Playing FPS Games with Deep Reinforcement Learning

Devendra Singh Chaplot

with Guillaume Lample, Emilio Parisotto, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, Ruslan Salakhutdinov
Outline

• Learning to play deathmatches in Doom (AAAI-17)
• Learning to execute natural language instructions (AAAI-18)
• Learning to localize (ICLR-18)
Playing deathmatches in Doom

https://www.youtube.com/watch?v=oo0TraGu6QY
Executing language instructions

https://www.youtube.com/watch?v=JziCKsLrudE
Active Localization

https://www.youtube.com/watch?v=T5Ezx-_QfU0
Reinforcement Learning
Reinforcement Learning

Agent

Environment
Reinforcement Learning

Agent

State $S_t$

Environment

Go to the green torch
Reinforcement Learning

Agent

$\alpha_t$

Environment

$S_t$

Action

State

Go to the green torch

Carnegie Mellon University
Reinforcement Learning

Agent

Reward $r_t$

Action $a_t$

State $S_t$

Environment

Go to the green torch
Learning to play deathmatches
Motivation

Deepmind: Training an agent to play Atari games [Mnih et al. 2013]
Playing deathmatches

• Deathmatch: maximize the number of frags

• Challenges
  • Partially observable 3D environment
  • Deathmatch – involves learning a wide variety of skills: exploration, navigation, picking useful objects, and combat
  • Generalization to unknown maps
Deep Q-Learning

Conv 1
32 filters
Shape 8 x 8
Stride 4

Layer 1
15 feature maps
(5 frames x RGB)
Shape 60 x 108

Conv 2
64 filters
Shape 4 x 4
Stride 2

Layer 2
32 feature maps
Shape 14 x 26

Layer 3
64 feature maps
Shape 6 x 12

Layer 3'
Size 4008

Layer 4
Size 512

Action scores
(Size n for n actions)

[Mnih et al. 2013]
Deep Recurrent Q-Network (DRQN)

[Deep Recurrent Q-Network](Hausknecht and Stone, 2015)
Simple scenarios

https://www.youtube.com/watch?v=7JPNeE_ePRo

Defend the center

https://www.youtube.com/watch?v=W0mI_c2LjJM

Navigation
Arnold - Network Architecture

Layer 1
3 feature maps
Shape 60 x 108

Conv 1
32 filters
Shape 8 x 8
Stride 4

Layer 2
32 feature maps
Shape 14 x 26

Conv 2
64 filters
Shape 4 x 4
Stride 2

Layer 3
64 feature maps
Shape 6 x 12

Layer 3'
Size 4608

Layer 4
Size 512

Visual features
(Size k for k features)

LSTM

Action scores
(Size n for n actions)
Generalization
Generalization

• Domain randomization: Train on random textures
Random Textures
Random Textures
Random Textures
Random Textures
Random Textures
Random Textures
Random Textures
Demo Video

https://www.youtube.com/watch?v=IQK1zs2LDJ0
Importance of auxiliary tasks
Intelligent behavior

• Avoiding lava
• Moving backwards before shooting to avoid suicide
• Learning to crouch in order to minimize exposed surface area
• Learning to dodge rockets.
Results

• Beats average humans
• Undefeated against 32 human players at the AAAI-17 demo (Best Demo Award)
• Won the Full Deathmatch at Visual Doom AI Competition, 2017.
Transfer Learning

- Limited computing resources
- Long training times
Multi-task Reinforcement Learning
Multi-task Reinforcement Learning

• Single model to perform multiple tasks
Multi-task Reinforcement Learning

• Single model to perform multiple tasks
• Task can be given as input:
  • Program
  • Symbols
  • Language instructions
Multi-task Reinforcement Learning

• Single model to perform multiple tasks
• Task can be given as input:
  • Program
  • Symbols
  • Language instructions
• Zero-shot task generalization: Can the model generalize to new tasks not seen during training?
Learning to execute natural language instructions
Task-oriented language grounding

Train
- Go to the short red torch
- Go to the blue keycard
- Go to the largest yellow object
- Go to the green object

Test
- Go to the tall green torch
- Go to the red keycard
- Go to the smallest blue object

Go to the green torch
Challenges

• *recognize* objects in raw pixel input,
• *explore* the environment, handle occlusion
• *ground* each concept of the instruction in visual elements or actions,
• *reason* about the pragmatics of language, and
• *navigate* to the correct object while avoiding incorrect ones.

