Deep reinforcement learning

Michal CHOVANEC, PhD.

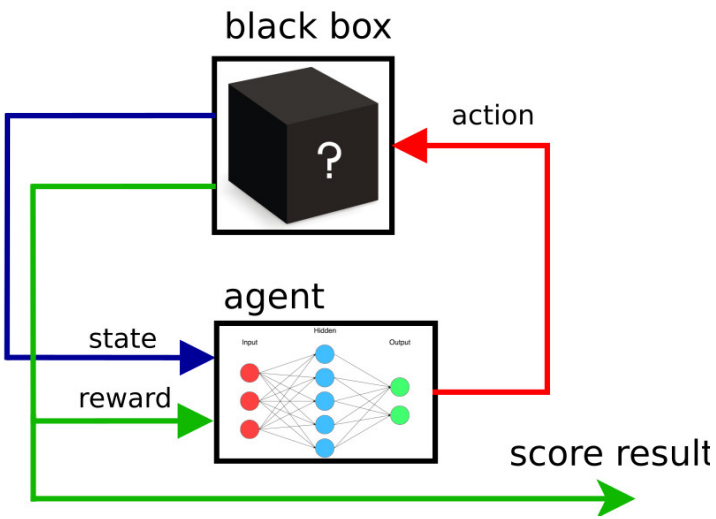
March 2018

Faculty of Management Science and Informatics

Problem definition

- learn to play game with unknow rules
- input : state and reward
- output : action and total score
- $Q(s, a)$: learn Q function

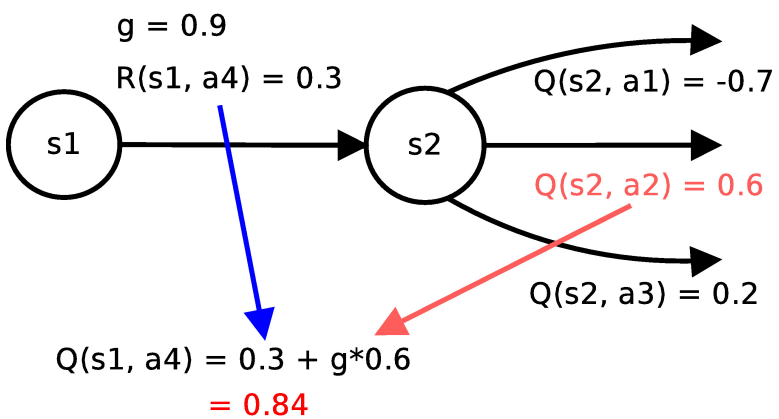
agent never sees required value (required action)



Q-learning algorithm

$$Q'(s,a) = R(s,a) + \gamma \max_{a' \in A} Q(s',a')$$

where
$Q(s,a)$ is previous state
$Q(s',a')$ is actual state
$R(s,a)$ is reward obtained in state s after executing action a
γ is discount factor $\gamma \in \langle 0,1 \rangle$



