TUTORIAL

Evolutionary algorithms and machine learning: Synergies, Challenges and Opportunities

GIOVANNI SQUILLERO, Polytechnic of Turin, Turin, TO, Italy

ALBERTO PAOLO TONDA, National Research Institute for Agriculture, Food and Environment, Paris, Ile-de-France, France

Published: 08 July 2020

Citation in BibTeX format

GECCO '20: Genetic and Evolutionary Computation Conference
July 8 - 12, 2020
Cancún, Mexico

Conference Sponsors:
SIGEVO

GECCO '20: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (July 2020)
https://doi.org/10.1145/3377929.3389863
ISBN: 9781450371278

Evolutionary Algorithms and Machine Learning

Synergies, Challenges and Opportunities

Giovanni Squillero
Politecnico di Torino
Torino, Italy
squillero@polito.it

Alberto Tonda
INRAe
Paris, France
alberto.tonda@inrae.fr

GECCO '20 Companion, July 8–12, 2020, Cancún, Mexico

ACM ISBN 978-1-4503-7127-8/20/07...$15.00
https://doi.org/10.1145/3377929.3389863

Instructors

- **Giovanni Squillero** is an associate professor of computer science at Politecnico di Torino, Italy. His research focuses on approximate optimization, mixing the whole spectrum of bio-inspired metaheuristics, computational intelligence, and selected topics from machine learning. Up to April 2020, he is credited as an author in 3 books, 33 journal articles, 10 book chapters, and 146 papers in conference proceedings; he is also listed among the editors in 15 volumes.
- **Alberto Tonda** is a permanent researcher at INRAe and Université Paris-Saclay, Paris, France. His research interests include semi-supervised modeling of complex systems, evolutionary optimization and machine learning. He is currently chair of COST Action CA15118 FoodMC, a European networking project on in-silico modelling in food science. Alberto Tonda authored 29 scientific papers published in refereed international journals, 2 books and over 50 contributions in international conferences.

What you should get out of this talk

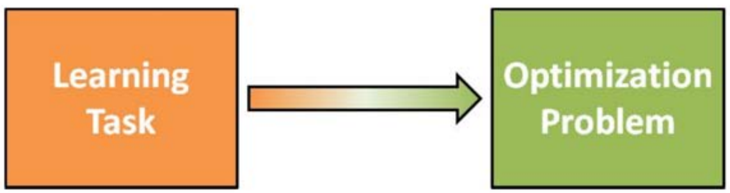
- ML and EC share a common ancestor, there is evidence for interbreeding between the two fields, and eventually diverged for good, practical reasons
- Synergies
  - Success stories of ML used in EC
  - Success stories of EC used in ML
- Challenges
  - Evolutionary Machine Learning
- Opportunities
  - How could ML and EC still be beneficial to one another?

Quick (and dirty) summary of Machine Learning

- You have data, collected from a specific phenomenon
- You would like to develop a predictive model for the phenomenon
- Classical approach
  - Develop ad-hoc algorithm with human knowledge
- Machine Learning (ML) approach
  - Use generic (existing) algorithm, able to…
  - …extract and reproduce information from data
  - …provide predictions for unseen data
  - Basically, the predictive model *learns* from available (training) data

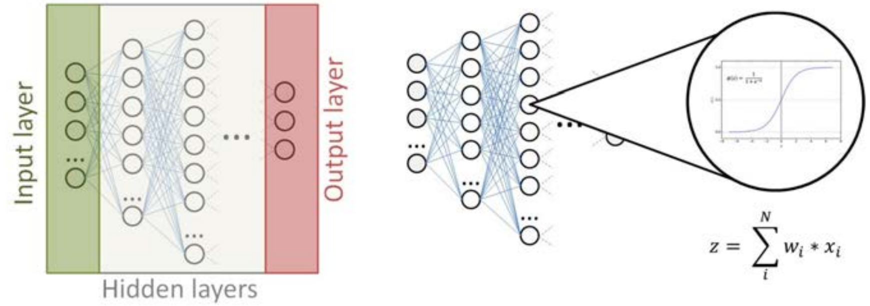
Quick (and dirty) summary of Machine Learning

- Restate **learning task** as an **optimization problem**
- Solve the optimization problem relying on data



Quick (and dirty) summary of Machine Learning

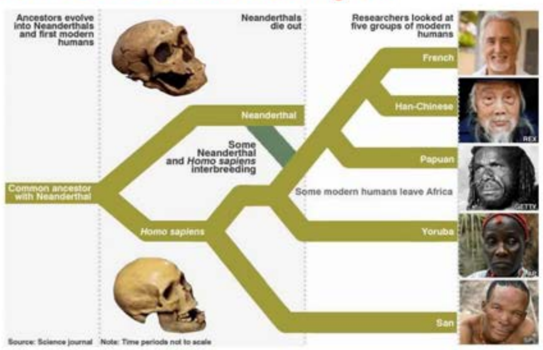
- We assume *some* background knowledge of Neural Networks