**Multitask Learning:** Single model to tackle multiple instructions  
**Zero-Shot Learning:** Generalize to unseen attribute-object pairs
Network overview

Go to the green torch

Natural Language Instruction \((L)\)

Image \((I_t)\)

State Processing Module

GRU Network \(g(L; \theta_{GRU})\)

Instruction Representation \(x_L = g(L; \theta_{GRU})\)

Multimodal Fusion \((M)\)

Conv Network \(f(I_t; \theta_{conv})\)

Image Representation \(x_t = f(I_t; \theta_{conv})\)

State Representation \(M(x_L, x_t)\)

Policy Learning Module

Policy \(\Pi(a | I_t, L)\)
Multimodal Fusion

• Baseline Approach: Concatenation
• Proposed Approach: Gated-Attention

• Gated-Attention [Dhingra et al. 2016]
  • attention weights for features maps, determines which filters to attend to
  • element-wise product (Gating)
  • creates instruction-specific convolutional filter representations
Gated-Attention

\[ x_l = f(L; \theta_{\text{conv}}) \]

\[ x_L = g(L; \theta_{\text{GRU}}) \]

\[ a_L = h(x_L) \]

\[ M_{GA}(x_L, x_l) = M(a_L) \odot x_l \]

To policy learning module
Policy Learning

• Asynchronous Advantage Actor-Critic (A3C) [Mnih et al. 2016]
  • uses a deep neural network to parametrize the policy and value functions and runs multiple parallel threads to update the network parameters.
  • use entropy regularization for improved exploration
  • use Generalized Advantage Estimator to reduce the variance of the policy gradient updates (Schulman et al.)
Environment

• 18 objects
• 5 types of objects
• Different colors and sizes
• Superlative instructions:
  • Largest, smallest
• Combinations
  • Tall green torch
  • Largest red object
• 70 instructions
Environment difficulty
Environment difficulty

EASY

MEDIUM

[Images of game environments: Easy and Medium]
Environment difficulty

- **Easy**
- **Medium**
- **Hard**
Results

Easy

Accuracy

Time (hrs)

A3C GA
A3C Concat

Medium

Accuracy

Time (hrs)

A3C GA
A3C Concat

Hard

Accuracy

Time (hrs)

A3C GA
A3C Concat
Demo Video

https://www.youtube.com/watch?v=o_G6was03N0
Attention map
Attention map

Instruction Representation
\[ x_L = g(L; \theta_{GRU}) \]

Attention Vector
\[ a_L = h(x_L) \]

- blue
- armor
- pillar
- torch
- skullkey
Attention map

Instruction Representation
\[ x_L = g(l; \theta_{GRU}) \]

\[ a_L = h(x_L) \]

Attention Vector

blue
red

armor
pillar
torch
skullkey
Attention map

blue
red
green

armor
pillar
torch
skullkey

Instruction Representation
\[ x_L = g(L; \theta_{GRU}) \]

Attention Vector

\[ a_L = h(x_L) \]

Carnegie Mellon University
Attention map

Instruction Representation
\( x_L = g(l; \theta_{GRU}) \)

Attention Map

- blue
- red
- green
- yellow
- armor
- pillar
- torch
- skullkey
t-SNE Visualizations
t-SNE Visualizations
t-SNE Visualizations
t-SNE Visualizations

- Red
- Blue
- Green
- Yellow
- Color unspecified

- Largest
- Smallest
- Tall
- Short
- Size unspecified

- Armor
- Pillar
- Torch
- Keycard
- Skullkey
- Type unspecified
Planning

- Learning to play deathmatches and execute natural language instructions mostly require *reactive* policies
  - Agent doesn’t require long-term memory
  - It can decide actions based on last few frames
Planning

• Learning to play deathmatches and execute natural language instructions mostly require reactive policies
  • Agent doesn’t require long-term memory
  • It can decide actions based on last few frames