SARSA algorithm

State Action Reward State Action

$$Q'(s,a) = (1-\alpha)Q(s,a) + \alpha(R(s,a) + \gamma Q(s',a'))$$

where

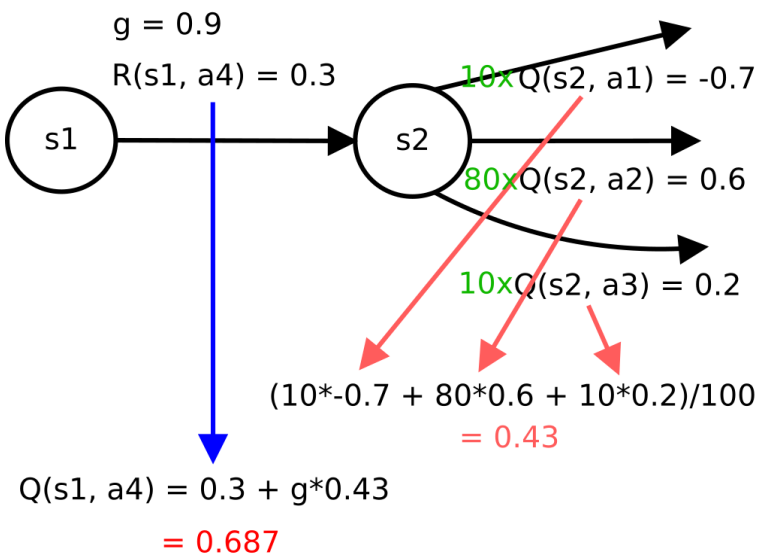
$Q(s,a)$ is previous state

$Q(s',a')$ is actual state

$R(s,a)$ is reward obtained in state s after executing action a

γ is discount factor $\gamma \in \langle 0,1 \rangle$

α is learning rate $\alpha \in (0,1)$



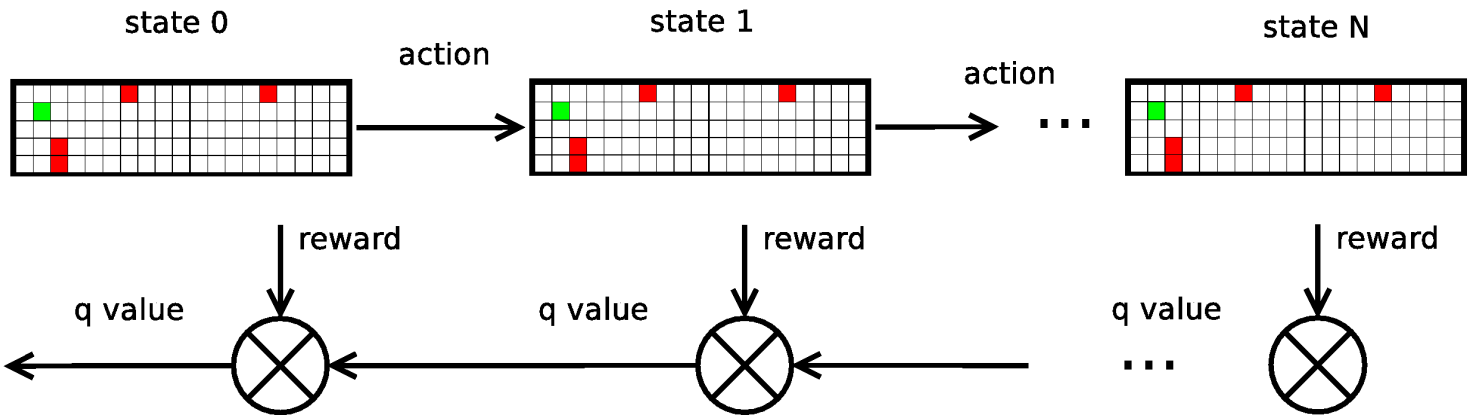
Storing Q values

- table
- linear combination of basis function (handmade features)
- Kenerva's sparse encoding
- neural network

