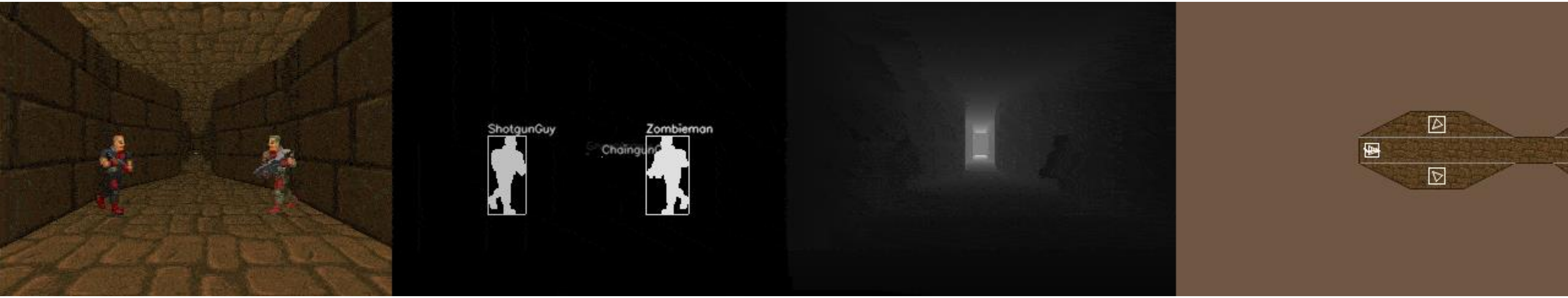
Prof. Kuan-Ting Lai

2020/6/5

HW3 - VizDoom

VizDoom

- https://github.com/mwydmuch/ViZDoom
- A RL learning environment for playing Doom, a popular shooting game
- M. Kempka, M. Wydmuch, G. Runc, J. Toczek & W. Jaśkowski, ViZDoom: A Doom-based AI Research Platform for Visual Reinforcement Learning, IEEE Conference on Computational Intelligence and Games, pp. 341-348, Santorini, Greece, 2016



Installing VizDoom

- Linux
  - Installing dependencies
    - https://github.com/mwydmuch/ViZDoom/blob/master/doc/Building.md#linux_deps
  - sudo pip install vizdoom

- Windows
  - Download pre-built
    - **1.1.7 (2018-12-29):**
    - Python 2.7 (64-bit)
    - Python 3.5 (64-bit)
    - Python 3.6 (64-bit)
    - Python 3.7 (64-bit)

Running VizDoom on Windows

- Download Windows pre-built Python 3.6 (64-bit)
- Extract to c:\vizdoom
- Run “C:\vizdoom\vizdoom.exe” for a quick check
- You should see Freedoom 0.11.3 launched!

Running VizDoom on Windows using Python

- Download and install Python 3.5.4 (64bit) – be sure to use 64bit!

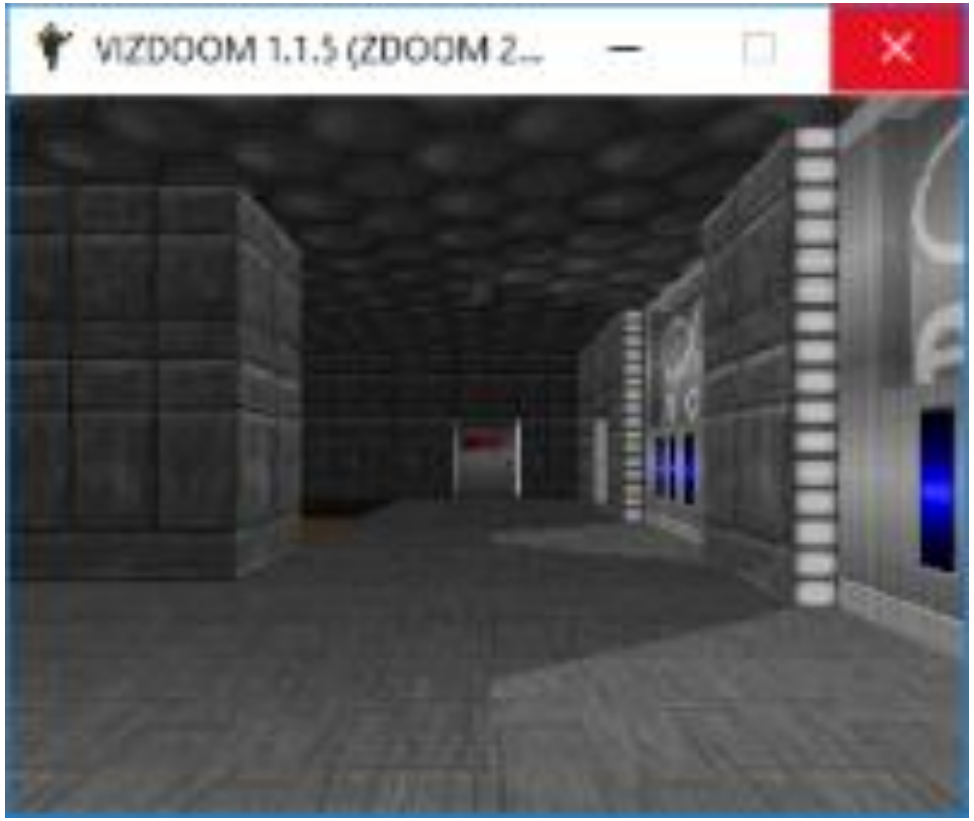
- Open the Windows System Properties dialog (search for “Edit the system environment variables” in Windows Search) and Click on “Environment Variables…”

