interesting, but competitive with the work of creative and inventive humans.”

To set the stage for our current work we provide examples of HUMIE winners in two important areas: pure mathematics and games.

[1] see `sigevo.org`

Pure Mathematics

Spector has produced significant new results in the application of genetic programming to mathematics, in collaborative work with Distinguished Professor of Mathematics David M. Clark, at the State University of New York at New Paltz. Together with two Hampshire College undergraduates and one Hampshire alumnus, they applied genetic programming to a problem in pure mathematics, in the study of finite algebras.

Algebraists have been looking at finding "terms" that represent specific functions in specific algebras for several decades, with particular interest attaching to the discovery of terms for Mal'cev functions (the significance of which was first made clear in 1954), Pixley functions (1963), the discriminator function (1970), and majority functions (1975). The most effective methods previously developed for finding these terms are uniform search (including exhaustive search and random search) and construction via the primality theorem. In exhaustive search terms are enumerated systematically from smallest to largest, while in random search terms within a range of sizes are generated in random order. Exhaustive search will always produce the smallest term of the required type if such a term exists, but it requires astronomical amounts of time, except for the very smallest algebras or the very simplest terms. Random search has similarly problematic performance characteristics but without any guarantees concerning size or success. Construction via the primality theorem gives the most time efficient method known to describe these terms that applies to any primal algebra, but except for the very smallest algebras the terms it produces have astronomical length.

Spector and colleagues documented the application of genetic programming to these term-finding problems, producing human-competitive results in the discovery of particular algebraic terms (e.g., discriminator, Pixley, majority, and Mal'cev terms) and showing that genetic programming exceeded the performance of every prior method of finding these terms in either time or size by several orders of magnitude (Spector et al. 2008). This result earned the gold medal in the 5th Annual HUMIES Awards for Human-Competitive Results Produced by Genetic and Evolutionary Computation, held at the 2008 Genetic and Evolutionary Computation Conference. Subsequently, this work led to the development of new mathematical theory that has been published independently (Clark 2013).

Games

Ever since the dawn of artificial intelligence in the 1950s, games have been part and parcel of this lively field. In 1957, a year after the Dartmouth Conference that marked the official birth of AI, Alex Bernstein designed a program for the IBM 704 that played two amateur games of chess. In 1958, Allen Newell, J. C. Shaw, and Herbert Simon introduced a more sophisticated chess program (beaten in thirty-five moves by a 10-year-old beginner in its last official game played in 1960). Arthur L. Samuel of IBM spent much of the fifties working on game-playing AI programs, and by 1961 he had a checkers program that could play at the master's level. In 1961 and 1963 Donald Michie described a simple trial-and-error learning system for learning how to play Tic-Tac-Toe (or Noughts and Crosses) called MENACE (for Matchbox

Educable Noughts and Crosses Engine). These are but examples of highly popular games that have been treated by AI researchers since the field's inception.

Why study games? On this matter Susan L. Epstein wrote:

> There are two principal reasons to continue to do research on games... First, human fascination with game playing is long-standing and pervasive. Anthropologists have catalogued popular games in almost every culture... Games intrigue us because they address important cognitive functions... The second reason to continue game-playing research is that some difficult games remain to be won, games that people play very well but computers do not. These games clarify what our current approach lacks. They set challenges for us to meet, and they promise ample rewards (Epstein 1999).

Studying games may thus advance our knowledge in both cognition and artificial intelligence, and, last but not least, games possess a competitive angle which coincides with our human nature, thus motivating both researcher and student alike.

Over the past 7 years Sipper has done extensive research in the area of games (Sipper et al 2007; Hauptman and Sipper 2005b, a; Hauptman and Sipper 2007b; Azaria and Sipper 2005a, b; Benbassat and Sipper 2010; Hauptman and Sipper 2007a; Hauptman et al. 2009; Shichel et al. 2005), which culminated in his recent book, "Evolved to Win" (Sipper 2011) (see also www.moshesipper.com/games). Among the games successfully tackled are: chess, backgammon, checkers, Reversi, Robocode (tank-war simulation), Rush Hour, and FreeCell. These exhibit the full range from two-player, full-knowledge, deterministic board games, through stochastic, simulation-based games, to puzzles.