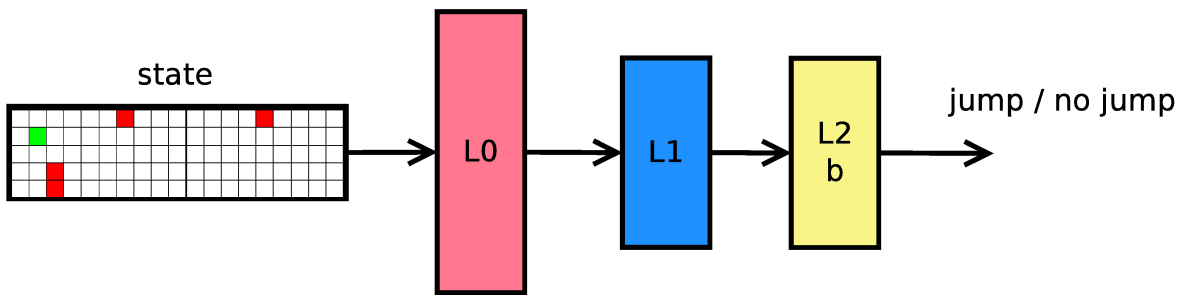
problems

- state correlations
- nonstationary Q values
- convergence to optimal strategy

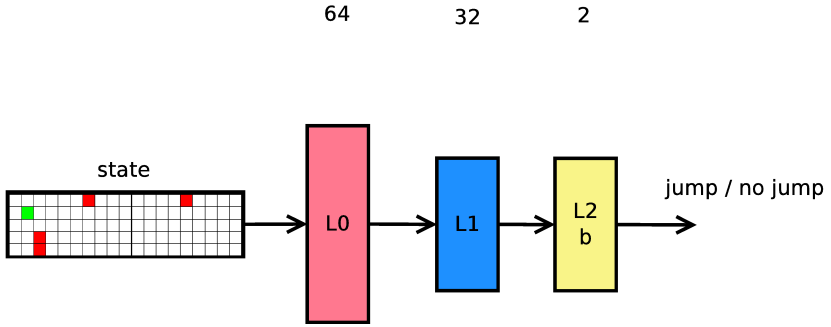
Neural network approximator - deep reinforcement learning



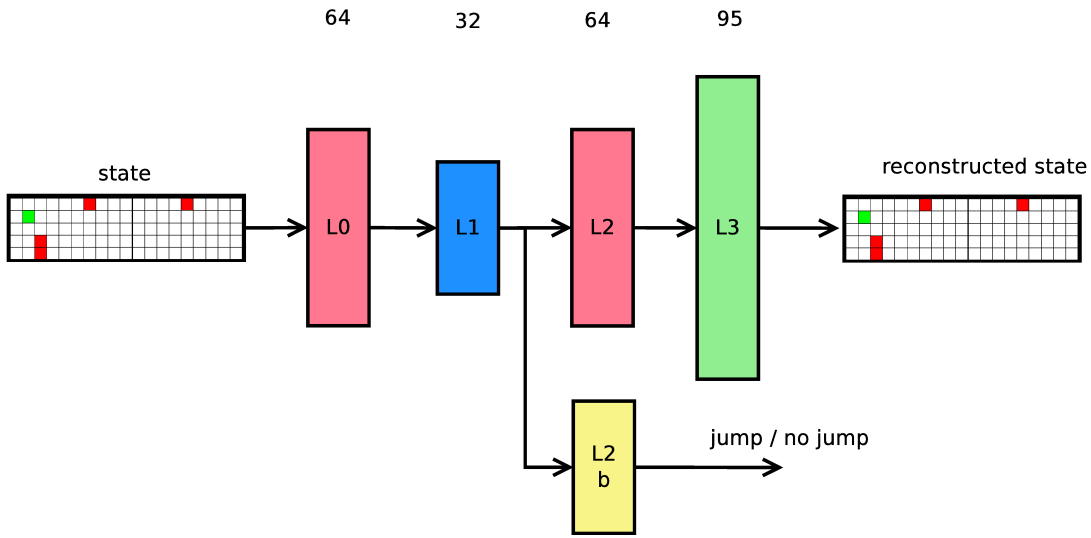
64 32 2



Speed up learning



common feed forward neural network



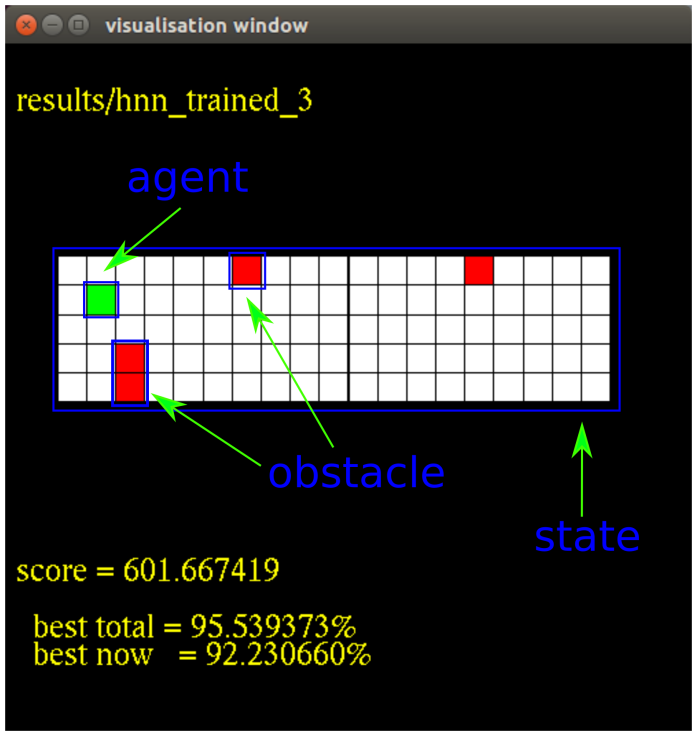
stacked autoencoder + feed forward neural network

