grad center AI class
17 nov 2011
learning to play the tron game

**outline:**

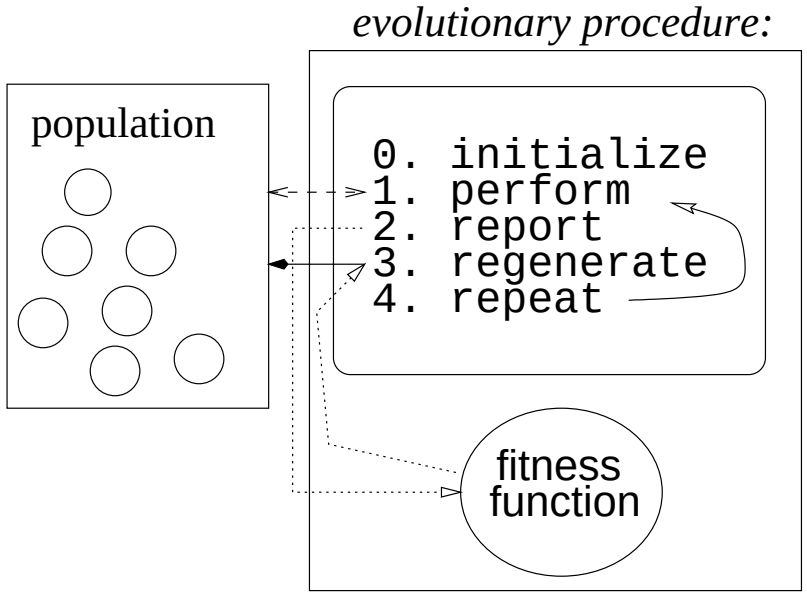
- brief introduction to machine learning
- the tron game
- first experiment: genetic programming
- second experiment: neural networks

*guest lecturer:*
Prof Elizabeth Sklar, CUNY Brookyln College
`http://www.sci.brooklyn.cuny.edu/~sklar`
email: *sklar@sci.brooklyn.cuny.edu*

# one-page overview of machine learning

- learning can be categorized as a type of search
- key components
  - method for representing candidate solution
  - method for adjusting candidate solution
  - method for evaluating candidate solution
- examples of representations
  - lookup table, genetic algorithm, genetic program, neural network
- examples of adjustment methods
  - evolutionary learning, reinforcement learning, statistical learning
- categories of methods for evaluating the candidate
  - supervised learning, unsupervised learning

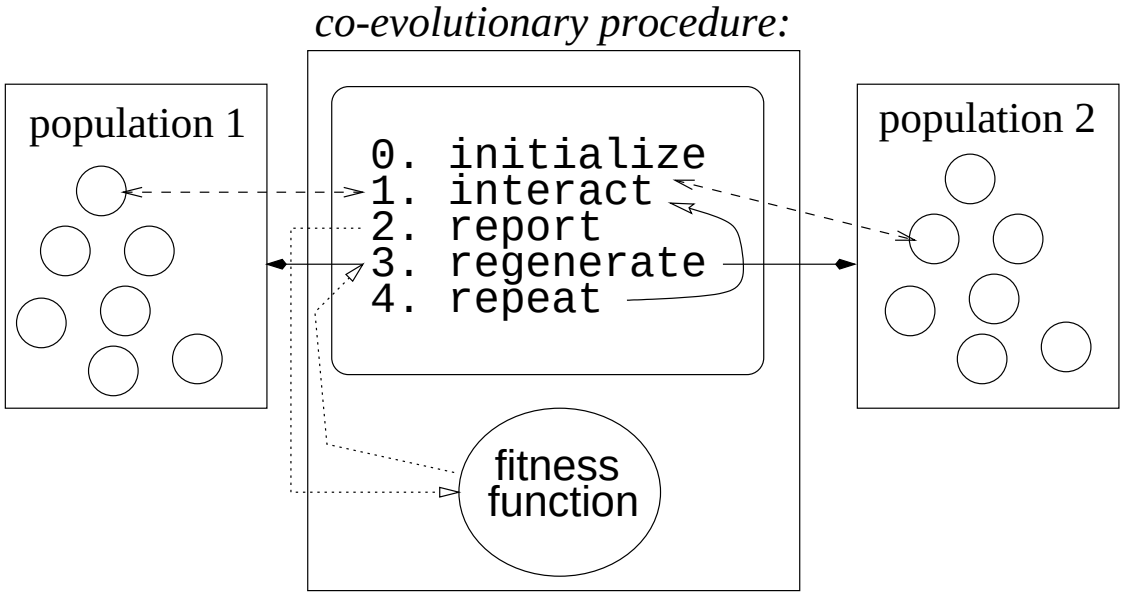
# evolutionary learning



# genetic algorithm

- *genotype:* binary string
  each bit represents the presence or absence of some attribute
- *phenotype:* embodiment of the genotype
- reproduction operators:
  - mutation
  - cross-over

# co-evolutionary learning



# learning to play games

- learning from humans
  - e.g., Checkers [Samuels 1959]
  - too many games are needed
  - humans are noisy
  - humans are learning
- learning from computers
  - e.g., Backgammon [Tesauro 1992]
  - lack of generalization
  - deceptive landscape
  - premature convergance