A recent line of research has attempted to build a more general form of evolution-based game intelligence by employing a structure known as a policy, which is an ordered set of search-guiding rules (Hauptman et al. 2009; Elyasaf et al. 2012). Policies are complex structures that allow one to define specific conditions under which certain actions are performed. They might specify, for example, that a certain stratagem for solving a puzzle becomes relevant when certain conditions hold. The combination of policies and evolution might just prove powerful enough to set up a general "strategizing machine" (Sipper et al. 2007), i.e., one able to automatically evolve successful game strategies given a description of the game in question. Sipper's work on games has garnered five HUMIE awards to date.

9.3 A Compendium of a Decade's Worth of HUMIE Winners

The main intention behind analyzing a decade's worth of HUMIE winners is to be able to determine whether there were any particular aspects that were similar across the various domains that the winners covered. In 2005, John R. Koza, Sameer H. Al-Sakran, and Lee W. Jones (Koza et al. 2005) did some preliminary analysis, looking for cross-domain features of programs evolved using Genetic Programming. We intend to do something similar, but do so from the specific point of view of trying to identify the aspects of the applications (of any evolutionary computation technique, and not just Genetic Programming) that make them human-competitive.

With the HUMIES over a decade old now, we also have a larger set of results to analyze relative to the number of results analyzed in the original Koza paper (Koza et al. 2005).

The actual HUMIES competition is designed to recognize work that has already been published that holds its own against humans in one of many ways. For example, a result produced by EC that reproduces a past patent, or qualifies as a new patentable invention is considered human-competitive. Similarly, a result that is publishable in its own right as a new scientific result (notwithstanding the fact that the result was mechanically created) is also considered human competitive (Koza 2010). Table 9.1 lists, in detail, the various criteria for a program to be considered human-competitive, and also supplies a count of the number of HUMIE winners that have matched each criterion in the past decade of the HUMIES.

The 42 HUMIE winners of the past decade are listed in Table 9.2. In addition to a very brief description of the winning entry and its author(s), the table includes the specific algorithm used by that entry, where GP is Genetic Programming, GA is Genetic Algorithms, ES refers to Evolutionary Strategies, DE refers to Differential Evolution and GBML refers to Genetics Based Machine Learning. A special category for "noise" is also included. An entry is marked as "noisy" if the data that was used to evolve the solution may inherently have some noise, such as data collected from say, a physical measurement, where the source of the noise is the measurement error. An example of an entry that does not have any noise would be trying to perform symbolic regression to fit a curve that is already known mathematically. Since the data that must be fit already has a known mathematical function, there's no real noise involved here as far as the data points that the program that does the regression sees — all data points are perfectly accurate. In our analysis of the HUMIES (Tables 9.1, 9.2, 9.3, 9.4, 9.5), we explicitly chose to ignore whether the entry won a gold, silver, or bronze award, since we believe that this is insignificant to the analysis because an entry that has won any award at all is necessarily human-competitive.

9.4 Lessons Learned

First and foremost, we note that techniques from evolutionary computation have been used to solve problems from a very wide variety of domains in a human competitive way ; the past 10 years of HUMIES awards alone have winners that have solved problems in a human-competitive way in 21 different domains (see Table 9.4). This clearly suggests that techniques based on EC are rather widely applicable, and are not confined to specific fields.

Second, Genetic Programming (GP) and Genetic Algorithms (GAs) certainly seem to be winning strategies at the HUMIES, with 22 papers based on GP and 15 papers based on GA's having won the HUMIES so far (See Table 9.3). In other words, a combined 37 papers out of the 42 overall HUMIE winners, or roughly 88 % of the winners, used either GP or GAs.

Table 9.1 Categories A-H, with a count of the number of HUMIE winners so far winning in this category, and a description of what each category means. Categories and Full Descriptions were obtained from the HUMIES website (Koza 2010)

Category	Brief description	Count	Full description
A	Patented invention	10	The result was patented as an invention in the past is an improvement over a patented invention or would qualify today as a patentable new invention
B	Equal to accepted scientific result	20	The result is equal to or better than a result that was accepted as a new scientific result at the time when it was published in a peer-reviewed scientific journal
C	Could be put in archive of results	8	The result is equal to or better than a result that was placed into a database or archive of results maintained by an internationally recognized panel of scientific experts
D	Publishable as new scientific result	29	The result is publishable in its own right as a new scientific result independent of the fact that the result was mechanically created
E	Best incremental solution	25	The result is equal to or better than the most recent human-created solution to a long-standing problem for which there has been a succession of increasingly better human-created solutions
F	Achievement in field at time of discovery	25	The result is equal to or better than a result that was considered an achievement in its field at the time it was first discovered
G	Indisputable difficulty	26	The result solves a problem of indisputable difficulty in its field
H	Human competition contender	9	The result holds its own or wins a regulated competition involving human contestants (in the form of either live human players or human-written computer programs)