Outline

- Introduction
  - A common origin?
  - Shared themes and crossways
  - Popular moments in AI and EC
- Synergies
  - EA can solve ML problems
  - Neuroevolution
  - Discovering coresets
  - Adversarial ML
  - Reinforcement learning and Competitive Co-evolution
- Challenges
  - Performance
  - Black magic/Trustworthiness
  - Large vs. Small Data
  - Number of features
  - Overfitting
- Opportunities
  - Capacity vs Fitting
  - Stochastic optimization in ML
  - EA can solve ML problems
  - Toward white-box ML (Explainable AI)
  - Exploring embeddings

A common origin?
Shared themes and crossways?



Green, Richard E., et al. "A draft sequence of the Neandertal genome." *science* 328.5979 (2010): 710-722.

A common origin?

- Both ML and EC scholars point to the **very same** paper as the **starting point** of their fields:
  - Turing AM. Computing "Machinery and Intelligence". *Mind*. 1950 Oct 1;LIX(236):433–60
- The **term** "Machine Learning" was popularized by Arthur Samuel in a paper describing an **evolutionary approach** for playing checkers
  - Samuel AL. *Some Studies in Machine Learning Using the Game of Checkers. IBM Journal of Research and Development*. 1959 Jul;3(3):210–29.
- Seminal works in EC explicitly refer to the "Machine Learning" keyword
  - E.g., Goldberg DE, Holland JH. "Genetic Algorithms and Machine Learning". *Machine Learning*. 1988 Oct 1;3(2):95–9
  - Goldberg DE. *Genetic Algorithms in Search, Optimization and Machine Learning*. 1st ed. USA: Addison-Wesley Longman Publishing Co., Inc.; 1989. 372 p.

Shared themes and crossways

- Learning without the need of human expertise
- DeepMind's AlphaZero
  - "Mastering chess and shogi **by self-play** with a general reinforcement learning algorithm"
- Fogel's Blondie24
  - "Evolving neural networks to play checkers **without relying on expert knowledge**"
- "Overlapping subsquares" vs. "Convolutional neural network"

Silver D, Hubert T, Schrittwieser J, Antonoglou I, Lai M, Guez A, et al. "Mastering chess and shogi by self-play with a general reinforcement learning algorithm". arXiv:171201815 [cs]. 2017 Dec.

Chellapilla K, Fogel DB. "Evolving neural networks to play checkers without relying on expert knowledge". IEEE Transactions on Neural Networks. 1999 Nov;10(6):1382–91.

Shared themes and crossways

- Some boosting methods creates an ensemble of learners, **removing** points that have been **already solved** and **focusing** on the **remaining ones**
- Some EAs that target the creation of multiple populations for cumulatively solving a problem **remove** the part of the problem that have been **already solved** and **focus** on the **remaining ones**

Hansen, 2009. *Benchmarking a BI-Population CMA-ES on the BBOB-2009 Function Testbed.* GECCO'09

Freund, Y., & Schapire, R. E. 1995. *A decision-theoretic generalization of on-line learning and an application to boosting.* Springer, Berlin, Heidelberg.

Shared themes and crossways

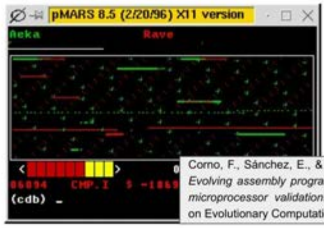
- **Reinforcement Learning** in ML is important and likely to play a pivotal role in the future
  - AlphaZero can be described as "a generic **reinforcement learning** algorithm"
  - Deep Reinforcement Learning (DRL) and Deep **Q-Networks** (DQNs) were demonstrated able to achieve impressive results
  - Multi-Agent RL (MARL) and Multi-Agent Deep RL (MADRL) are emerging techniques to handle problems where **multiple agents** need to communicate and cooperate
- **Reinforcement Learning** did play a pivotal role in EC
  - Holland's Learning Classifier Systems (LCS) are rule-based systems able to evolve and generalize set of **q-Learning-like** rules
  - Cooperative Coevolution is a well-known technique in EC to handle problems where **multiple agents** need to communicate and cooperate

Shared themes and crossways

- Reinforcement Learning shares similarities with Co-evolution
  - In (Deep) RL, agents are trained on data, play against each other
  - Their games generate new data, that is then used to train the agents even more
  - In modern applications, agents are deep NNs, that replace the classical tables
- (Competitive) Co-evolution for games
  - Each individual in the population represents a different style of play
  - Individuals play against each other, obtaining a relative fitness score
  - The "learning" is modeled as the individuals' genome
  - Successful individuals "hand down" part of their style of play to children

Shared themes and crossways

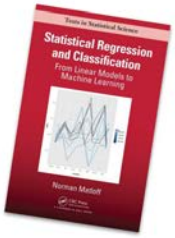
- Example of competitive co-evolution for games: Core Wars
  - A player in the game is a program in Redcode (similar to assembly)
  - Player and opponent are executed one line at a time, alternatively
  - Objective of the game is to force opponent to execute a non-valid instruction
  - Using competitive co-evolution, a Redcode program (WhiteNoise) was created
  - WhiteNoise was the champion of a competitive hill for months
  -



Corno, F., Sánchez, E., & Squillero, G. (2005). *Evolving assembly programs: how games help microprocessor validation.* IEEE Transactions on Evolutionary Computation, 9(6), 695-706.