- Create a new environment variable called PYTHONPATH with the value C:\vizdoom

- If your shell window is opened, close and re-open your shell

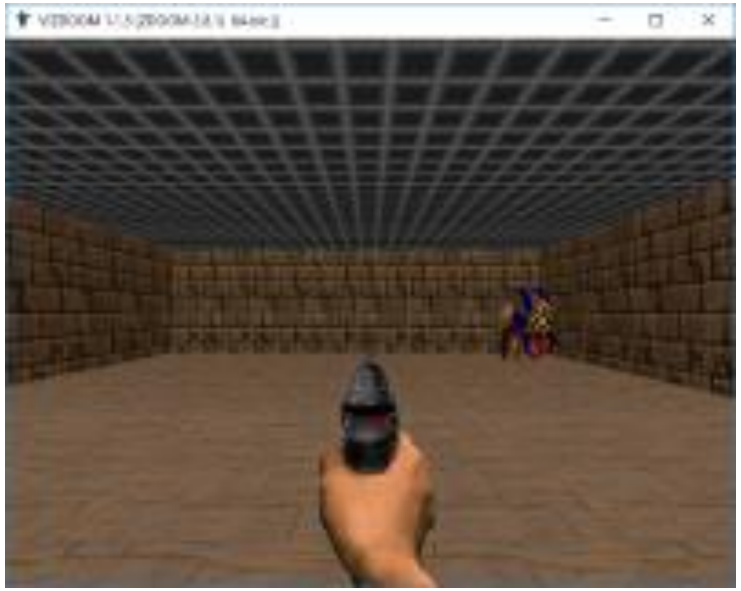
Test Your Environment

- Open a Windows PowerShell, type python to open Python’s interactive mode and enter
  - import vizdoom
  - game = vizdoom.DoomGame()
  - game.init()



Running basic.py

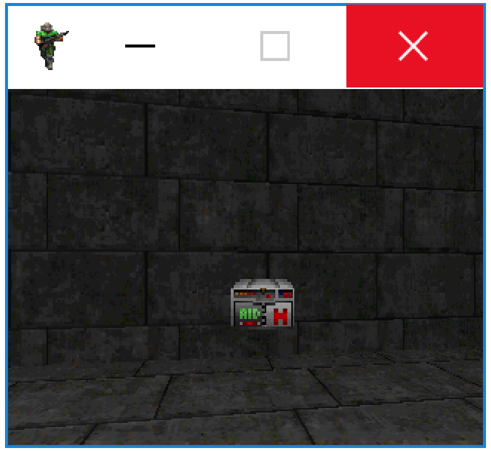
- Now edit “C:\vizdoom\examples\basic.py” with a text or Python editor of your choice and change
  - game.set_doom_scenario_path(“../../scenarios/basic.wad”)
- to
  - Game.set_doom_scenario_path(“../scenarios/basic.wad”)
- Open c:\vizdoom\examples in Windows PowerShell and type
  - python basic.py



Training VizDoom on Windows

Running example “learning_tensorflow.py”

- We provide a new map “D3_battle.wad” and config file “D3_battle.cfg”
- Put “D3_battle.wad” and “D3_battle.cfg” into “c:/vizdoom/scenarios/”
- Open and edit “C:\vizdoom\examples\learning_tensorflow.py”, change default config file path:
  - DEFAULT_CONFIG (“../scenarios/D3_battle.cfg”)