Next, the problem "type" analysis from Table 9.5 seems to show an interesting trend, with a lot of the problems that evolutionary computation (EC) seems to solve human-competitively being design problems. We use the term design in a rather broad way, to include both designing concrete entities such as an antenna, as well as designing more subtle entities, such as say, designing a winning strategy for a game. This leads us to note that EC may be particularly well suited to designing new

Table 9.2 The 42 HUMIE winners of the past decade

Entry	First author	Alg	Type	Year	N	Setting	Application area	Technique	A	B	C	D	E	F	G	H
Antanae	Lohn	GP	Design	2004	X	Government	Electrical engineering, Antennas	Developmental GP				X	X		X	
Quantum	Spector	GP	Programming	2004		Academia	Quantum	Stack-based, Developmental		X		X				
SAT GP heuristics	Fukunaga	GP	Design	2004		Academia	Optimization	Strongly typed GP				X				
Kinematic machine straight line	Lipson	GP	Design	2004		Academia	Mechanical engineering	Developmental GP					X			
Organization Design Optimization	Khosraviani	GP	Optimization	2004		Academia	Operations research	Standard GP				X		X	X	
Circuit design	Stoica	GA	Design	2004	X	Government	Electronics	Mixtrinsic evolution (SW & HW)	X			X				
2D Photonic crystals	Preble	GP	Design	2005		Academia	Photonics	Tree and bitmap representations		X		X	X			
Quantum attosecond dynamics	Bartels	ES	Optimization	2005	X	Academia	Quantum	Standard ES				X	X	X		
Optical lens systems	Koza	GP	Design	2005		Industry	Optics	Developmental GP	X				X	X	X	

Alg Algorithm, *N* Noisy data, A-H are defined in Table 9.1.

Table 9.2 (continued)

Entry	First author	Alg	Type	Year	N	Setting	Application area	Technique	A	B	C	D	E	F	G	H
Quantum fourier transform algorithm	Massey	GP	Design	2005		Academia	Quantum	Developmental GP		X				X	X	
Assembly programs	Edgar	GP	Programming	2005		Academia	Software engineering	Micro GP			X					X
Space systems design	Terrile	GA	Optimization	2005		Government	Mechanical engineering						X	X	X	
Game playing	Sipper	GP	Design	2005		Academia	games	standard GP								X
Image compression	Grasemann	GA	Design	2005		Academia	Image processing	Coevolutionary		X			X	X	X	
Sinusoidal oscillators	Aggarwal	GA	Design	2006		Academia	Electronics	GA		X		X	X	X		
Photochemistry	Sastry	GA	Optimization	2006		Academia	Chemistry	Multi-objective GA		X	X	X	X			
Ellipse detection	Yao	GA	Classification	2006	X	Academia	Image processing	Multi-population	X	X				X		
Interest point detection	Olague	GP	Classification	2006	X	Government	Computer vision	Standard GP		X	X	X	X	X	X	
Polymer optical fibres	Manos	GA	Design	2007		Academia	Polymers	Developmental GA	X	X		X	X			

Table 9.2 (continued)

Entry	First author	Alg	Type	Year	N	Setting	Application area	Technique	A	B	C	D	E	F	G	H
Mate-In-N chess problem	Sipper	GP	Design	2007		Academia	Games	Koza-style GP		X		X		X	X	X
Diagnosing prostate cancer	Llor	GBML	Classification	2007	X	Academia	Medicine			X		X	X			
Automated alphabet reduction method	Bacardit	GA	Clustering	2007		Academia	Biology	Extended Compact Genetic Algorithm (EDA)		X		X	X		X	
Finite algebras	Spector	GP	Regression	2008		Academia	Mathematics	Stack-based, Developmental		X		X	X	X	X	
RTL benchmark circuits	Pecenka	GA	Design	2008		Industry, Academia	Electronics	Non-binary GA				X			X	
Evolving automatic defect classification	Glazer	GA	Classification	2008		Industry, Academia	Electronics	Standard GA	X			X	X	X	X	
Software patches	Forrest	GP	Programming	2009		Academia	Software engineering	AST with weighted program path							X	
User identification on smart phones	Shahzad	GA	Classification	2009		Academia	Security	GA with particle swarm optimization	X				X	X	X	