Shared themes and crossways

- Genetic Programming has been used for **Symbolic Regression** since the 1990s
- **Regression** is a popular application in modern ML



Popular moments in AI / ML

- **1997**: DeepBlue defeated then-reigning world chess champion Garry Kasparov in a six-game match
- **2011**: Watson defeated two renowned champions at Jeopardy
- **2016**: AlphaGo sealed 4-1 victory over Go grandmaster Lee Sedol

Popular moments in AI / ML

- **1997**: DeepBlue defeated then-reigning world chess champion Garry Kasparov in a six-game match
- **2011**: Watson defeated two renowned champions at Jeopardy
- **2012**: AlexNet achieved an astonishing top-5 error of 15.3% in ImageNet Large Scale Visual Recognition Competition
- **2016**: AlphaGo sealed 4-1 victory over Go grandmaster Lee Sedol

Popular moments in EC

- **1964**: Der Spiegel published an article on using Evolutionary Computation for solving aerodynamic problems
- **2017**: Facebook admits using an evolutionary tool for uncovering critical software bugs

Synergies

Synergies — EA can solve ML problems

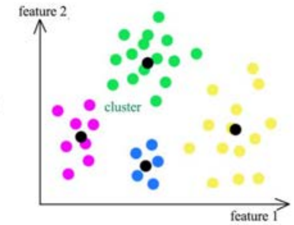
- Problems in ML can have vast, irregular search spaces
- Current solutions are hand-designed or heuristic
- EAs can provide alternative, non-human, (possibly) better solutions!

Synergies — Neuroevolution

- Exploit EC to generate/tweak hyperparameters of neural networks (NNs)
  - Number of neurons, number of layers, types of layers, learning rate, etc.
  - Currently practitioners copy what worked (e.g. ImageNet) and modify it manually
  - Neuroevolution uses EAs to explore space of possible NN topologies (see other tutorial)
- **NEAT** (and **HyperNEAT**, and **EXAMM**)
  - Stanley KO, Miikkulainen R. “Evolving Neural Networks through Augmenting Topologies”. *Evolutionary Computation*. 2002 Jun;10(2):99–127.
  - D’Ambrosio DB, Gauci J, Stanley KO. “HyperNEAT: The First Five Years”. In: *Growing Adaptive Machines: Combining Development and Learning in Artificial Neural Networks*. Berlin, Heidelberg: Springer; 2014.
  - Desell T, ElSaid A, Ororbia AG. “An Empirical Exploration of Deep Recurrent Connections Using Neuro-Evolution”. In: *Applications of Evolutionary Computation*. Cham: Springer International Publishing; 2020. p. 546–61.

Synergies — Finding coresets with EAs

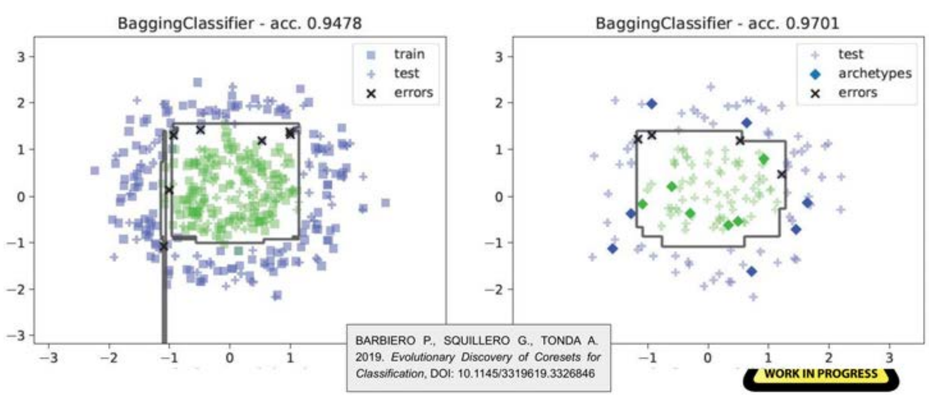
- Issues with large ML datasets (in number of samples and features)
  - Hard to interpret for humans
  - Training a ML algorithm on the whole dataset takes a considerably long time (or it is outright impossible)
- Coresets
  - A **coreset** is the minimum number of training samples that does not lower performance of ML techniques “too much”
  - They represent the “typical samples” for all the classes (for classification)
  - They can be samples already in the dataset, or virtual (also called **prototypes**)



