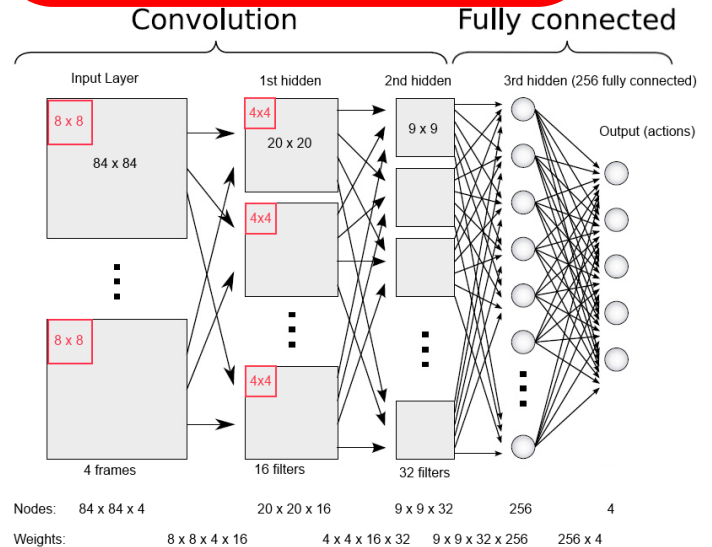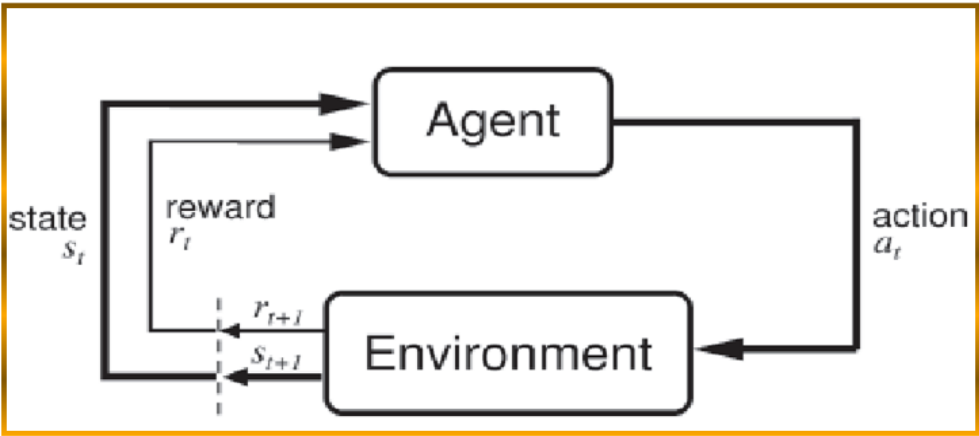# Replicating DeepMind

Computational neuroscience project by
Ardi, Ilya, Kristjan and Taivo

Tartu, Estonia, 2014/05/25

Playing Atari with Deep Reinforcement Learning

Results

	B. Rider	Breakout	Enduro	Pong	Q*bert	Seaquest	S. Invaders
Random	354	1.2	0	−20.4	157	110	179
Sarsa [3]	996	5.2	129	−19	614	665	271
Contingency [4]	1743	6	159	−17	960	723	268
DQN	**4092**	**168**	**470**	**20**	**1952**	**1705**	**581**
Human	7456	31	368	−3	18900	28010	3690

People associated with DeepMind

- Demis Hassabis
    - Poker player, computer game designer and neuroscientist

- Shane Legg
    - Universal Intelligence: A Definition of Machine Intelligence
        - *„Intelligence measures an agent's ability to achieve goals in a wide range of environments."*

- Jaan Tallinn
    - Investor of DeepMind

Structure of the DeepMind

- Play lots of games
  - Play lots of frames
    - Get a frame and **<u>pre-process it</u>**

Pre-processing

- Image
  1. 210 × 160 RGB
  2. 110 × 84 gray-scale
  3. 84 x 84

- Every 4th frames

- State contains 4 frames

- Reward is binarized

Structure of the DeepMind

- Play lots of games
  - Play lots of frames
    - Get a frame and <u>pre-process it</u>
    - Choose action using **<u>Deep Neural Net (DNN)</u>** and current state
      - Do it randomly with probability *e*