Sparse weights - less computing

$$\Delta w = \eta E x \frac{df(y)}{dw} - \lambda sgn(w)$$

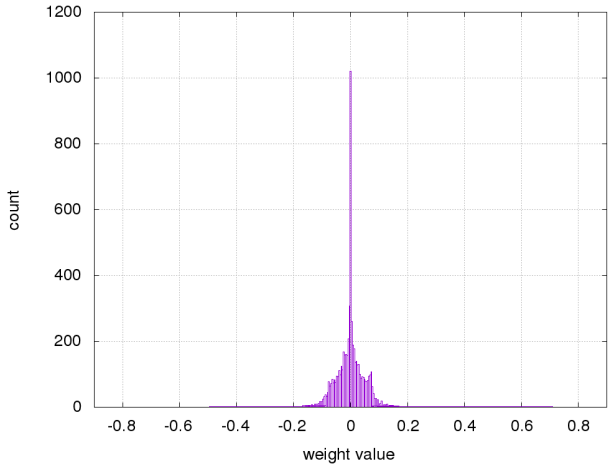
where
E is error,
x is input,
y is output,
f is activation function (ReLU, tanh, softmax ...),
η is learning rate ,
λ is sparsity parameter

Arcade game experiment

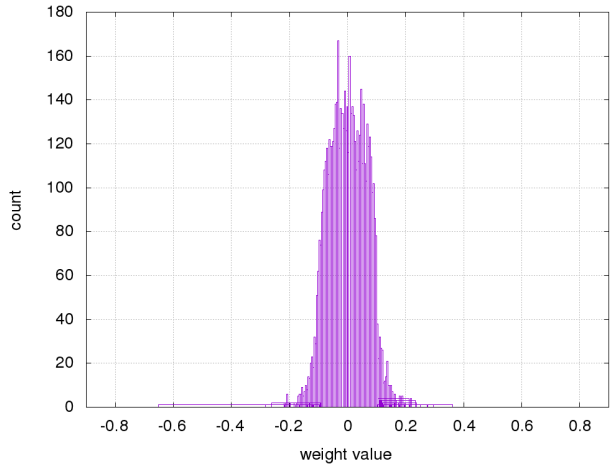


	FNN sparse	FNN no sparse	AE+FNN sparse	AE+FNN no sparse
unsupervised iterations	0	0	100000	100000
supervised iterations	200000	200000	200000	200000
iterations per slice	0	0	50000	50000
learning rate	0.0005	0.0005	0.0005	0.0005
init weight range	0.1	0.1	0.1	0.1
dropout	0	0	0	0
lambda	0.00000001	0	0.00000001	0