Synergies — Finding coresets with EAs

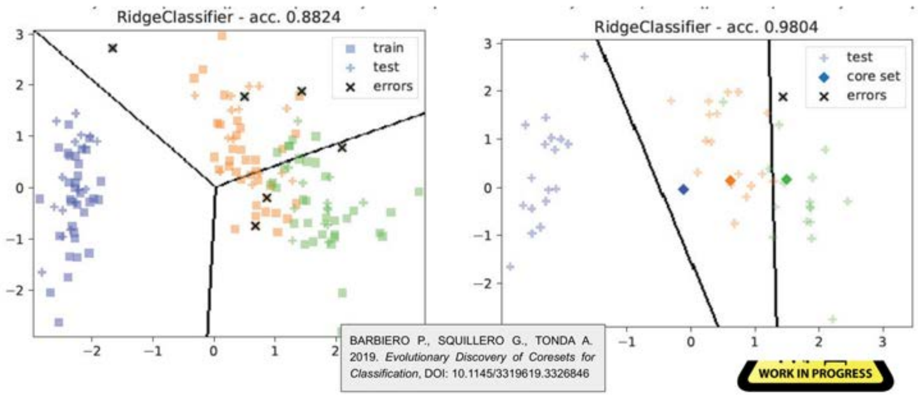
- The search space for coresets is vast
  - Variable (unknown) number of samples in the coreset
  - Consider all possible samples in the training set + prototypes (virtual samples)
- The problem might be multi-objective!
  - As removing training samples will likely lower the performance of the ML algorithm
  - There are two conflicting objectives: lower number of samples in coreset, keep error low
- Finding coresets with EAs
  - Individual representation: list of indexes, referring to samples in training set
  - OR matrix of variable size, where each line represent a prototype (virtual sample)
  - Fitness function: average performance of a ML algorithm in a cross-validation

Synergies — Finding coresets with EAs



BARBIERO P., SQUILLERO G., TONDA A. 2019. *Evolutionary Discovery of Coresets for Classification*, DOI: 10.1145/3319619.3326846

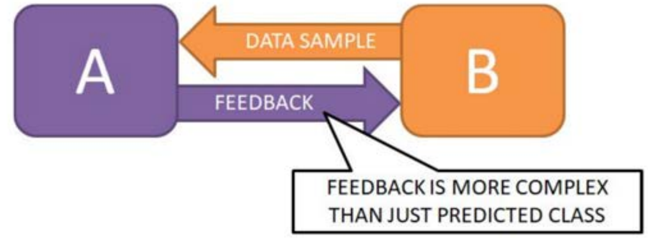
Synergies — Finding coresets with EAs



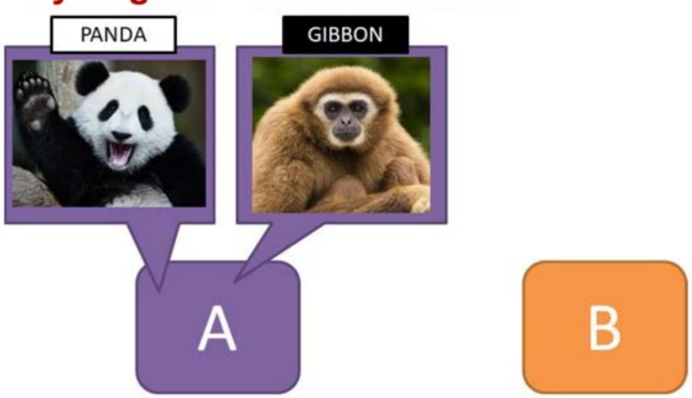
BARBIERO P., SQUILLERO G., TONDA A. 2019. *Evolutionary Discovery of Coresets for Classification*, DOI: 10.1145/3319619.3326846

Synergies — Adversarial ML

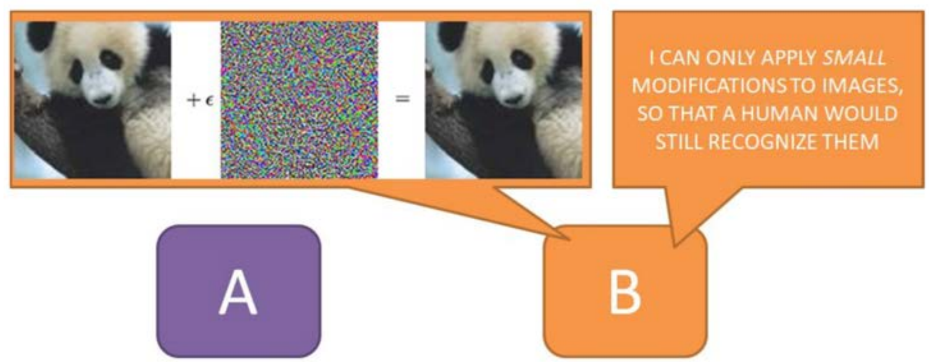
- Adversarial ML (sometimes called "Generative")
  - Once a ML model (e.g. a classifier) is trained, find **counterexamples** badly classified
  - Counterexamples can provide more **insight on the inner working** of the algorithm
  - Adversarial ML pits a second ML algorithm AGAINST the model
  - The second ML algorithm generates samples, using output of the trained model as feedback