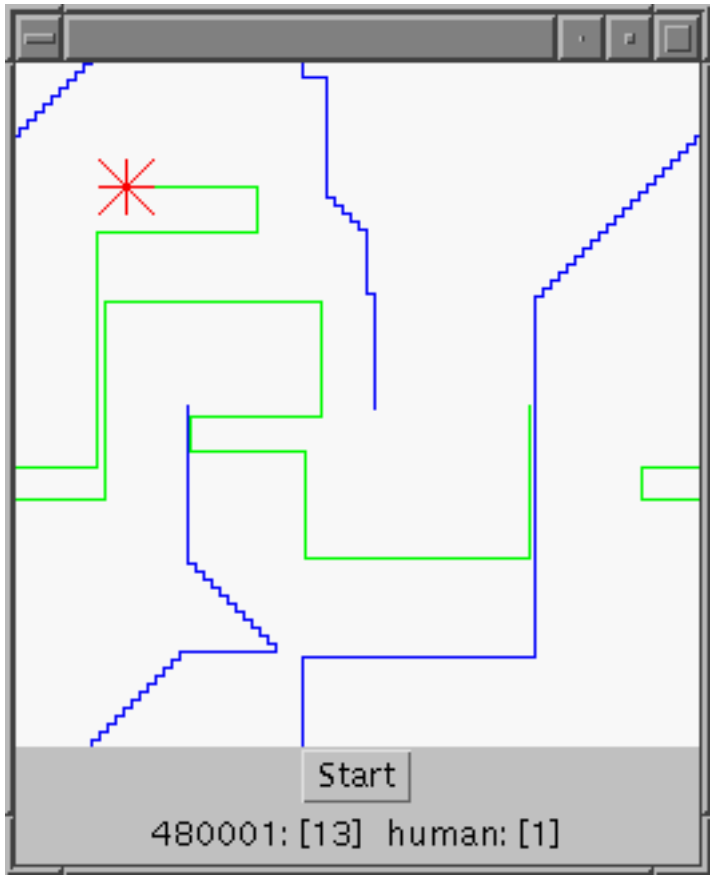
# learning from humanity

- Data-mining the clickstream:
  - Warren Teitelman, 1969, DWIM
  - Allen Cypher, 1991, Eager
  - Pattie Maes, 1994, Collaborative Interface Agents
- Our work:
  - mine the **clickstream** to train agents using supervised learning
  - take a *population-based* approach
  - find the "perfect partner": a population of graded agents that are deployed as needed

# the Tron game

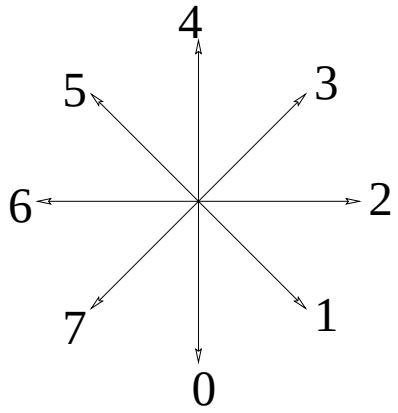
- First experiment: in September 1997, we built a Java version of the real-time video game called Tron and released it on the Internet.
  - take advantage of the Internet...
  - users *collectively* supply a *fitness function*
  - agents *collectively* embody an *intelligent opponent*
- Human visitors to our web site play games against an evolving population of intelligent agents.
- Second eperiment: follow-on work done in 1999–2000 that examines the idea of data mining the Internet clickstream and using this data to train software agents to emulate the behavior of humans.
- Our aim was to use this technique as the basis for constructing a population of "graded" agents that can be used as a suite of opponents in future games.
- This work was conducted jointly with Dr Alan Blair (UNSW, Sydney, Australia) and Dr Pablo Funes (Icosystem, Cambridge, MA).

# task domain: Tron



# agent architecture

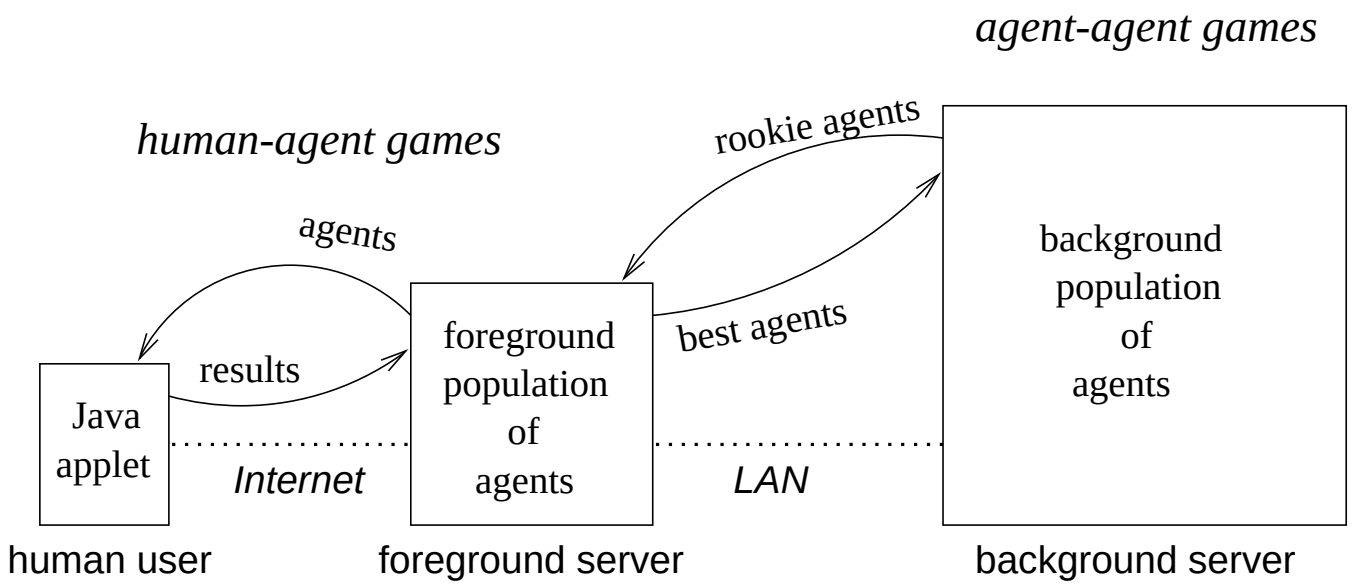
- sensors:




- actions:
  - *left*
  - *right*
  - *straight*
- controller:
  - genetic program (GP) (first experiment)
  - neural network (second experiment)

the first experiment

# system architecture



# genetic programming (GP)

- John Koza (1992)
- like Genetic algorithms (GA)... BUT: genotype $\rightarrow$ Lisp s-expression
- Tron GP parameters
  - $+, -, *, \%, IFLTE$
  - LEFT, RIGHT
  - sensors $\rightarrow$ s0,s1,...,s7
  - $\Re$
  - maximum depth $= 17$
  - maximum length $= 512$ symbols

# exploration vs exploitation

- foreground population:
  - 100 members:
  - 90 "veterans" $\times$ 5 games $\Rightarrow$ 82% *exploration*
  - 10 "newbies" $\times$ 10 games $\Rightarrow$ 18% *exploitation*
- background population:
  - 1000 members
  - 25 member training set:
    - 15 best from foreground
    - 10 best from background
  - 1000 $\times$ 25 games
  - best half mate and generate a new bottom half

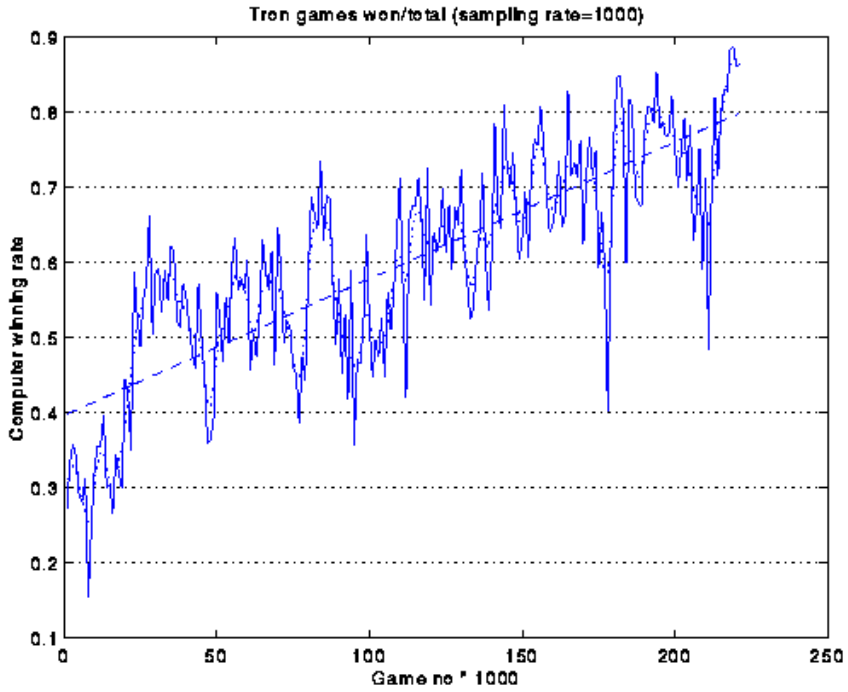
# analysis of GP 460003

*(IFLTE 0.98413 s0 (IFLTE s2 s6 (- s3*
*(IFLTE s2 (IFLTE (+ s0 s1) s5 s4 (RIGHT))*
*(% s7 s4) (LEFT))) (LEFT)) (% s0 s6))*