Table 9.2 (continued)

Entry	First author	Alg	Type	Year	N	Setting	Application area	Technique	A	B	C	D	E	F	G	H
GP-Rush - Rush hour puzzle	Hauptpman	GP	Design	2009		Academia	Games	Policy-based GP		X		X		X	X	X
Descriptor operators	Perez	GP	Design	2009		Government	Computer vision	Standard GP	X	X	X	X	X	X	X	
Protein structure prediction	Krasnogor	GP	Regression	2010	X	Academia	Biology	Standard					X	X		X
Domain-Independent satisficing planning	Pierre	MH	Planning	2010		Academia	Planning			X		X	X		X	
Solving iterated functions using GP	Schmidt	GP	Regression	2010		Academia	Mathematics	Symbolic regression algorithm					X	X	X	
Mixed-Integer evolution stategies-medical images	Thomas	ES	Optimization	2010	X	Academia	Image processing, Medicine	Mixed-Integer evolution strategies					X	X	X	
FreeCell	Elyasaf	GA	Design	2011		Academia	Games	Standard GA		X		X		X	X	X

Table 9.2 (continued)

Entry	First author	Alg	Type	Year	N	Setting	Application area	Technique	A	B	C	D	E	F	G	H
Verficiation algorithm for evolvable hardware	Lukas	GP	Optimization	2011		Academia	Electronics	Cartesian GP				X	X		X	
Optimal broad-band Stokes/Mueller polarimeter	Letnes	GA	Design	2011		Academia	Optics	Standard GA	X	X	X	X	X	X	X	
Game design	Browne	GP	Design	2012		Academia	Games	Evolving rule trees			X	X		X		
Automated Probe Microscopy	Woolley	GA	Optimization	2012		Academia	Mechanical engineering	Cellular GA	X			X	X	X	X	X
Automated program repair	Dewey-Vogt	GP	Programming	2012		Academia	Software engineering	GP over AST edit operations				X			X	
FreeCell	Sipper	GP	Design	2013		Academia	Games	Policy-based GP		X		X		X	X	X
Jupiter moon search	Izzo	DE	Design	2013		Government	Mechanical engineering	Self-adaptation differential evolution algorithm, Asynchronous island model			X	X		X		X
Solid state NMR pulse sequences	Bechmann	GA	Optimization	2013		Academia	Physics	Standard GA	X	X	X	X	X	X	X	

Table 9.3 A summary of the algorithms used by the HUMIE winners

Algorithm	Count
Genetic Programming (GP)	22
Genetic Algorithms (GA)	15
Evolutionary Strategies (ES)	2
Differential Evolution (DE)	1
Genetics Based Machine Learning (GBML)	1
Metaheuristic	1

Table 9.4 Categorization of the application domains of the HUMIE winners. Note that some winners may come under multiple application categories. The number in brackets after the application categories denotes the number of HUMIE winners in that particular application category

Application	Count	Application category
Antennas	1	Engineering (19)
Biology	2	Science (7)
Chemistry	1	Science (7)
Computer vision	2	Computer science (7)
Electrical engineering	1	Engineering (19)
Electronics	5	Engineering (19)
Games	6	Games (6)
Image processing	3	Computer science (7)
Mathematics	2	Mathematics (3)
Mechanical engineering	4	Engineering (19)
Medicine	2	Medicine (2)
Operations research	1	Engineering (19)
Optics	2	Engineering (19)
Optimization	1	Mathematics (3)
Photonics	1	Engineering (19)
Physics	1	Science (7)
Planning	1	Computer science (7)
Polymers	1	Engineering (19)
Quantum	3	Science (7)
Security	1	Computer science (7)
Software engineering	3	Engineering (19)

entities from scratch in a human-competitive way. The authors note that classifying problems based on a "type" is a slightly subjective process and that some problems may fit several types at times, but we believe that the above analysis is still sufficient to note how good EC is when it comes to design problems.