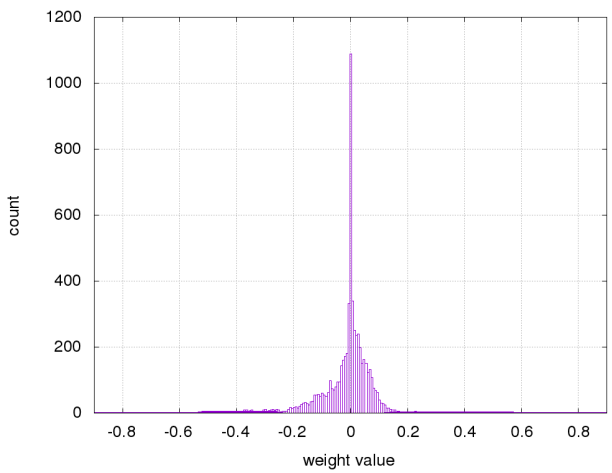
Sparsity results



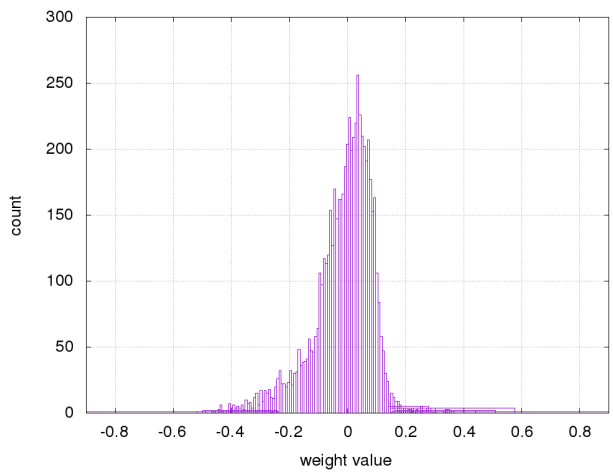
FNN sparse weights histogram



FNN no sparse weights histogram

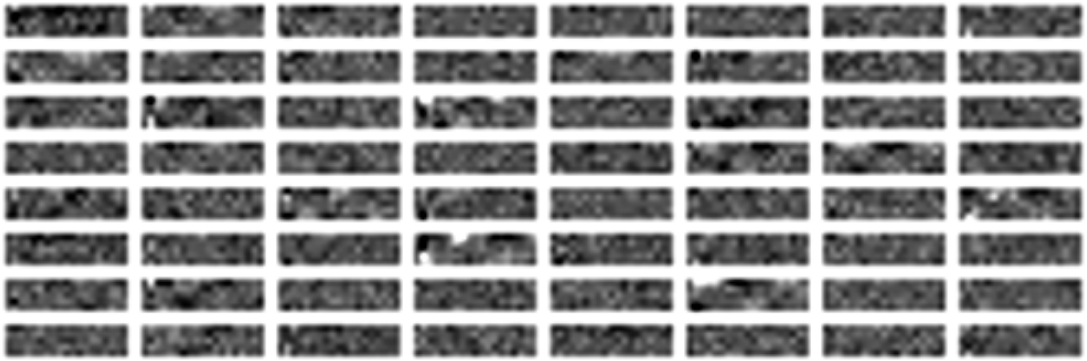


AE+FNN sparse weights histogram

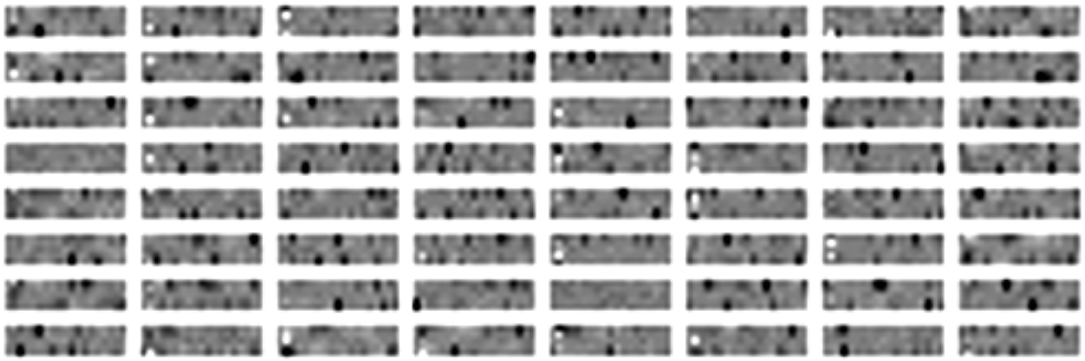


AE+FNN no sparse weights histogram

Sparsity results

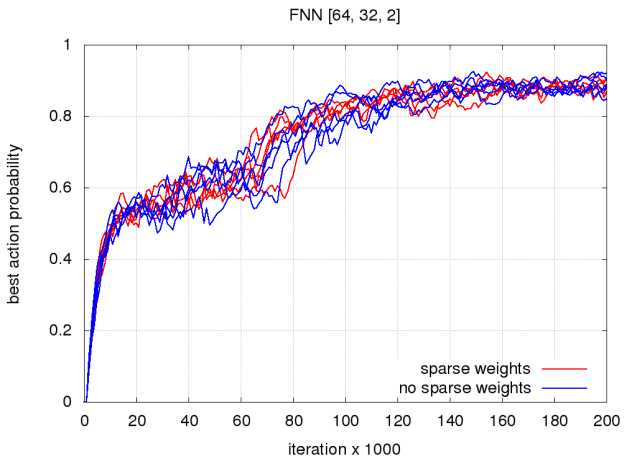


