Storyline

1. Q-Learning
   1. Re-visit state s\_t, if already existing
   2. perform action with maximum value (action-value: q),
   3. recalculate q-value for taken action with r\_t, and expected, discounted reward based upon s\_t1
2. Too many possible states: even with lunar lander (8 inputs). What to do?
   1. ANN-Approximate / estimate q-value based on s\_t,
   2. perform action based on the argmax of the estimation of q-values,
   3. recalculate q-value for taken action with r\_t, and (*another approximation*, based on s\_t1) expected, discounted reward based upon s\_t1 (target for supervised learning problem)
   4. calculate error for taken action (mse)
   5. backprop error through ANN
3. Too shaky! Do experience replay: train ANN on batch of transitions every x-th timestep, in stead of single observations/s\_t each time step
4. Do proper parametrization: epsilon, alpha, gamma (dropout)
5. 1D inputs vs 2D inputs: use different input preprocessing
6. Computationally intensive!
   1. Use AWS.
   2. OK. But how about alternatives?: Policy Gradients TODO COMPARE
7. SUPPORT FOR 2
   1. What is a neural net; by this meaning fully-connected layer: Create a ANN from scratch
   2. Maybe: What is a convolution? TODO