Table 9.5 A count of the broad types of problems that the HUMIE winners solved

Problem Type	Count
Classification	5
Clustering	1
Design	20
Optimization	8
Planning	1
Programming	4
Regression	3

Another interesting trend at the HUMIES seems to be the abundance of papers that have combined domain specific knowledge effectively with evolution in a way where evolution helps combine and adapt existing human knowledge in innovative new ways. For example, Stephanie Forrest's paper (Forrest et al. 2009) uses existing human knowledge embedded in non-faulty parts of the code to repair parts of the code that are faulty. Policy based GP (Hauptman et al. 2009; Elyasaf et al. 2012) is another such area where human knowledge is integrated into an EC system that then evolves a solution that is human competitive. This particular trend certainly suggests a rethink of the artificial (intelligence)-to-(human) intelligence (A/I) ratio, suggested by John Koza et al., which states that GP delivers a high amount of artificial intelligence relative to the relatively minimal amount of human intelligence that is put in to the system (Koza et al. 2003). In the context of human-competitive machine intelligence, our analysis suggests that looking for a high A to I ratio is not the best way to seek promising problems. Instead, we suggest that the focus should be on the additional knowledge gained by some automatic technique that makes the system human competitive. To quote Moshe Sipper from his book *Evolved to Win* (Sipper 2011),

> Rather than aiming to maximize A/I we believe the "correct" equation is:
>
> $$A - I \geq M_\varepsilon$$
>
> where M_ε stands for "meaningful epsilon". When wishing to attain machine competence in some real-life, hard-to-learn domain, then, by all means, imbue the machine with as much I(ntelligence) as possible! After all, if imbuing the I reduces the problem's complexity to triviality, then it was probably not hard to begin with. Conversely, if the problem is truly hard, then have man and machine work in concert to push the frontiers of A as far as possible. Thus, it is not max(A/I) that is of interest but the added value of the machine's output: Granting the designer "permission" to imbue the machine with as much I as he can, will it then produce a $\Delta A = A - I$, namely, added intelligence, that is sufficiently meaningful? Even if this meaningful epsilon M_ε is small in (some) absolute terms, its relative value can be huge (e.g., a chip that can pack 1–2 % more transistors, or a game player that is slightly better and thus world champion).

We believe that this approach of looking at the additional intelligence gained by an automated system is crucial not just for Genetic Programming, but for Artificial Intelligence on the whole. We strongly encourage people to build artificial intelligence systems that make as much use of existing human knowledge as possible.

9.5 Concluding Remarks

Overall, analyzing a decade's worth of HUMIES papers seems to suggest that when combined effectively with domain-specific knowledge, GP and GA are approaches to EC that are highly effective in producing human-competitive results, in a very wide array of fields. We strongly encourage further research in the human-competitive domain, particularly with EC approaches such as GA and GP, used in conjunction with human knowledge in the current field. One aspect that we note in particular is the significance of collaboration with experts outside the computer science domain, which leads to several interesting insights in multiple fields.

We would also like to mention the gradual shift from a high Artificial Intelligence to Human Intelligence (A/I) ratio, towards a focus on the additional intelligence gained by using an intelligent system, irrespective of how much human intelligence one supplies to it. One interesting aspect that must be brought up when moving away from high (A/I) is interpretability, particularly when a human is involved in a feedback loop used to improve the system. In her recent work, Cynthia Rudin has been suggesting interpretability as a key feature in several prediction systems, and notes that experts are more likely to use an interpretable system compared to a black box system that they do not understand (Letham et al. 2012; Rudin et al. 2012; Wang et al. 2013). While most of evolutionary computation has historically been using a black-box approach, interpretability might eventually become rather important too (both in EC-based approaches and in other machine learning approaches) to build human-competitive systems, particularly when human experts are involved in both building the system and improving its quality.

In a recent paper Kiri Wagstaff argues that much of current machine learning (ML) research has lost its connection to problems of import to the larger world of science and society (Wagstaff 2012). In reference to the much-used (and perhaps much-abused) UCI archive she eloquently writes,

> "Legions of researchers have chased after the best iris or mushroom classifier. Yet this flurry of effort does not seem to have had any impact on the fields of botany or mycology."