• Many real-world tasks require long-term planning
Planning

• Learning to play deathmatches and execute natural language instructions mostly require reactive policies
  • Agent doesn’t require long-term memory
  • It can decide actions based on last few frames

• Many real-world tasks require long-term planning
• One of pre-requisites for planning is ability to predict it’s own location
Learning to localize
Localization
Localization

Estimating the location of an autonomous agent given:
Localization

Estimating the location of an autonomous agent given:

• a map of the environment
Localization

Estimating the location of an autonomous agent given:

- a map of the environment
- Agent observations
Motivation

• Localization is considered as the **basic precondition for truly autonomous agents** by Burgard et al. (1998)

• Downstream tasks: target-navigation, planning

• Applications: autonomous vehicles, factory robots, housekeeping robots, delivery drones
Passive Localization

Agent Observations

Map Information

Localization

Predictions

$t = 1$

$t = 2$
Active Localization

Agent Observations

$t = 1$

$t = 2$

Map Information

Active Localization

Predictions

$x$ $y$ $o$

Location

Action

$x$ $y$ $o$

Location

Action

Carnegie Mellon University
Active Localization

Agent Observations

Map Information

Active Localization

Predictions

Location

Action

$t = 1$

$t = 2$
Active Neural Localization

\( y_t \): Random variable denoting location of the agent at time \( t \)
\( s_t \): Agent’s observation at time \( t \)
\( a_t \): Action taken by the agent at time \( t \)
\( M \): Information given about the Map
\( \odot \): Element-wise dot product

\( Bel(y_{t-1}) \): Belief of the location of the agent at time \( t \) before observing \( s_t \)
\( Bel(y_t) \): Belief of the location of the agent at time \( t \) after observing \( s_t \)
\( Lik(s_t) \): Likelihood of observing \( s_t \) in each state \( y_t \)
\( \pi(a_t|Bel(y_t)) \): Policy learnt by the agent, probability of taking action \( a_t \) given \( Bel(y_t) \)
\( f_T \): Transition function
Representation of Belief and Likelihood

$x$ $y$ $\theta$

$O \times M \times N$

Map size

Number of orientations
Representation of Belief and Likelihood

3-dimensional tensor representing $x$-coordinate, $y$-coordinate and orientation

$O \times M \times N$

Map size

Number of orientations
Representation of Belief and Likelihood

3-dimensional tensor representing $x$-coordinate, $y$-coordinate and orientation

Each element represents the probability of the agent being present in the corresponding location

$O \times M \times N$

Map size

Number of orientations
Demo video: Doom

https://www.youtube.com/watch?v=rdhKu8GqVlw
Demo video: Unreal

https://www.youtube.com/watch?v=T5Ezx-_QfU0
Results: Accuracy

Accuracy

Markov Localization (Resnet)
Active Markov Localization (Slow)

Active Markov Localization (Fast)
Active Neural Localization

Number of landmarks
Test Setting
Train Setting

Unseen Mazes
Seen Textures
Maze3D
Unseen mazes
Unseen textures

With lights
Without lights
Unreal3D with lights
All
Maze3D to Unreal3D
Domain adaptation