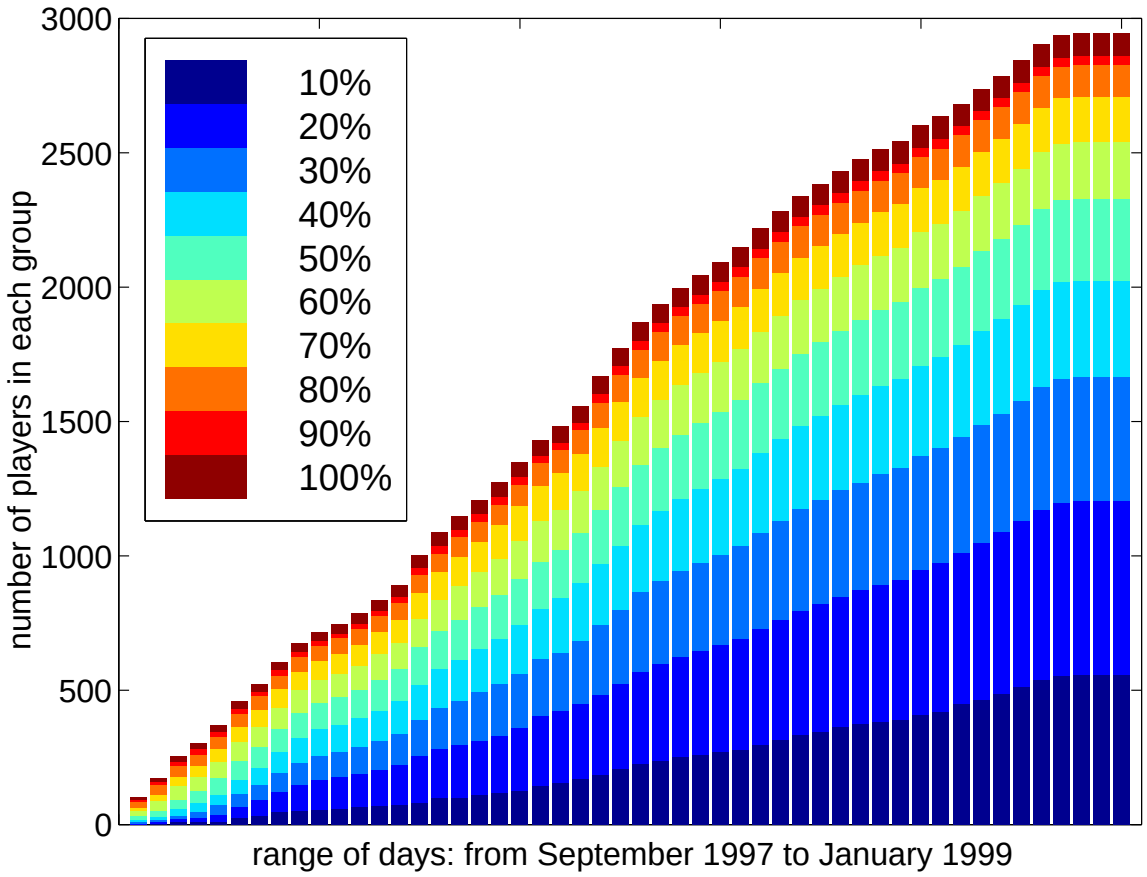
```
if ( s0 < 0.98413 ) {
  go straight
}
else if ( s6 >= s2 ) {
  turn left
}
else if (( s0 + s1 ) >= s5 ) {
  turn right
}
else
  turn left
}
```

# results

- In the Internet experiment, we collected data on over 200,000 games played by over 4000 humans and 3000 agents.
- Our general performance measure is the **win rate**, calculated as the number of games won divided by the number of games played.



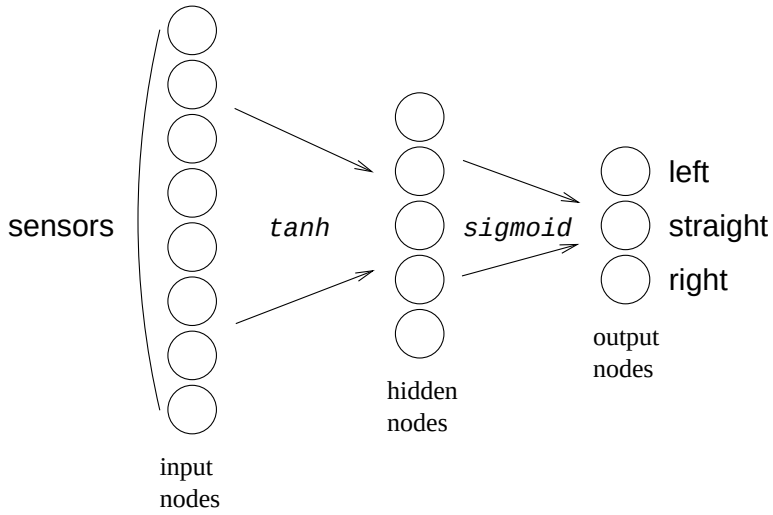
# distribution of abilities in human population



the second experiment

# modeling human behavior

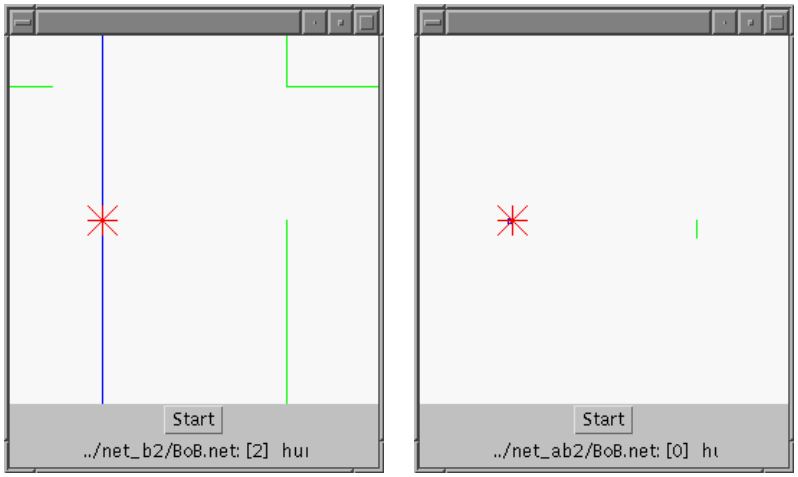
- In addition to saving win rates of players, we also save the moves of every game, in a compressed form called the **moves string**.
- Our idea was to "train" neural networks to emulate human players, using the **moves strings** as training data.
- neural network controller architecture:



# training process

- use supervised learning
- designate:
  - *trainer*
  - *opponent*
  - *trainee*
- select series of **moves strings**
- replay game between trainer and opponent ("VCR")
- let trainee predict trainer's move at each time step
- adjust *weights* of trainee's network accordingly

# challanges

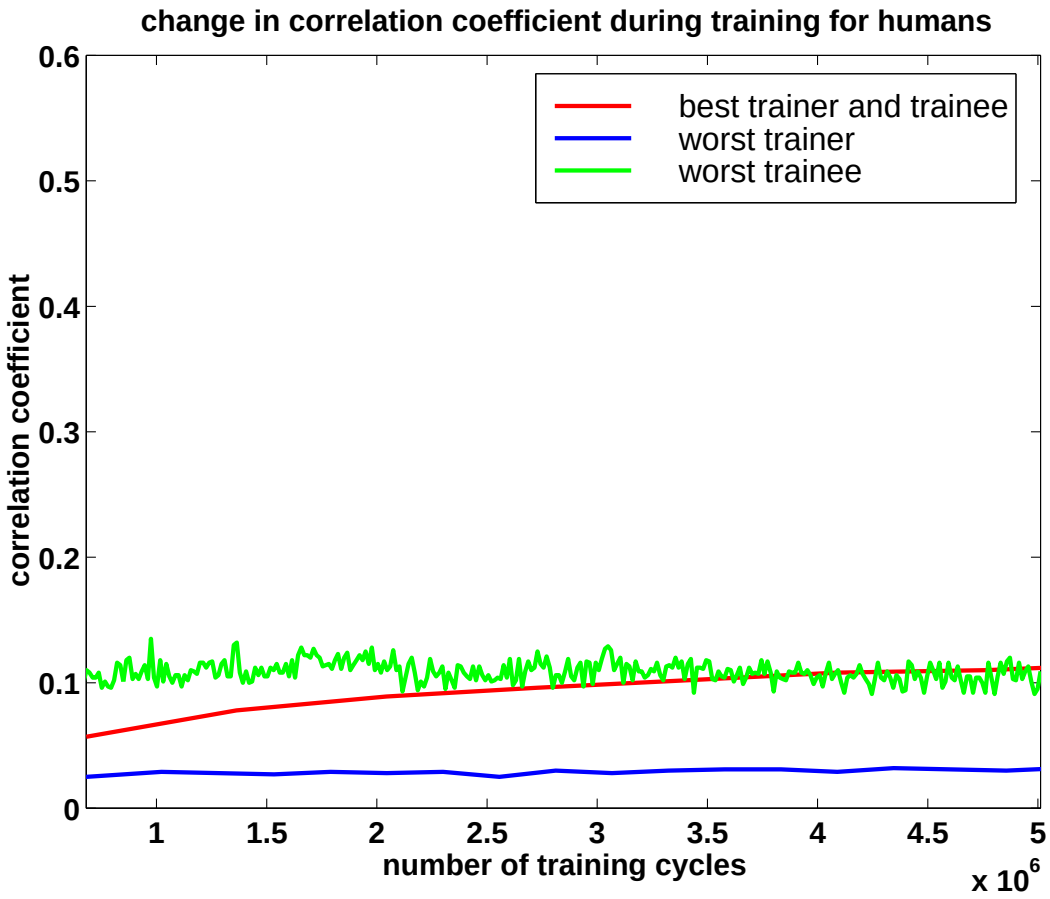



- frequency of moves table:

| | | **trainee** | | |
|---|---|---|---|---|
| | | $left$ | $straight$ | $right$ |
| **trainer** | $left$ | 852 | 5360 | 161 |
| | $straight$ | 5723 | 658290 | 5150 |
| | $right$ | 123 | 4668 | 868 |

# results: improvement during training

- Look at **correlation** between "trainer" and "trainee".