Wagstaff identifies several problems that underlie the "Machine Learning for Machine Learning's Sake" stance, including: overly focusing on benchmark data sets, with little to no relation to the real world; too much emphasis on abstract metrics that ignore or remove problem-specific details, usually so that numbers can be compared across domains; and lack of follow-through:

> "It is easy to sit in your office and run [some] algorithm on a data set you downloaded from the web. It is very hard to identify a problem for which machine learning may offer a solution, determine what data should be collected, select or extract relevant features, choose an appropriate learning method, select an evaluation method, interpret the results, involve domain experts, publicize the results to the relevant scientific community, persuade users to adopt the technique, and (only then) to truly have made a difference."

She argues for making machine learning matter: asking how one's work impacts the original problem domain; greater involvement of domain experts; and considering the potential impact on society of a problem one elects to work on. She proposes a number of admirable Impact Challenges as examples of machine learning that matters, e.g., a law passed or a legal decision made that relies on the result of an ML analysis, and $ 100 M saved through improved decision making provided by an ML system.

We think that Wagstaff actually bolsters research into human-competitive results produced by EC. Despite her opining that, "human-level performance is not the gold standard. What matters is achieving performance sufficient to make an impact on the world", we think that the problems in Table 9.2 are very strongly coupled to the real world. Indeed, most, if not all, of them have involved expertise (and often actual experts) in a real-world problem domain, and the competition itself sets out to underscore the impact of such research on society at large. Thus, the HUMIE winners may all be unknowingly responding to Wagstaff's challenge, creating machine learning that matters.

Acknowledgments This material is based upon work supported by the National Science Foundation under Grants No. 1017817, 1129139, and 1331283. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Azaria Y, Sipper M (2005a) GP-gammon: genetically programming backgammon players. Genet Program Evol Mach 6(3):283–300. doi:10.1007/s10710-005-2990-0 http://www.cs.bgu.ac.il/sipper/papabs/gpgammon.pdf. Accessed 12 Aug 2005

Azaria Y, Sipper M (2005b) Using GP-gammon: using genetic programming to evolve backgammon players. In: Keijzer M, Tettamanzi A, Collet P, van Hemert JI, Tomassini M (eds) Proceedings of the 8th European conference on genetic programming. Lecture notes in computer science, vol 3447. Springer, Lausanne, pp 132–142. doi:10.1007/b107383

Barnum H, Bernstein HJ, Spector L (2000) Quantum circuits for OR and AND of ORs. J Phys A: Math Gen 33(45):8047–8057. (http://hampshire.edu/lspector/pubs/jpa.pdf)

Benbassat A, Sipper M (2010) Evolving lose-checkers players using genetic programming. In: IEEE conference on computational intelligence and game, IT University of Copenhagen, Denmark, pp 30–37. doi:10.1109/ITW.2010.5593376 http://game.itu.dk/cig2010/proceedings/papers/cig10-005-011.pdf

Benbassat A, Elyasaf A, Sipper M (2012) More or less? two approaches to evolving game-playing strategies. In: Riolo R, Vladislavleva E, Ritchie MD, Moore JH (eds) Genetic programming theory and practice X, genetic and evolutionary computation. Springer, Ann Arbor, chap 12, pp 171–185 doi:10.1007/978-1-4614-6846-2-12 http://dx.doi.org/10.1007/978-1-4614-6846-2-12

Clark DM (2013) Evolution of algebraic terms 1: term to term operation continuity. Int J Algebra Comput 23(05):1175–1205 doi:10.1142/S0218196713500227 http://www.worldscientific.com/doi/abs/10.1142/S0218196713500 227, http://www.worldscientific.com/doi/pdf/10.1142/S0218196713500227

Elyasaf A, Hauptman A, Sipper M (2012) Evolutionary design of FreeCell solvers. IEEE Trans Comput Intell AI Games 4:270–281 doi:10.1109/TCIAIG.2012.2210423 http://ieeexplore.ieee.org/xpls/abs-all.jsp?arnumber=6249736

Epstein SL (1999) Game playing: the next moves. In: AAAI/IAAI, pp 987–993

Forrest S, Nguyen T, Weimer W, Le Goues C (2009) A genetic programming approach to automated software repair. In: Raidl G, Rothlauf F, Squillero G, Drechsler R, Stuetzle T, Birattari M, Congdon CB, Middendorf M, Blum C, Cotta C, Bosman P, Grahl J, Knowles J, Corne D, Beyer HG, Stanley K, Miller JF, van Hemert J, Lenaerts T, Ebner M, Bacardit J, O'Neill M, Di Penta M, Doerr B, Jansen T, Poli R, Alba E (eds) GECCO '09: Proceedings of the 11th annual conference on genetic and evolutionary computation, ACM, Montreal, pp 947–954 doi:10.1145/1569901.1570031 http://citeseerx.ist.psu.edu/viewdoc/summary?doi:10.1.1.147.7 651, best paper