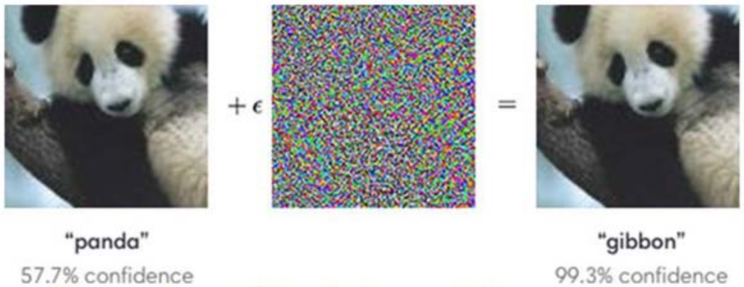
Synergies — Adversarial ML



Synergies — Adversarial ML



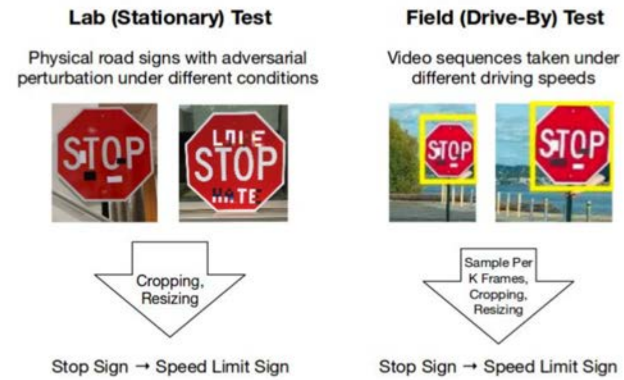
Synergies — Adversarial ML



https://openai.com/blog/adversarial-example-research/

Synergies — Adversarial ML

K. EYKHOLT et al. 2018. *Robust Physical-World Attacks on Deep Learning Models*, https://arxiv.org/pdf/1707.08945.pdf



Synergies — Adversarial ML

J. SU et al. 2019. *One pixel attack for fooling deep neural networks*, IEEE TEC https://arxiv.org/abs/1710.08864

- Adversarial ML is an optimization problem
  - Genome is a series of modifications applied to images
  - Fitness is feedback from the trained ML model (minimize correct class confidence)
  - Search space is vast (all possible samples!)

- EAs can be applied to adversarial ML!
  - A particularly clever example is a ONE-PIXEL adversarial attack!
  - Genome is just the position and permutation of one pixel in an image
  - Fitness is "confidence" (probability) associated to each class
  - Algorithm used was differential evolution

Synergies — Adversarial ML

- Fooling text classification with EAs
  - Li, D., Vargas, D. V., & Kouichi, S. (2019, June). *Universal Rules for Fooling Deep Neural Networks based Text Classification.* https://arxiv.org/pdf/1901.07132.pdf

- Interesting resources on Adversarial ML
  - Embeddings, https://www.depends-on-the-definition.com/introduction-to-embeddings-with-neural-networks/
  - Image generation, https://thispersondoesnotexist.com/
  - Text generation, https://talktotransformer.com/

Challenges

Challenges — Performance

- The limited acceptance of EC in the industrial world may be explained by its inability to tackle real-size problem
- The time required to produce a reasonable solution is often not acceptable
- Most published studies focus on **toy problem** (most notably, Holland original works)
    - EAs are theoretically parallelizable at the level of generation, allowing an almost-linear increase performances
    - Unlike other methodologies, an EA can be stopped at any moment providing the best solution found so far (trade off time/quality)

Challenges — Black magic/Trustworthiness

- The limited acceptance of EC in the industrial world may also be explained by their inherent **stochasticity**, **non-reproducibility** of the results
    - Yet, many industrial processes are based on random variations or non reproducible
    - … and most EC results are “almost” reproducible
- The relatively slow acceptance of ML in the industrial world may be explained by the difficult **interpretability** of the resulting models
    - Relying on intrinsically stochastic processes like *stochastic gradient descent* is not usually considered a diriment problem
    - Non-interpretable models may be **incorrect**, **biased** or lead tor **unfair** results