Uploading Your Scores to Kaggle

- We only have one test environment but the Kaggle results require two rows (one for public and one for private). Please fill-in your final training score in both rows
- We use Absolute Error (AE), the target score is 100. The lower, the better
- Kaggle will convert your score into AE automatically. **Just enter your game score!**
- We will invite the top 8 players to compete in a deathmatch

Id	Predicted
d3_battle_pubic	7
d3_battle_private	7

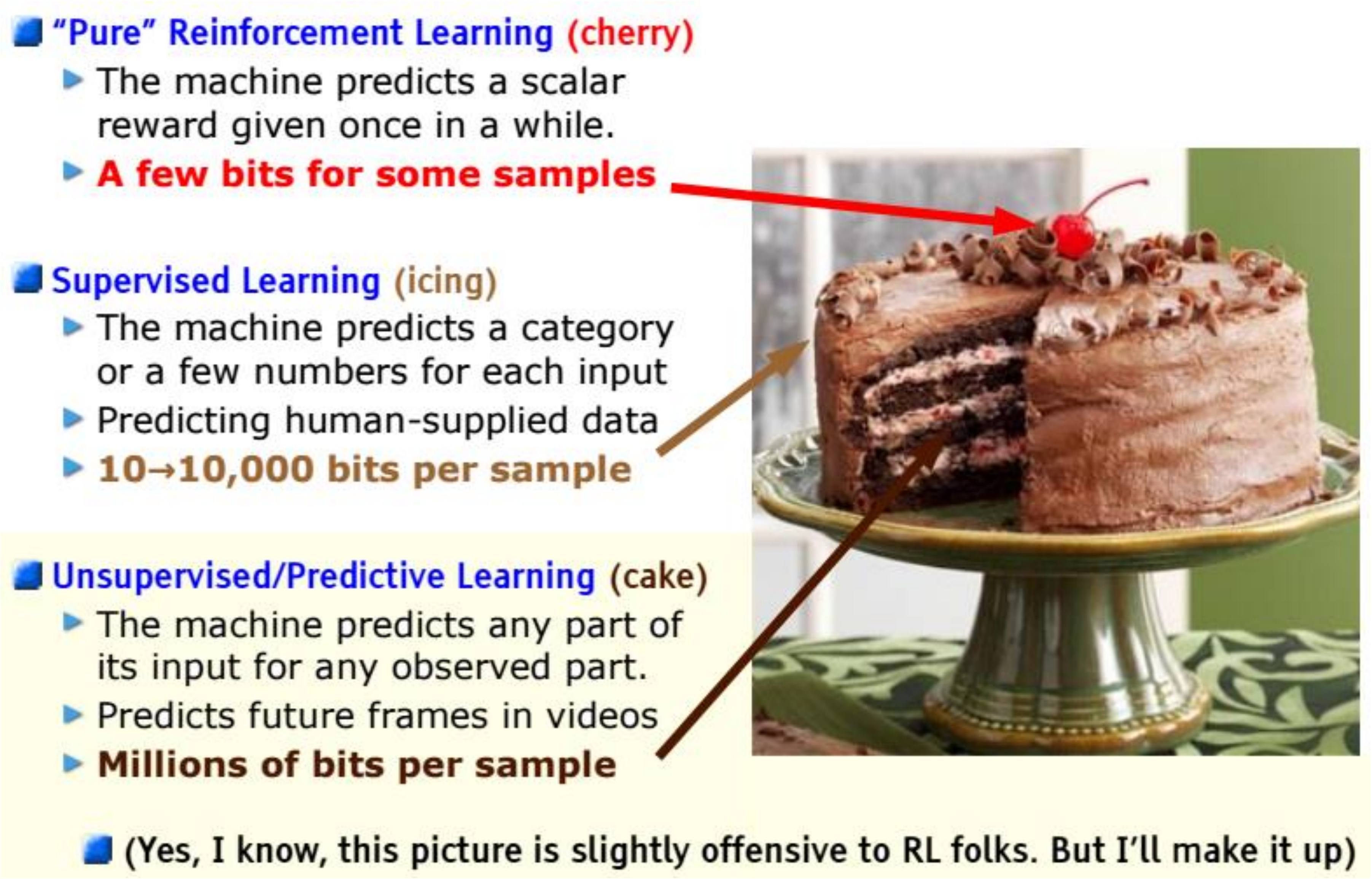
Example Algorithm: Direct Future Prediction

- Concepts of Direct Future Prediction (DFP)
  - https://flyyufelix.github.io/2017/11/17/direct-future-prediction.html
- Winner of 2017 VizDoom Competition
- Outperform other algorithms (including A3C and variants of DQN) by more than 50%

Reformulate RL as Supervised Learning

- Reinforcement learning vs. Supervised Learning
- Jordan & Rumelhart (1992) argue that
  - RL may be more efficient when the environment provides only a sparse scalar reward signal
  - SL can be advantageous when dense multidimensional feedback is available.