Glazer A, Sipper M (2008) Evolving an automatic defect classification tool. In: Giacobini M, Brabazon A, Cagnoni S, Caro GAD, Drechsler R (eds) Applications of evolutionary computing: proceedings of EvoWorkshops 2008. Lecture notes in computer science, vol 4974. Springer, Heidelberg, pp 194–203

Hauptman A, Sipper M (2005a) Analyzing the intelligence of a genetically programmed chess player. In: Rothlauf F (ed) Late breaking paper at Genetic and Evolutionary Computation Conference (GECCO'2005), Washington, DC. http://www.cs.bham.ac.uk/~wbl/biblio/gecco2005lbp/papers/21-hauptmann.pdf

Hauptman A, Sipper M (2005b) GP-endchess: using genetic programming to evolve chess endgame players. In: Keijzer M, Tettamanzi A, Collet P, van Hemert JI, Tomassini M (eds) Proceedings of the 8th European conference on genetic programming. Lecture notes in computer science, vol 3447. Springer, Lausanne, pp 120–131. doi:10.1007/b107383

Hauptman A, Sipper M (2007a) Emergence of complex strategies in the evolution of chess endgame players. Adv Complex Syst 10(Suppl 1):35–59

Hauptman A, Sipper M (2007b) Evolution of an efficient search algorithm for the mate-in-N problem in chess. In: Ebner M, O'Neill M, Ekárt A, Vanneschi L, Esparcia-Alcázar AI (eds) Proceedings of the 10th European conference on genetic programming. Lecture notes in computer science, vol 4445. Springer, Valencia, pp 78–89. doi:10.1007/978-3-540-71605-1-8

Hauptman A, Elyasaf A, Sipper M, Karmon A (2009) GP-rush: using genetic programming to evolve solvers for the Rush Hour puzzle. In: Raidl G, Rothlauf F, Squillero G, Drechsler R, Stuetzle T, Birattari M, Congdon CB, Middendorf M, Blum C, Cotta C, Bosman P, Grahl J, Knowles J, Corne D, Beyer HG, Stanley K, Miller JF, van Hemert J, Lenaerts T, Ebner M, Bacardit J, O'Neill M, Di Penta M, Doerr B, Jansen T, Poli R, Alba E (eds) GECCO '09: Proceedings of the 11th Annual conference on genetic and evolutionary computation, ACM, Montreal, pp 955–962. doi:10.1145/1569901.1570032 http://dl.acm.org/citation.cfm?id=1570032

Koza JR (2008) Human-competitive machine invention by means of genetic programming. Artif Intell Eng Des, Anal Manuf 22(3):185–193. doi:10.1017/S0890060408000127

Koza JR (2010) Human-Competitive Awards: HUMIES. http://www.human-competitive.org. Accessed 6 May 2014

Koza JR, Keane MA, Streeter MJ, Mydlowec W, Yu J, Lanza G (2003) Genetic programming IV: routine human-competitive machine intelligence. Kluwer Academic. http://www.springer.com/us/book/9780387250670

Koza JR, Al-Sakran SH, Jones LW (2005) Cross-domain features of runs of genetic programming used to evolve designs for analog circuits, optical lens systems, controllers, antennas, mechanical systems, and quantum computing circuits. In: Evolvable Hardware, 2005. Proceedings. 2005 NASA/DoD Conference on, IEEE, pp 205–212

Letham B, Rudin C, McCormick TH, Madigan D (2012) Building interpretable classifiers with rules using bayesian analysis. Department of Statistics Technical Report tr609, University of Washington

Rudin C, Waltz D, Anderson RN, Boulanger A, Salleb-Aouissi A, Chow M, Dutta H, Gross P, Huang B, Ierome S, Isaac D, Kressner A, Passonneau RJ, Radeva A, Wu L (2012) Machine learning for the New York City power grid. IEEE Trans Pattern Anal Mach Intell 34(2):328–345

Shichel Y, Ziserman E, Sipper M (2005) GP-robocode: using genetic programming to evolve robocode players. In: Keijzer M, Tettamanzi A, Collet P, van Hemert JI, Tomassini M (eds)