FNN sparse weights visualisation

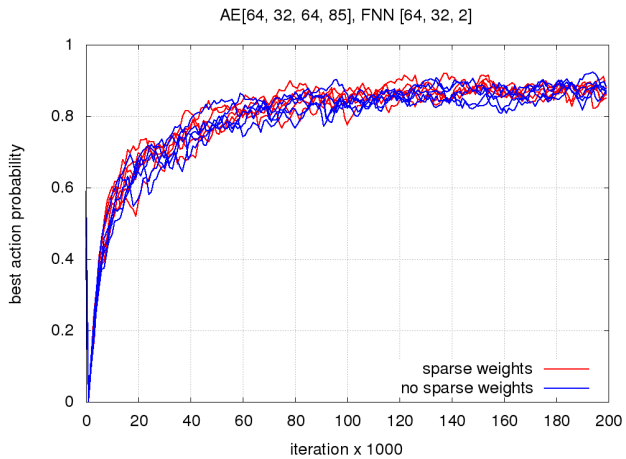


AE+FNN sparse weights visualisation

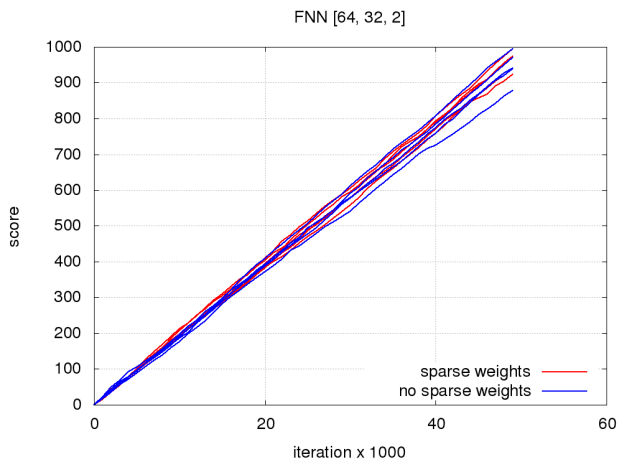
Score results



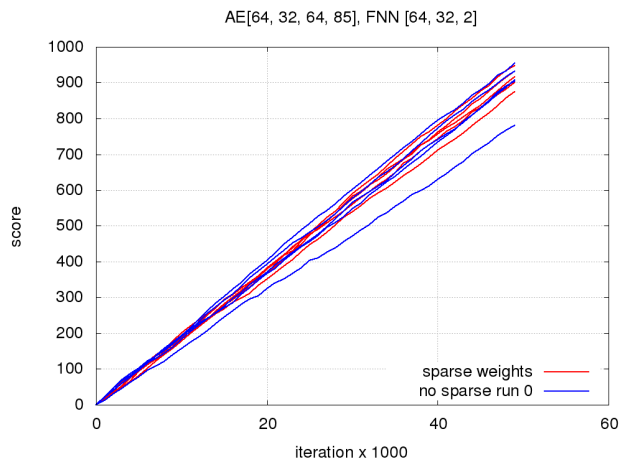
FNN progress comparison



AE+FNN progress comparison

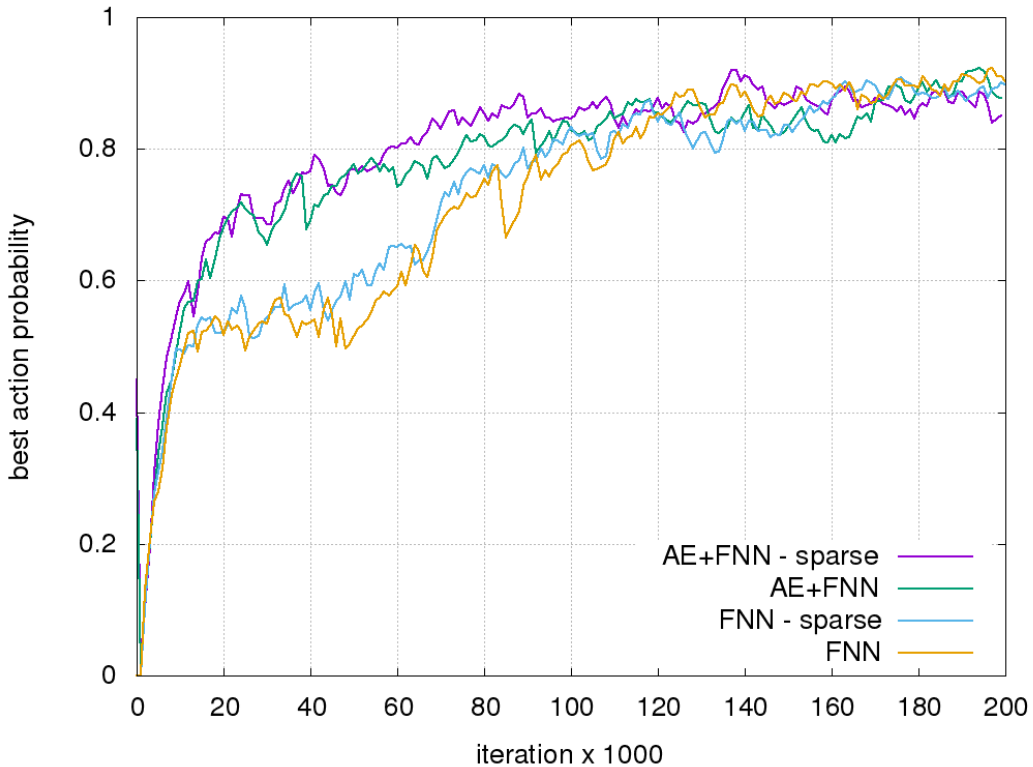


FNN score



AE+FNN score

