cisc3665
game design
fall 2011
lecture # IV.2
data collection and analysis

topics:

- data collection
- data analysis
- research in my group

data collection

- player information
  - login: username, password
  - demographics — typically used for research
    gender, age, ethnicity, other possible attributes
- playing habits
  - when do they play?
  - how often do they play?
  - how long/how many games do they play at once (in one login session)?
- player game play
  - how do they play?
  - do they win/lose?
  - what are the characteristics of their play?

data analysis

- analyze the data collected!
- for academic research or for commercial marketing
- standard statistics:
  - mean, standard deviation
  - mode (most popular)
  - median (middle)
- more interesting data visualizations to show behaviors
  - trajectories
  - clusters
- behavior modeling / opponent modeling

research in my group

- learning behavior models from game play data

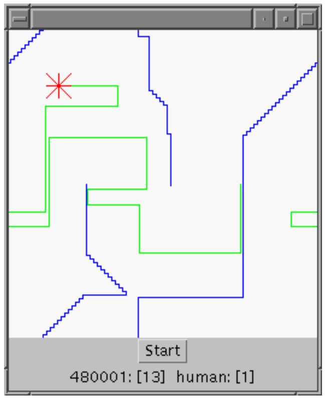
Tron

- In September 1997, we built a Java version of the real-time video game called Tron and released it on the Internet.
- Human visitors to our web site play games against an evolving population of intelligent agents.
- The results of this experiment were presented at SAB'98, the Fifth International Conference on Simulation of Adaptive Behavior, which was in Zurich in August 1998.
- The work described here is 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).

motivation

- 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 for use in on-line learning environments
  - take a *population-based* approach
  - find the "perfect partner": a population of graded agents that are deployed as needed

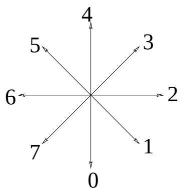
task domain: Tron



480001: [13] human: [1]

agent architecture

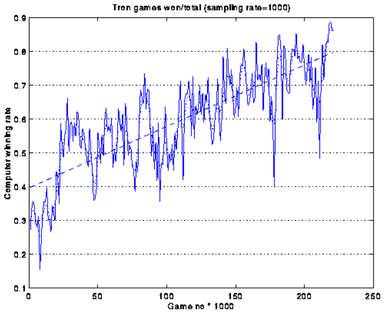
- sensors:



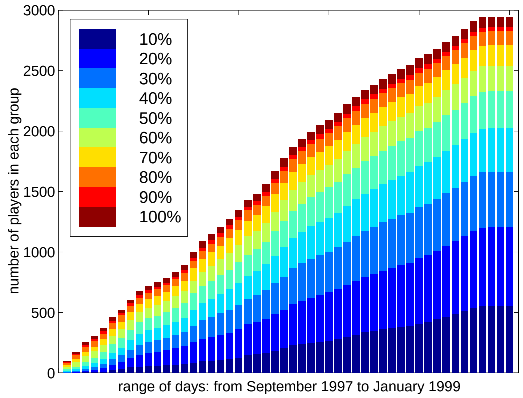
- actions:
  - *left*
  - *right*
  - *straight*
- controller:
  - genetic program (GP)
  - neural network

results of Tron internet experiment

- 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.

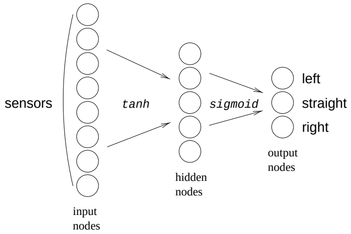


distribution of abilities in human population



modeling human behavior

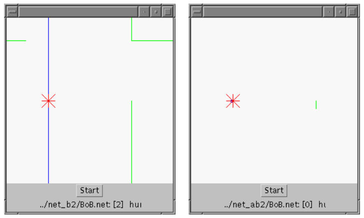
- In addition to saving win rates of players, we also save the moves of every game, in a compressed form called the **moves string**.
- Our idea was to "train" neural networks to emulate human players, using the **moves strings** as training data.
- neural network controller architecture:



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



../net_b2/BoB.net: [2] hu

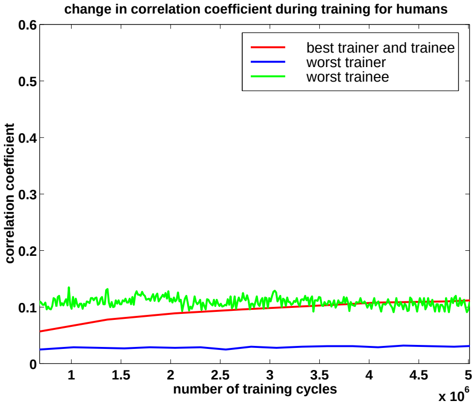
../net_ab2/BoB.net: [0] h

- 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”.