Proceedings of the 8th European conference on genetic programming. Lecture notes in computer science, vol 3447. Springer, Lausanne, pp 143–154. doi:10.1007/b107383

Sipper M (2002) Machine nature: the coming age of bio-inspired computing. McGraw-Hill, New York

Sipper M (2006) Attaining human-competitive game playing with genetic programming. In: El Yacoubi S, Chopard B, Bandini S (eds) Proceedings of the 7th international conference on cellular automata, for research and industry, ACRI. Lecture Notes in computer science, vol 4173. Springer, Perpignan, p 13. doi:10.1007/11861201-4, invited Lectures

Sipper M (2011) Evolved to Win. Lulu http://www.lulu.com/. Accessed 6 May 2014

Sipper M, Azaria Y, Hauptman A, Shichel Y (2007) Designing an evolutionary strategizing machine for game playing and beyond. IEEE Trans Syste, Man Cybern, Part C: Appl Rev 37(4):583–593. doi:10.1109/TSMCC.2007.897326

Spector L (2004) Automatic quantum computer programming: a genetic programming approach, genetic programming, vol 7. Kluwer Academic, Boston. http://www.wkap.nl/prod/b/1-4020-7894-3

Spector L, Clark DM, Lindsay I, Barr B, Klein J (2008) Genetic programming for finite algebras. In: Keijzer M, Antoniol G, Congdon CB, Deb K, Doerr B, Hansen N, Holmes JH, Hornby GS, Howard D, Kennedy J, Kumar S, Lobo FG, Miller JF, Moore J, Neumann F, Pelikan M, Pollack J, Sastry K, Stanley K, Stoica A, Talbi EG, Wegener I (eds) GECCO '08: Proceedings of the 10th annual conference on genetic and evolutionary computation, ACM, Atlanta, GA, USA, pp 1291–1298. doi:10.1145/1389095.1389343 http://www.cs.bham.ac.uk/~wbl/biblio/gecco2008/docs/p1291.pdf

Wagstaff K (2012) Machine learning that matters. arXiv preprint arXiv:12064656

Wang T, Rudin C, Wagner D, Sevieri R (2013) Detecting patterns of crime with series finder. In: Proceedings of the European conference on machine learning and principles and practice of knowledge discovery in databases (ECMLPKDD 2013)

Karthik Kannappan is a Master's Student in Computer Science at the University of Massachusetts, Amherst, currently working with Prof. Lee Spector at the computational intelligence laboratory. His research interests include automatic software synthesis, affective computing, and software engineering.

Lee Spector is a Professor of Computer Science at Hampshire College and an Adjunct Professor of Computer Science at the University of Massachusetts, Amherst. He received a B.A. in Philosophy from Oberlin College in 1984 and a Ph.D. in Computer Science from the University of Maryland in 1992. He is the Editor-in-Chief of the journal Genetic Programming and Evolvable Machines and a member of the editorial board of Evolutionary Computation. He is also a member of the SIGEVO executive committee and he was named a Fellow of the International Society for Genetic and Evolutionary Computation.

Moshe Sipper is a Professor of Computer Science at Ben-Gurion University, Israel.

Thomas Helmuth is a Ph.D. Candidate in Computer Science at University of Massachusetts, Amherst. He received his B.A. from Hamilton College in 2009 and his M.S. from University of Massachusetts, Amherst in 2012.

William La Cava is a PhD student at University of Massachusetts, Amherst. He received his B.S. and M.Eng. from Cornell University in 2009 and 2010, and went on to work at the National Wind Technology Center in Boulder, Colorado. William is interested in automatic system identification and automatic control design for complex systems. As an NSF student fellow in the UMass IGERT Offshore Wind Energy Program, he is developing symbolic regression methods to create data-based models of offshore wind turbines and avian behavior.

Jake Wisdom was a 2014 graduate of Hampshire College in Amherst MA, where he studied 3D animation, visual effects and computer graphics. He is currently working towards his MFA in visual effects at Savannah College of Art and Design with his primary interests in procedural animation and simulation.

Omri Bernstein is a recent Hampshire College alumnus now working as an instructor at Fullstack Academy of Code in New York City. While at Hampshire, he studied the applications of genetic programming to biology, as well as the applications of biological principles to genetic programming. For his senior thesis he developed an educational quantum computer programming environment, and he is currently working on the evolution of quantum algorithms.