Yann LeCun’s Cake



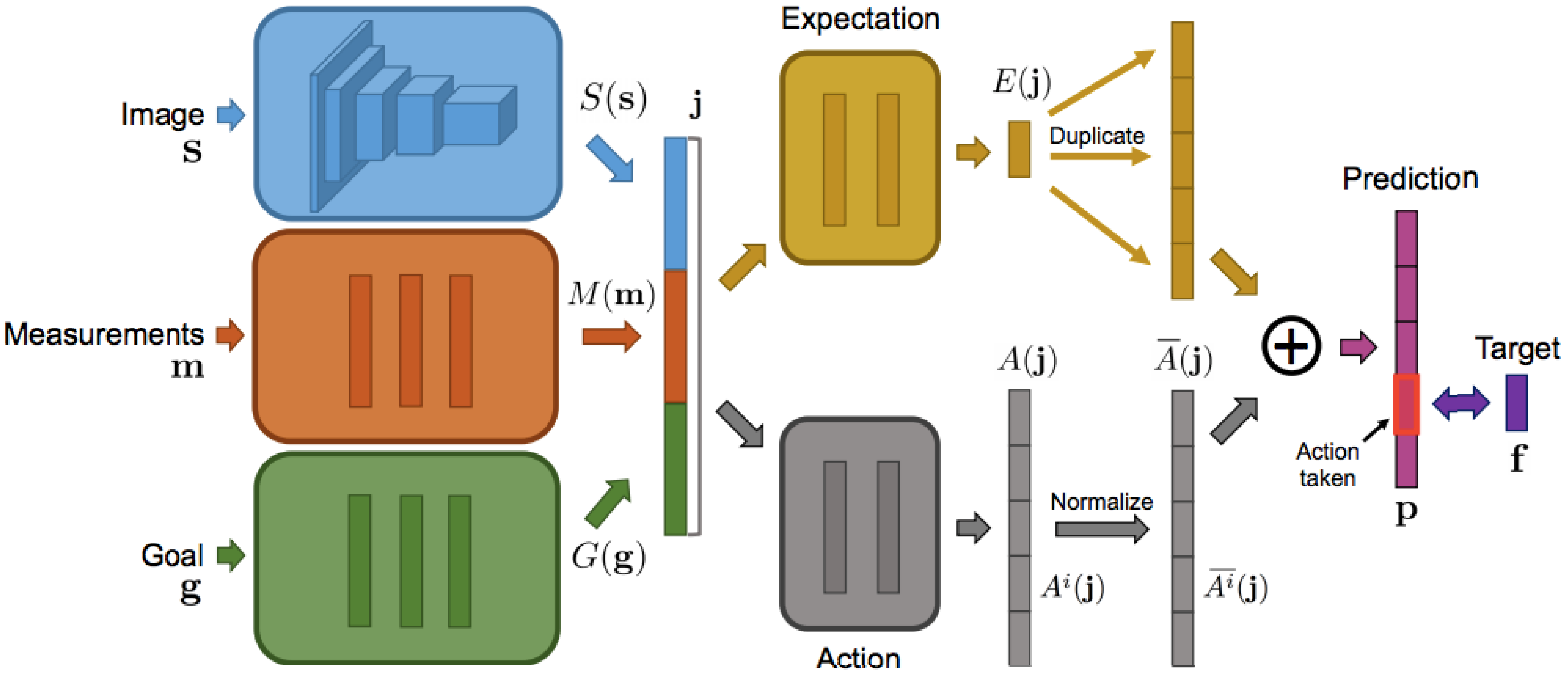
- **“Pure” Reinforcement Learning** (cherry)
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning** (icing)
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning** (cake)
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**
- (Yes, I know, this picture is slightly offensive to RL folks. But I’ll make it up)

Set a Unified Objective

- $U = f(m_t) = g \cdot m_t = 1 \times Kills - 0.5 \times Ammo\_used + 0.5 \times Health$
- $g = [1, -0.5, 0.5]$ is called goal vector
- Advantages:
  - Stabilize and accelerate training
  - Pursuing different goals at inference
    - $U = 1 \times kills + 0 \times Heath + 0 \times HealthPacks$

DFP Architecture

- Reward comes in the form of measurement ***m***
- Objective function $U = g \cdot m$



Feel free to ask me any questions if you’re stuck.

email:john0952270878@gmail.com