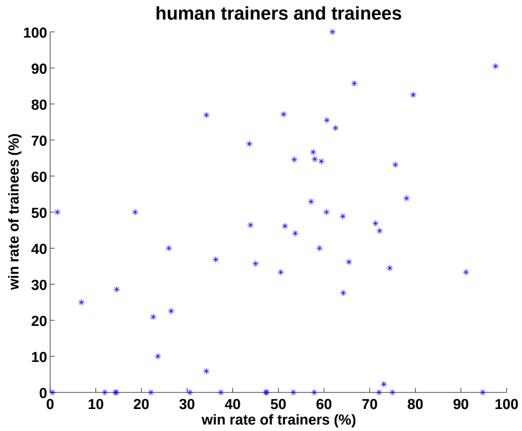


- If the trainee were a perfect clone of its trainer, then the *correlation coefficient* in the frequency of moves table would be 1.
- But this is difficult to achieve in practice because the humans are non-deterministic and may make different moves when faced with the same, or similar situations.
- 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.

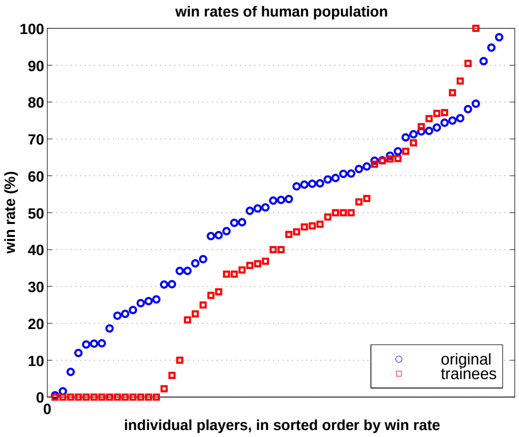
results: individual trainers

- 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:



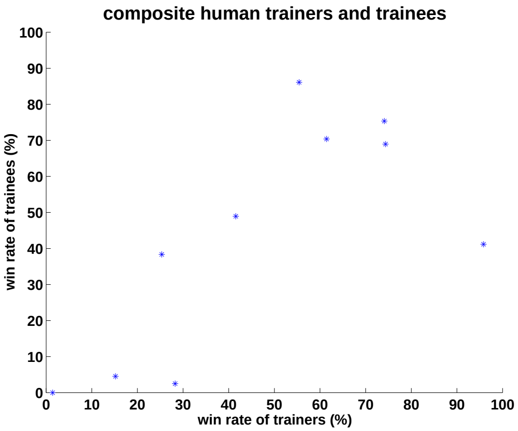
- population comparison:



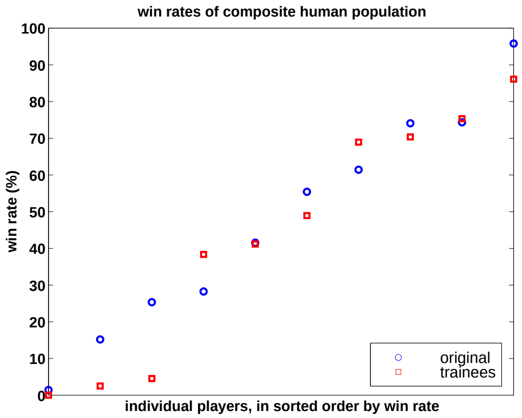
results: clustered trainers

- 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.

current work

- train a Tron player using *reinforcement learning*
- train players for other games, including:
  - Tetris
  - Poker
  - Gomoku, Reversi, MagicSet

to do

- work on homework assignment for unit IV, which is due November 17