Challenges — Large vs. Small Data

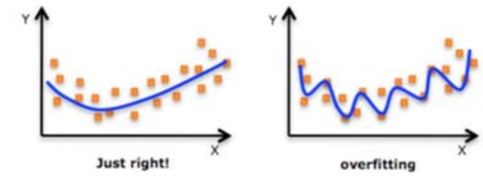
- Traditional ML techniques have been designed in order to process huge amount of data, such as images or documents fetched over the internet
- A growing number of applications require careful analyses of a reduced amount of data that are either scarce or expensive
- ML models need to be tweaked if not completely rethought
    - E.g., Zero-Shot/N-Shot/Few-Shot learning models

Challenges — Number of features

- High-dimensional spaces are well known to behave differently from low-dimensional ones (***curse of dimensionality***)
- EC/ML tools often need to reduce the number of variables to operate effectively
- **Dimensionality reduction**: the process of reducing the number of variables under consideration
  - Feature selection (e.g., *recursive feature elimination*)
  - Feature extraction (e.g., *principal component analysis*, *latent semantic analysis*)
  - Representation learning (e.g., *autoencoders*)

Challenges — Overfitting

- Overfitting is one of the most pressing issues in ML
- ML model has been trained on data
  - It fits the training data really well
  - It DOES NOT generalize for unseen data
  - The trained model captures unique properties of the training data…
  - …that only exist for **those data**



Challenges — Overfitting

- Example: classification male/female

Challenges — Overfitting

- Example: classification male/female

Challenges — Overfitting

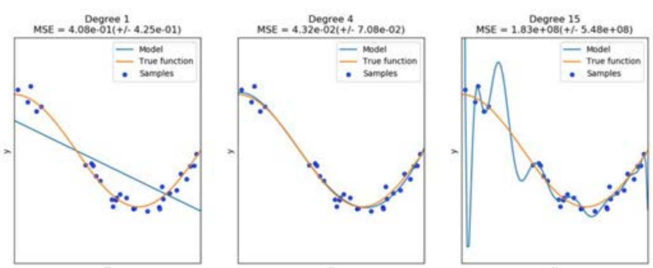
- Overfitting is **hard to estimate**: predict performance on data you don't have?
- Solutions focusing on data
  - Split data in training (validation) and test
  - n-fold cross-validation is a popular choice
- Solutions focusing on the model
  - Expert knowledge on *symptoms* of overfitting (e.g. large values for single weights in NNs)
  - Try to mitigate the symptoms (e.g. regularization, drop-out, …)

- Overfitting remains an **open issue**, no guarantee the model is **not** overfitted

Opportunities

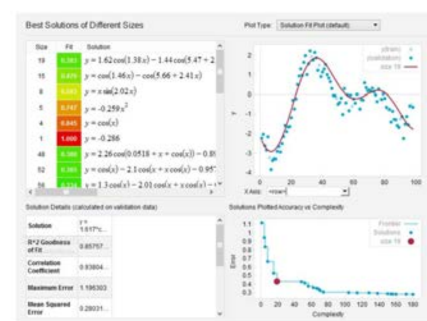
Opportunities — Capacity vs. Fitting

- **Capacity**: # of functions that a ML model can select as a possible solution
- **Fitting**: error with respect to the training data
- Ideally, we want to use the *CORRECT CAPACITY* for the target problem



Opportunities — Capacity vs. Fitting

- Not only, but we want to *minimize capacity* and *maximize fitting*
  - Simpler ML models have a better chance at generalizing (less risk of overfitting)
  - And of course, we'd like to fit the training data as much as possible
- **A multi-objective (MO) problem**!
  - ML community so far has seldom treated it as MO
  - EAs work really well for MO problems (state of the art)
  - EA-based solutions for ML exploit MO optimization



Opportunities — Capacity vs. Fitting

- Why are ML experts not framing the problem as MO?
  - (Zhang et al., 2016) shows a puzzling result
  - Deep networks with WAY larger capacity than necessary, do not overfit as badly as they could
  - In some way, the "correct solution" in the space of NN weights is a stronger attractor than "complete overfit"

Zhang, C., et al. 2016. *Understanding deep learning requires rethinking generalization.* https://arxiv.org/abs/1611.03530

Opportunities — White-box ML

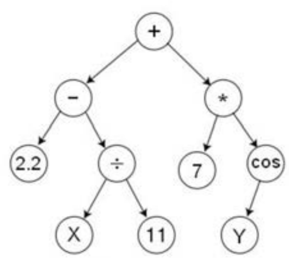
- ML models are often "black boxes"
  - They may deliver good results, but are impervious to human understanding
  - "Explainable AI" techniques can be used to have a better grasp of decision process
  - Adversarial ML was an example, there are more
- White-box machine learning?
  - Return models that can be understood by humans
  - One well-known and explored EA technique can be seen as "white-box ML"
  - Symbolic regression, used to obtain free-form equations

http://www.xkcd.com/1838/

Opportunities — White-box ML

Schmidt, M., & Lipson, H. 2009. *Distilling free-form natural laws from experimental data.* Science, 324(5923), 81-85.