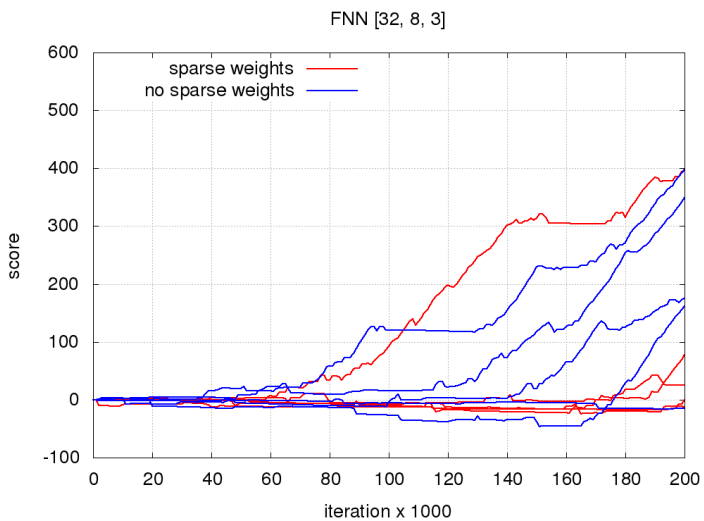
Results

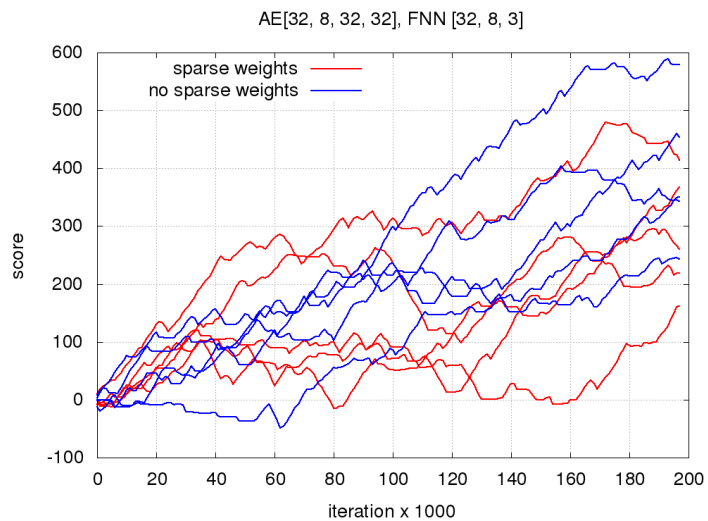


Training progress comparison

	average score	best score	worst score	average best action probability [%
FNN sparse weights	960.58	994.97	922.64	95.32
FNN nosparse weights	945.04	995.64	878.31	93.29
AE+FNN sparse weights	914.5	947.64	875.31	93.4
AE+FNN no sparse weights	908.58	954.31	780.32	93.12