Carnegie Mellon University
### Results: Runtime

<table>
<thead>
<tr>
<th>Test Setting</th>
<th>Number of landmarks</th>
<th>Train Setting</th>
<th>Number of landmarks</th>
<th>Test Setting</th>
<th>Number of landmarks</th>
<th>Train Setting</th>
<th>Number of landmarks</th>
<th>Test Setting</th>
<th>Number of landmarks</th>
<th>Train Setting</th>
<th>Number of landmarks</th>
<th>Test Setting</th>
<th>Number of landmarks</th>
<th>Test Setting</th>
<th>Number of landmarks</th>
<th>Test Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unseen Mazes</td>
<td>10</td>
<td>Unseen mazes</td>
<td>10</td>
<td>Unseen mazes</td>
<td>10</td>
<td>Unseen mazes</td>
<td>10</td>
<td>Unseen mazes</td>
<td>10</td>
<td>Unseen mazes</td>
<td>10</td>
<td>Unseen mazes</td>
<td>10</td>
<td>Unseen mazes</td>
<td>10</td>
<td>Unseen mazes</td>
</tr>
<tr>
<td>Seen Textures</td>
<td>5</td>
<td>Seen textures</td>
<td>5</td>
<td>Seen textures</td>
<td>5</td>
<td>Seen textures</td>
<td>5</td>
<td>Seen textures</td>
<td>5</td>
<td>Seen textures</td>
<td>5</td>
<td>Seen textures</td>
<td>5</td>
<td>Seen textures</td>
<td>5</td>
<td>Seen textures</td>
</tr>
<tr>
<td>Maze3D</td>
<td></td>
<td></td>
<td></td>
<td>Unreal3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unreal3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>With lights</td>
<td></td>
<td></td>
<td></td>
<td>With lights</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>With lights</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Without lights</td>
<td></td>
<td></td>
<td></td>
<td>Without lights</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Without lights</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unreal3D with lights</td>
<td></td>
<td></td>
<td></td>
<td>Unreal3D with lights</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unreal3D with lights</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>All</td>
<td></td>
<td></td>
<td></td>
<td>All</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>All</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maze3D to Unreal3D</td>
<td></td>
<td></td>
<td></td>
<td>Maze3D to Unreal3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Maze3D to Unreal3D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Markov Localization (Resnet)**
- **Active Markov Localization (Fast)**
- **Active Markov Localization (Slow)**
- **Active Neural Localization**
Summary: Playing Deathmatches

• Using auxiliary tasks to improve sample efficiency
  • Helps training convolutional filters to detect useful entities
  • Unsupervised Auxiliary Tasks (Jaderberg et al. 2016)
  • Learning to Navigate (Mirowski et al. 2016)

• Domain Randomization for generalization
  • Generalize to unknown maps using random textures
  • Simulation to the real world transfer (Tobin et al. 2017)
Summary: Language Grounding

• Gated-attention for multimodal fusion
  • Effective
  • Somewhat interpretable

• Zero-shot task generalization using language grounding
Summary: Active Localization

• Spatially-structured representation of belief and multiplicative interaction for belief propagation -> fully differentiable

• Perform end-to-end active localization with deep reinforcement learning
  • Allows perceptual model and policy model to be trained jointly
  • Doesn’t require labels, needs only a reward at the end of the episode

• Effective and efficient localization
Publications

• **Active Neural Localization**
  Devendra Singh Chaplot, Emilio Parisotto, Ruslan Salakhutdinov. (2018)
  6th International Conference on Learning Representations (ICLR-18), Vancouver, Canada

• **Gated-Attention Architectures for Task-Oriented Language Grounding**
  Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, Ruslan Salakhutdinov. (2018)
  32nd AAAI Conference on Artificial Intelligence (AAAI-18), New Orleans, USA.

• **Playing FPS Games with Deep Reinforcement Learning**
  Guillaume Lample*, Devendra Singh Chaplot*. (2017)
  31st AAAI Conference on Artificial Intelligence (AAAI-17), San Francisco, USA.

• **Arnold: An Autonomous Agent to play FPS Games** (Best Demo Award)
  Devendra Singh Chaplot*, Guillaume Lample*. (2017)
  31st AAAI Conference on Artificial Intelligence (AAAI-17), San Francisco, USA. (demo)
• **MIT TechReview** – “Machines Are Developing Language Skills Inside Virtual Worlds”

• **Techcrunch** – “Scientists teach machines to hunt and kill humans— in Doom deathmatch mode”

• **Popular Science** – “Trained A.I. beats humans in Doom deathmatches”

• **Salon** – “Meet your “Doom”: Carnegie Mellon researchers deliberately violate Asimov’s First Law of robotics, teach robots to kill”

• **Kotaku** – "Doom Bot Learns To Play Better Than Humans”

• **CMU news** – “Computer Out-Plays Humans in "Doom””

and many more …
Code

• Playing deathmatches
  • https://github.com/glample/Arnold

• Language grounding
  • https://github.com/devendrachaplot/DeepRL-Grounding

• Active Neural Localization
  • https://github.com/devendrachaplot/Neural-Localization
Contact

Devendra Singh Chaplot
Webpage: http://devendrachaplot.github.io/
Email: chaplot[at]cs.cmu.edu
Twitter: @dchaplot

Thank you!