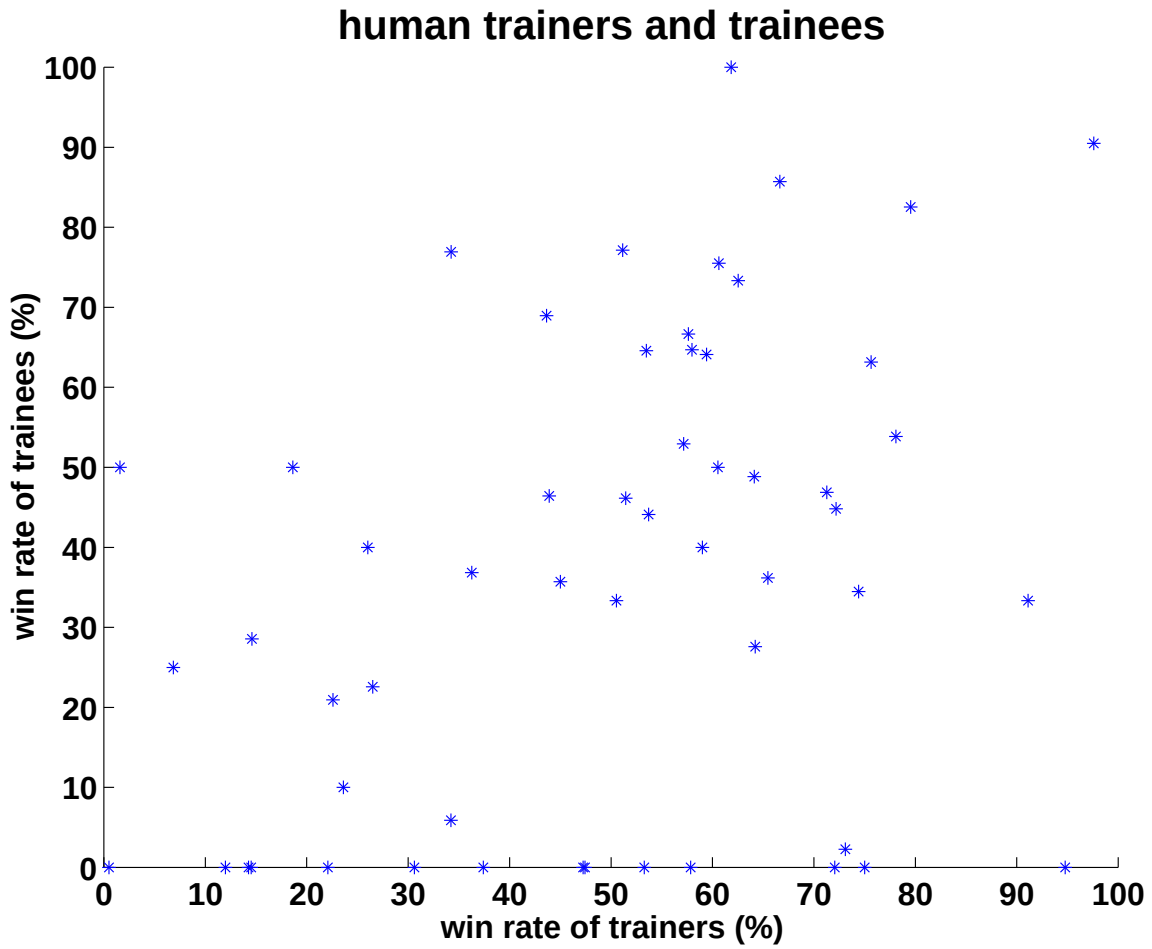


- If the trainee were a perfect clone of its trainer, then the *correlation coefficient* in the frequency of moves table would be $1$.
- But this is difficult to achieve in practice because the humans are non-deterministic and may make different moves when faced with the same, or similar situations.
- Also, the exact timing of a turn varies from one move to another.
- In reality, the correlation peaks at around $0.14$. For comparison, we computed correlation coefficients for $127$ random players (note: players that choose a move randomly at each time step), and get a much smaller correlation of $0.003$.