Snake game experiment

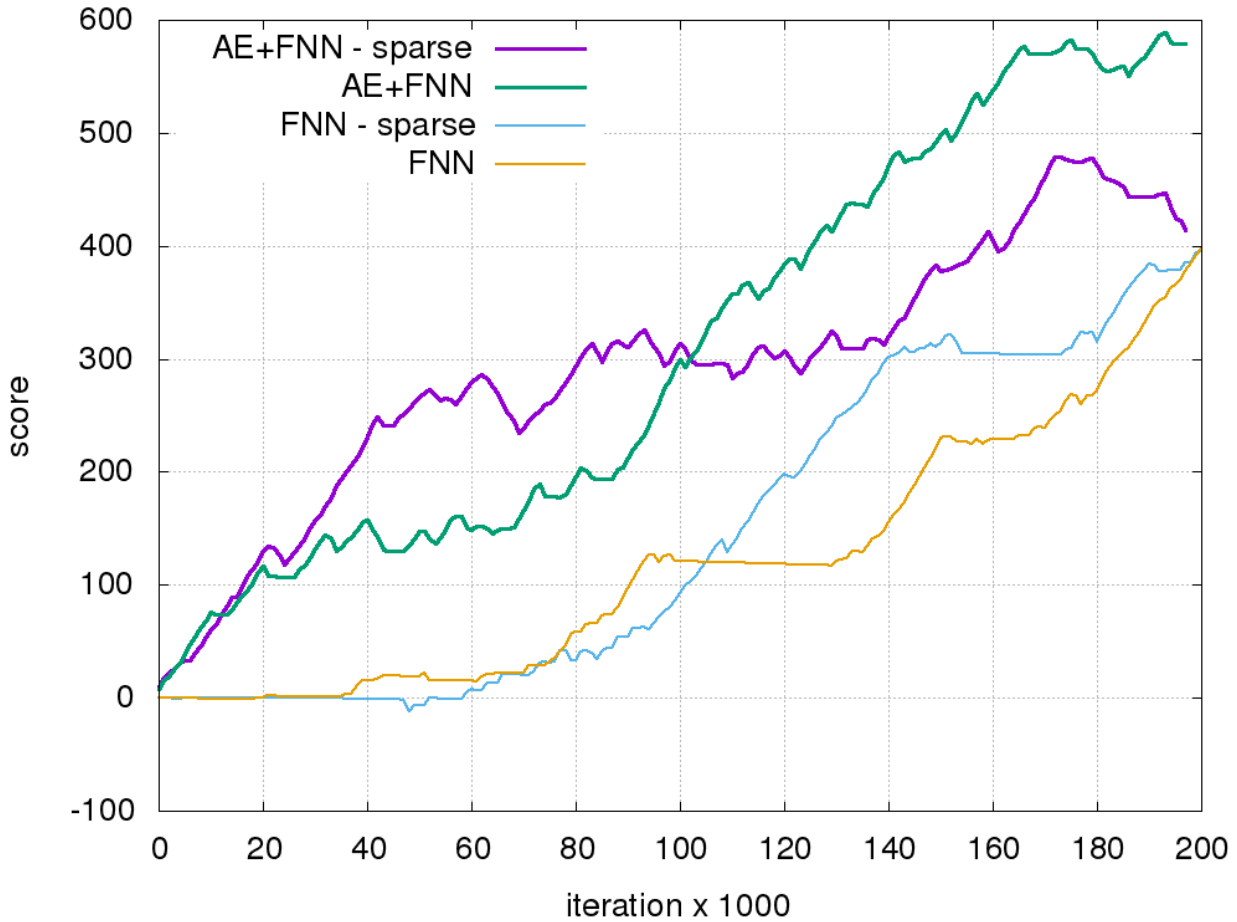




FNN score progress comparison

AE+FNN score progress comparison

Snake game experiment



Training worms score progress for best networks

Q&A

https://github.com/michalnand/robotics
https://github.com/michalnand/machine_learning

michal.nand@gmail.com