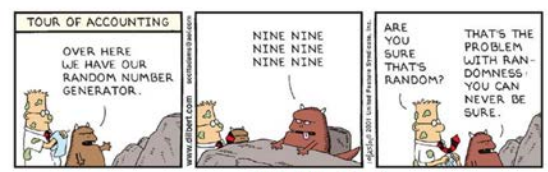
- Symbolic regression
  - Genome: a binary tree, representing an equation
  - Fitness: minimize error wrt training data; also "complexity" (number of terms)
  - Success story for EAs: published in *Science*, commercial product *Eureqa* from start-up *Nutonian*
- Pros and cons
  - Models are human-readable (up to a certain size)
  - Multiple choices of models (less complex, more accurate)
  - Probably less capacity than NNs
  - Modern developments (Geometric Semantic Genetic Programming) have higher capacity, but more black-box



$$\left(2.2 - \left(\frac{X}{11}\right)\right) + \left(7 * \cos(Y)\right)$$

Opportunities — Stochastic Optimization in ML

- Optimization over models in ML algorithms
  - Deterministic approaches: Decision Trees, Support Vector Machines, ...
  - Stochastic approaches: Random Forest, Bagging, Deep Learning, ...
- Interestingly, stochastic algorithms rarely use feedback (pure random!)
  - Stochasticity is used to prevent premature convergence
  - Or, in case of ensembles, to create weak predictors "specialized" in different parts of the data
- **Why don't they use EAs?**



https://dilbert.com/

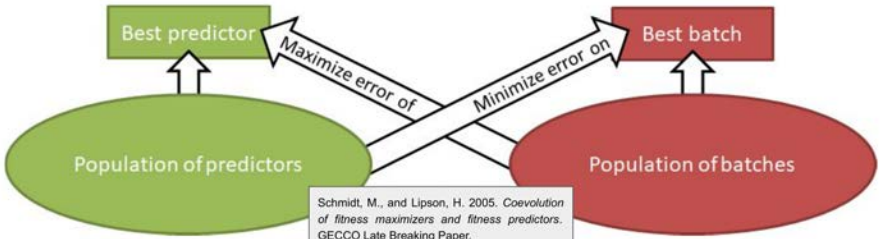
Opportunities — Stochastic Optimization in ML

- Deep Learning (DL) employs Stochastic Gradient Descent (SGD)
  - Used to optimize the weights of the NN, using backpropagation
  - Smaller steps than classical gradient descent
  - Takes into account only a small subset of the training data (*batch*) at each step
  - Helps avoiding premature convergence, local optima appear and disappear
- Why don't they use EA-based methodologies?
  - Some evidence from Chapter 5 of "The Deep Learning Book", by the gurus of DL
  - Empirical explorations of the search space of weights of NNs
  - Reveals LOTS of saddles, very few local optima
  - And SGD is great at escaping local optima
- Basically, they **do not need EAs** in this case

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press. https://www.deeplearningbook.org/

Opportunities — Stochastic Optimization in ML

- Interestingly, randomizing training samples is a recurrent idea
- In the domain of EA, it has been used for Symbolic Regression
- As we are EA practitioners, however, it became **co-evolutionary**



Schmidt, M., and Lipson, H. 2005. *Coevolution of fitness maximizers and fitness predictors.* GECCO Late Breaking Paper.

Opportunities — Stochastic Optimization in ML

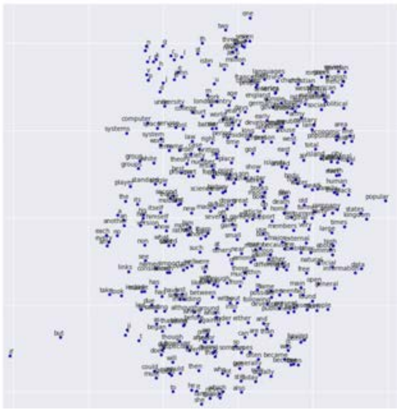
- Another perspective on batches: Lexicase selection
  - From the domain of EAs, again applied mostly to Symbolic Regression
  - When comparing individuals in the same generation, for reproduction or survival
  - Randomly shuffle the samples, and compare individuals sample by sample, in order
  - When the performance of two individuals differs on one sample, stop and select best
  - Improves diversity in the population, allowing "specialists" to survive

Helmuth, T., Spector, L., & Matheson, J. 2014. *Solving uncompromising problems with lexicase selection.* IEEE TEC, 19(5), 630-643.