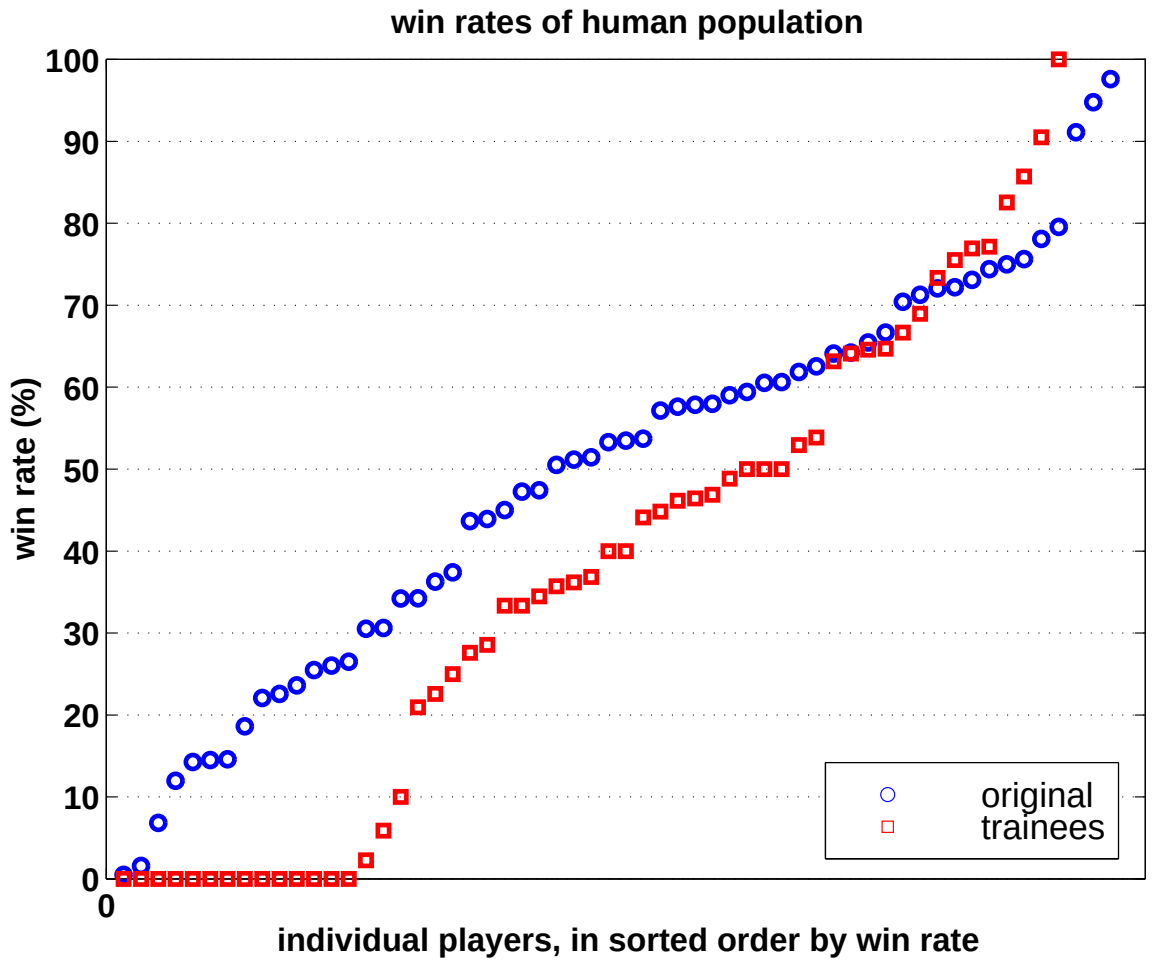
## results: individual trainers

- We trained networks based on input from the 58 humans who played more than 500 games in the Internet experiment.
- The left graph compares the win rate of individual trainees to that of their corresponding trainers. Notice that better human players generally produce better trainees.
- A few of the trainees play very poorly. These are cases where the network either fails to make any turns or makes turns at every move, in spite of our strategy of using the frequency of moves table. Also, note that in a number of cases, the trainee outperforms its trainer.
- The right graph emphasizes our population-based approach. Here, players are sorted within each population according to their win rate, so there is no direct correspondance between individual trainers and trainees, as in the upper graph.
- A variety of abilities has been produced.

- trainer-trainee comparison:



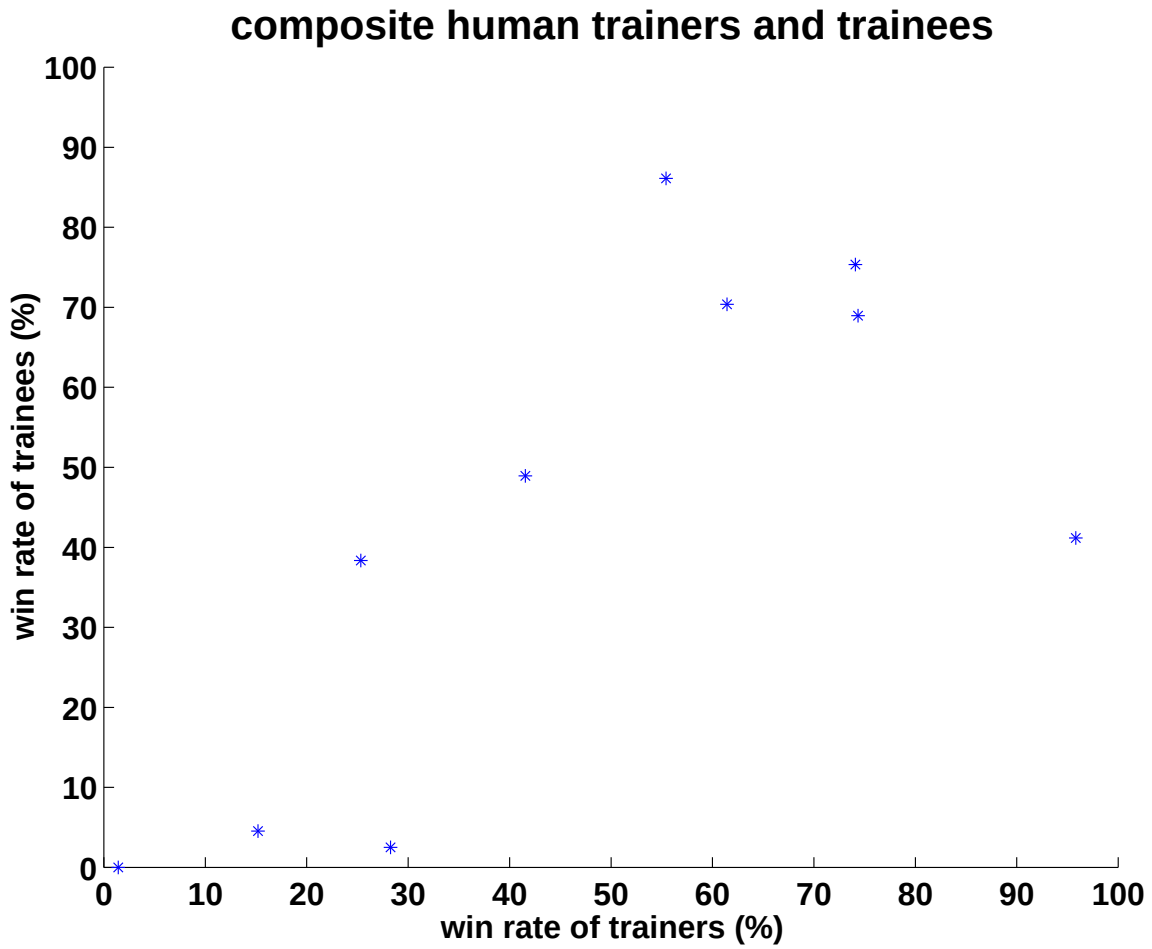
- population comparison:



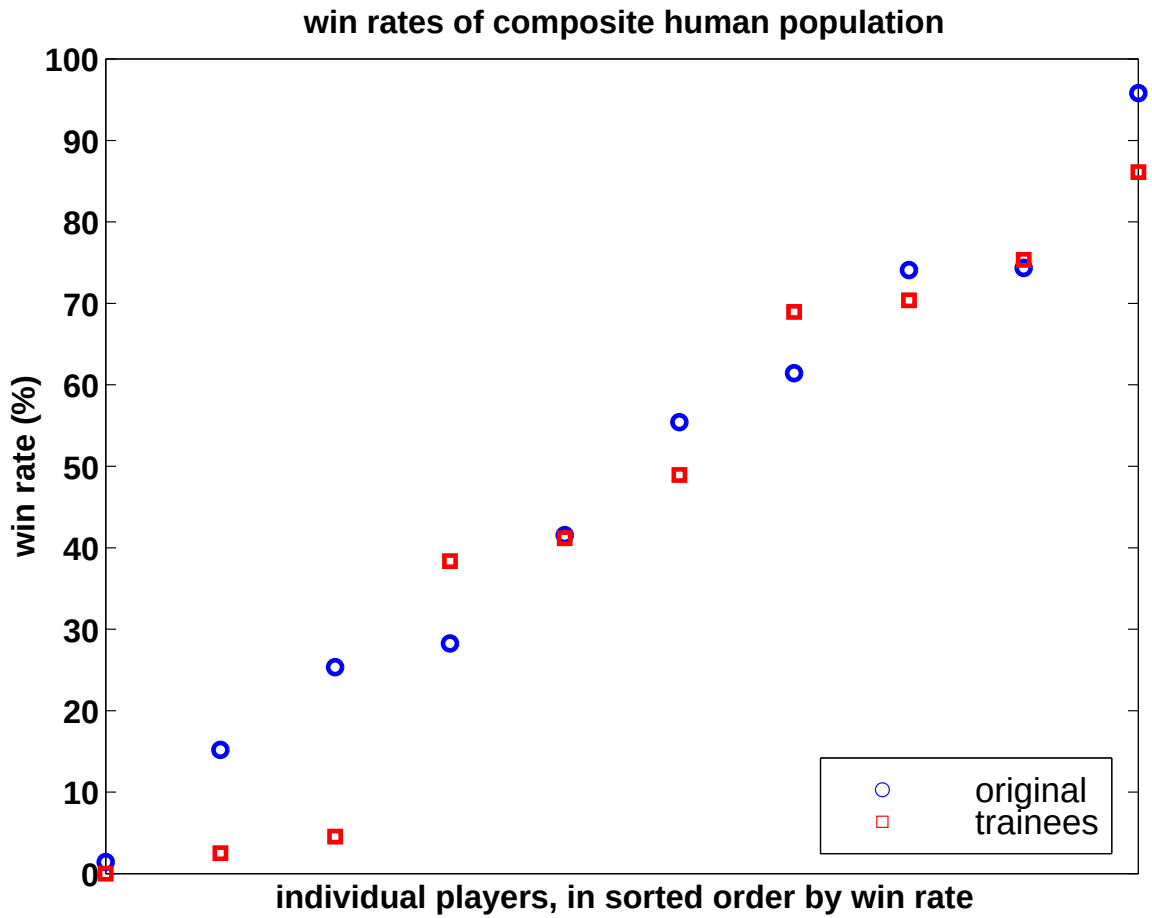
## results: clustered trainers

- The next set of plots shows the results obtained when we clustered human trainers according to their win rates.
- We trained one agent to emulate the behavior of each group of humans, with win rates of 10%, 20%, and so on.
- The first plot shows the correspondance between individual trainers and trainees.
- The second plot shows the distribution over the population of trainers and trainees.

- trainer-trainee comparison:



- population comparison:



# conclusions

- Neural networks can learn to play Tron by supervised learning, training on human data.
- The population of trained agents provides a distribution of abilities similar to that of the original human population.
- More promising results can be obtained if we train on groups of humans with similar features, rather than individual humans.

# references and further reading

- my papers on Tron
  - first experiment:
    Pablo Funes, Elizabeth Sklar, Hugues Juillé and Jordan B. Pollack (1998). *Animal-Animat Coevolution: Using the Animal Population as Fitness Function.* In Pfeifer, R. et. al. (eds), From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior (SAB-98), MIT Press, pp. 525-533.
  - second experiment:
    Elizabeth Sklar, Alan D. Blair, Pablo Funes and Jordan Pollack (1999). *Training Intelligent Agents Using Human Internet Data.* In Proceedings of the First Asia-Pacific Conference on Intelligent Agent Technology (IAT-99), pp 354-363.
- genetic algorithms
  - Melanie Mitchell and Stephanie Forrest (1995). *Genetic Algorithms and Artificial Life*. Artificial Life: An Overview. edited by Christopher G. Langton. publisher: MIT Press, pp268-289.
- genetic programming

- John R. Koza (1991). *Evolution and co-evolution of computer programs to control independently-acting agents.* From Animals to Animats. Proceedings of the First International Conference on Simulation of Adaptive Behavior. MIT Press.

- evolutionary neural networks
  - D. B. Fogel, L. J. Fogel and V. W. Porto (1990). *Evolving Neural Networks.* Biological Cybernetics, volume 63, pp487-493.
- co-evolution
  - Jordan B. Pollack and Alan D. Blair (1998). *Co-Evolution in the Successful Learning of Backgammon Strategy.* Machine Learning, volume 32, number 3, pp225-240.