Convolution

Fully connected
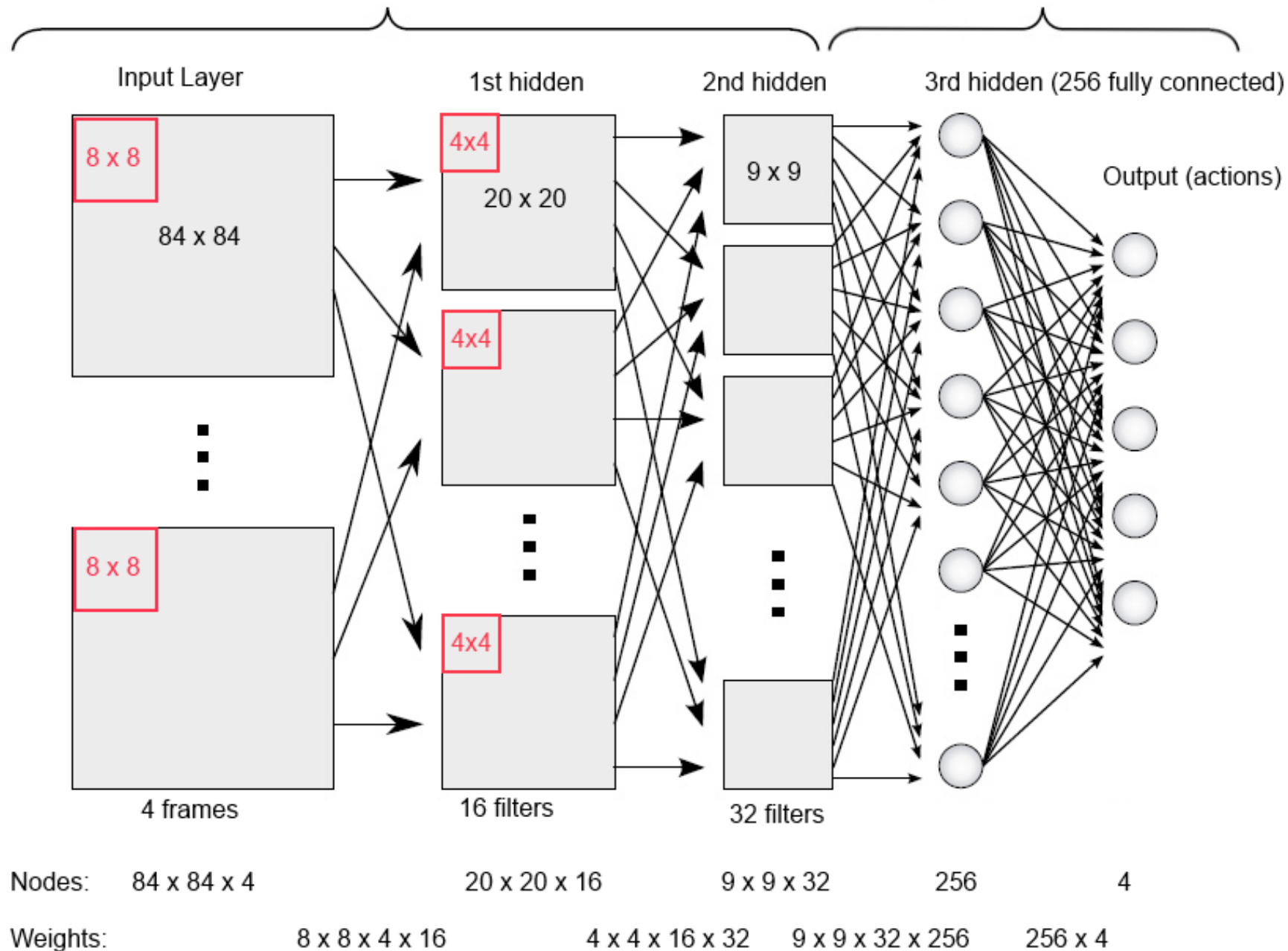
Input Layer

8 x 8

84 x 84

8 x 8

4 frames

1st hidden

4x4

20 x 20

4x4

4x4

16 filters

2nd hidden

9 x 9

32 filters

3rd hidden (256 fully connected)

Output (actions)

Nodes:	84 x 84 x 4		20 x 20 x 16	9 x 9 x 32	256	4
Weights:		8 x 8 x 4 x 16	4 x 4 x 16 x 32	9 x 9 x 32 x 256	256 x 4	