Opportunities — Exploring Embeddings

- Embeddings are currently a hot research topic
- Project input in a (meaningful) vectorial space
  - Displacements and distances in this space have a **meaning**
  - Mostly (but not only) used in Deep NNs
  - *Building* the vectorial space is the hard part
- Used mostly in Natural Language Processing (text) and images
- Well-known example is Word2Vec
  - Assign random high-dimensional vector to a specific word
  - Optimize, so that words that appear often nearby in text are close together in the vector space

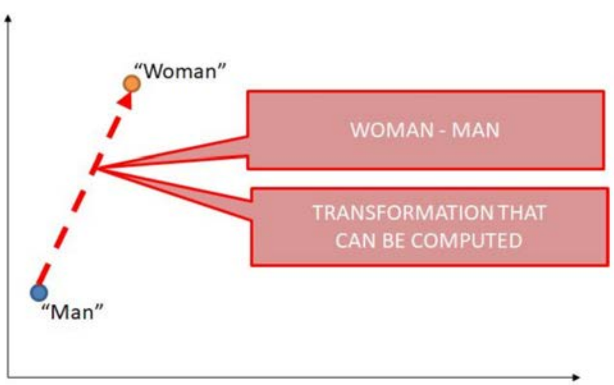
Opportunities — Exploring Embeddings



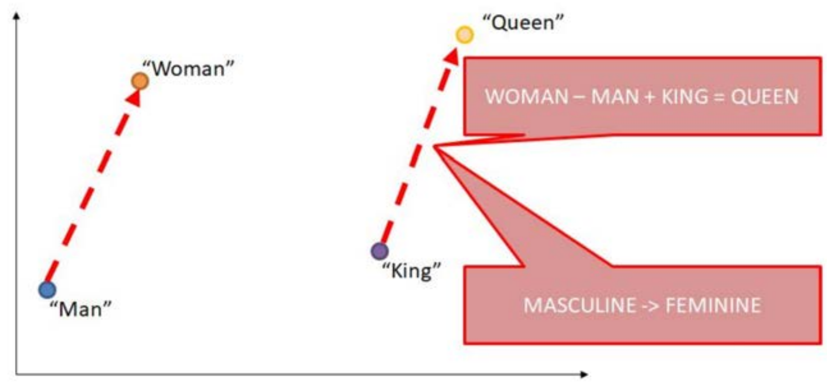
Opportunities — Exploring Embeddings

- “French”, “British”, “American”…
  - Adjectives for nationality!
  - Nearby, you have “languages”, “countries”
- Also, “England”, “Europe”, “International”, …

Opportunities — Exploring Embeddings



Opportunities — Exploring Embeddings



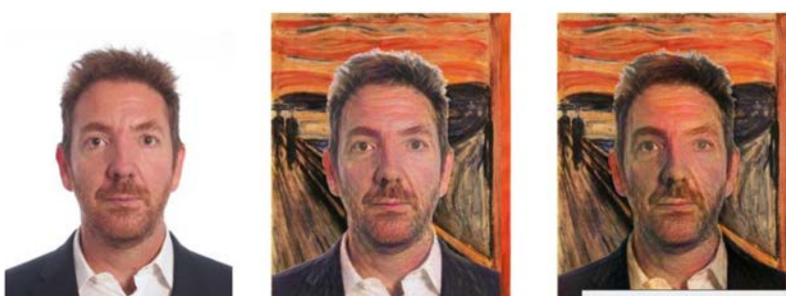
Opportunities — Exploring Embeddings

- Another impressive example is *Deep painterly harmonization*
- Nowadays also known as “style transfer”
  - Train Deep NN to classify different styles of paintings (and photos)
  - Take last two layers as embedding
  - Find position of original photo and target painting inside the embedding
  - Compute vector between the two, and slowly move photo *towards* painting
- The resulting point is then transferred to the pixel space

Opportunities — Exploring Embeddings

Luan, F. et al. 2018. *Deep painterly harmonization*. In Computer Graphics Forum. https://arxiv.org/abs/1804.03189

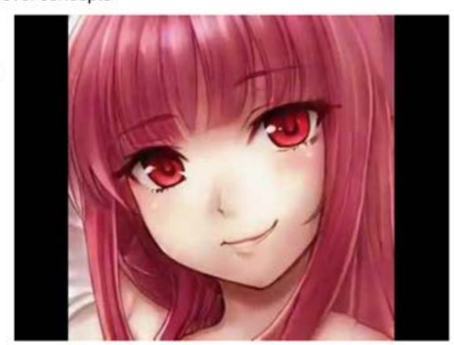
Opportunities — Exploring Embeddings



Luan, F. et al. 2018. *Deep painterly harmonization*. In Computer Graphics Forum. https://arxiv.org/abs/1804.03189

Opportunities — Exploring Embeddings

- *Exploration* of embeddings can provide great insight
  - Embeddings taken from NNs encode high-level concepts
  - For example, “style of painting”, “muscular man”, “evil-looking drawing”, …
- Right now, exploration of embeddings is at the very beginning
- If the appropriate fitness function is discovered, opportunity for EAs



Questions?

Resources

- "The deep learning book", https://www.deeplearningbook.org/
- Scikit-learn, Python module with tons of different ML algorithms, https://scikit-learn.org/stable/
- Keras, Python module with high-level interface to Tensorflow and other deep learning libraries, https://keras.io/