Structure of the DeepMind

- Play lots of games
  - Play lots of frames
    - Get a frame and pre-process it
    - Choose action using Deep Neural Net (DNN) and current state
      - Do it randomly with probability *e*
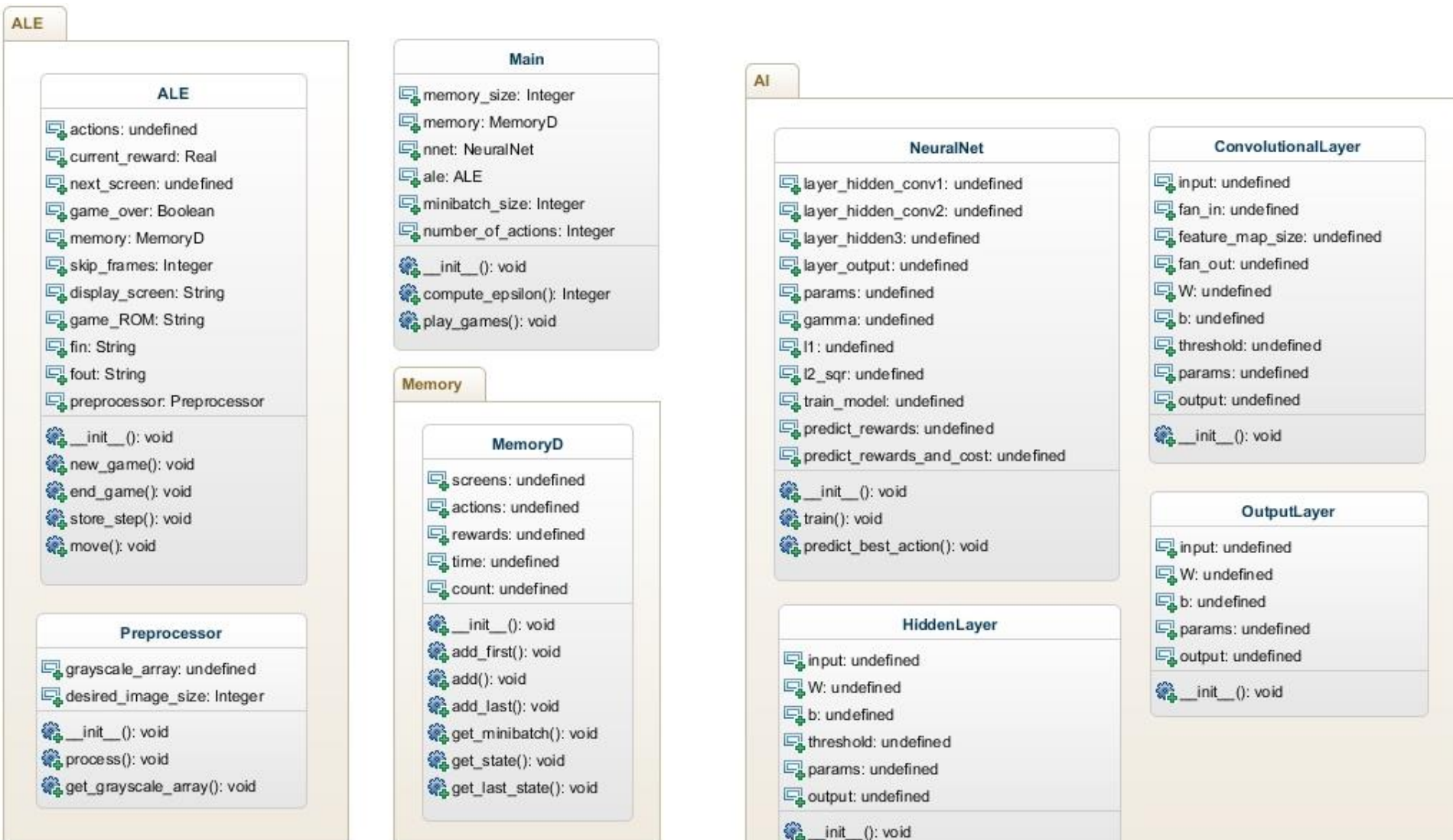   - Observe reward and next image and save everything to the memory
    - Train the network
      - 32 random transitions: 32 x *(state, reward, action, state)*
      - Q-learning
        - Reward $= r_t + \gamma \cdot DNN(\text{State}_{t+1})$
      - Gradient descent

Did we achieve our goal? Almost yes!

ALE

ALE
- actions: undefined
- current_reward: Real
- next_screen: undefined
- game_over: Boolean
- memory: MemoryD
- skip_frames: Integer
- display_screen: String
- game_ROM: String
- fin: String
- fout: String
- preprocessor: Preprocessor
- __init__(): void
- new_game(): void
- end_game(): void
- store_step(): void
- move(): void

Preprocessor
- grayscale_array: undefined
- desired_image_size: Integer
- __init__(): void
- process(): void
- get_grayscale_array(): void

Main
- memory_size: Integer
- memory: MemoryD
- nnet: NeuralNet
- ale: ALE
- minibatch_size: Integer
- number_of_actions: Integer
- __init__(): void
- compute_epsilon(): Integer
- play_games(): void

Memory

MemoryD
- screens: undefined
- actions: undefined
- rewards: undefined
- time: undefined
- count: undefined
- __init__(): void
- add_first(): void
- add(): void
- add_last(): void
- get_minibatch(): void
- get_state(): void
- get_last_state(): void

AI

NeuralNet
- layer_hidden_conv1: undefined
- layer_hidden_conv2: undefined
- layer_hidden3: undefined
- layer_output: undefined
- params: undefined
- gamma: undefined
- l1: undefined
- l2_sqr: undefined
- train_model: undefined
- predict_rewards: undefined
- predict_rewards_and_cost: undefined
- __init__(): void
- train(): void
- predict_best_action(): void

ConvolutionalLayer
- input: undefined
- fan_in: undefined
- feature_map_size: undefined
- fan_out: undefined
- W: undefined
- b: undefined
- threshold: undefined
- params: undefined
- output: undefined
- __init__(): void

OutputLayer
- input: undefined
- W: undefined
- b: undefined
- params: undefined
- output: undefined
- __init__(): void

HiddenLayer
- input: undefined
- W: undefined
- b: undefined
- threshold: undefined
- params: undefined
- output: undefined
- __init__(): void

What next?

- Need to understand and implement one algorithm
- Need to convert some stuff from CPU to GPU
- Need to parallelize some of the code
- Probably need to optimize some parameters

Replicating DeepMind

Computational neuroscience project by
Ardi, Ilya, Kristjan and Taivo

Tartu, Estonia, 2014/05/25