18-461/661: Current Topics in Machine Learning

Roshan Sharma, Jinhang Zuo, Yafei Hu, Yu-Jhe Li, Ching-Yi Lin, and Sundar Anand

Spring 2022

Carnegie Mellon University

Announcements

- HW 7 deadline is extended to Sunday, May 1. I strongly encourage you to complete this assignment early, so that you have time to study for the final exam.
- A practice final exam will be released before Wednesday’s lecture. Recitation on Friday will go over this practice exam.
- Wednesday’s lecture will be a summary of the course and detailed outline of logistics, suggested topics, etc. for the final exam.

Machine Learning for Speech and Language Processing

Roshan Sharma
CMU ECE

Carnegie Mellon University

Live Demo: Speech Recognition using Neural Networks

Overview

- Live Speech Recognition Demonstration
- Why Speech and NLP ?
- Sequence Models
- Types of Machine Learning Tasks and Examples
- Recent Trends: Self-Supervised Learning

Carnegie Mellon University

Why Speech Processing ?

Speech and text are the most natural means of communication.

- Natural: No additional training required
- Flexible: Adapt to dialogue partners, environment and situations
- Efficient: Communicate large amounts of information in lesser time
- Informative: about speaker, their cognitive-state, environment

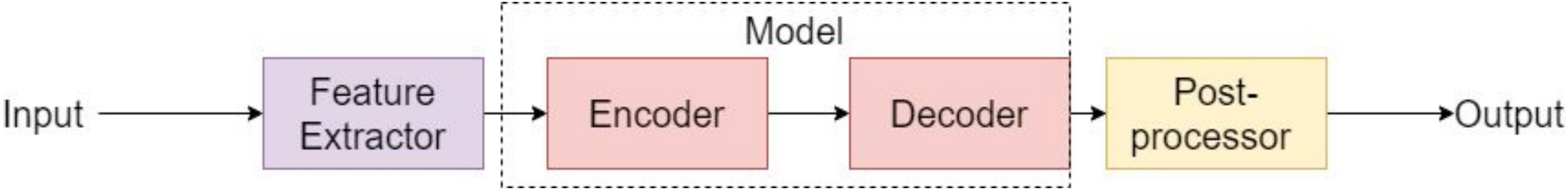
Though we have made incredible progress over the past few years, there remain many open research questions on machine understanding of text and speech!

Sequence Models

- Most Speech and NLP tasks rely on sequential information in speech frames or language tokens.
- For example, in speech recognition, a sequence of speech frames can correspond to one phoneme, and knowing past phonemes helps predict future phonemes.
- Sequence models take in a sequence of input features, and produce a sequence of labels as outputs.
- There are a few popular models for sequence based tasks: Hidden Markov Models, Neural Networks (Recurrent Neural Networks and Transformers)

System Overview: Speech/ NLP

For speech processing, the features are typically Mel Frequency Cepstral Coefficients or log Mel magnitude spectra
For NLP tasks, feature extraction is generally tokenization.



Types of tasks in Speech/NLP

- An example of **Regression** is Speech Enhancement where the goal is to learn a multiplier mask to attenuate noise from speech
- Emotion Recognition is a **many-to-one Multiclass classification** task, where the model outputs emotion labels from pooled multi-frame inputs
- Speech Recognition/Machine Translation is a **many-to-many Multiclass Classification** task, where the model outputs multiple language tokens corresponding to multiple frames of input speech
- Audio captioning/ Response Generation is an example of a **one-to-many Multiclass Classification** task

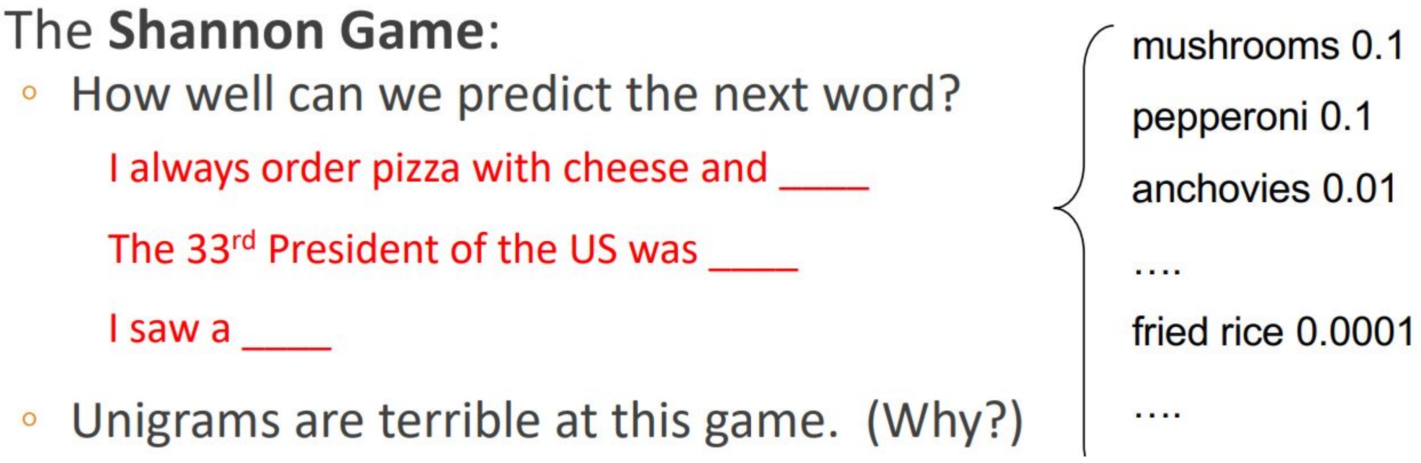
Examples of Speech and NLP Tasks

- Speech Recognition
- Speech Enhancement
- Speech Separation
- Speech Translation
- Speech Activity Detection
- Speech Synthesis
- Speaker Diarization
- Named Entity Recognition

- Speaker Recognition
- Language Identification
- Voice Conversion
- Intent Detection
- Emotion Recognition
- Dialog Models
- Wake-word Detection
- Natural Language Inference

- Machine Translation
- Language Modelling
- Information Extraction
- Response Generation
- Parsing
- Entity Extraction
- Structured Prediction
- Question Answering
- Summarization

Example: Language Modelling



The **Shannon Game**:

- How well can we predict the next word?
  - I always order pizza with cheese and ____
  - The 33rd President of the US was ____
  - I saw a ____
- Unigrams are terrible at this game. (Why?)

mushrooms 0.1

pepperoni 0.1

anchovies 0.01

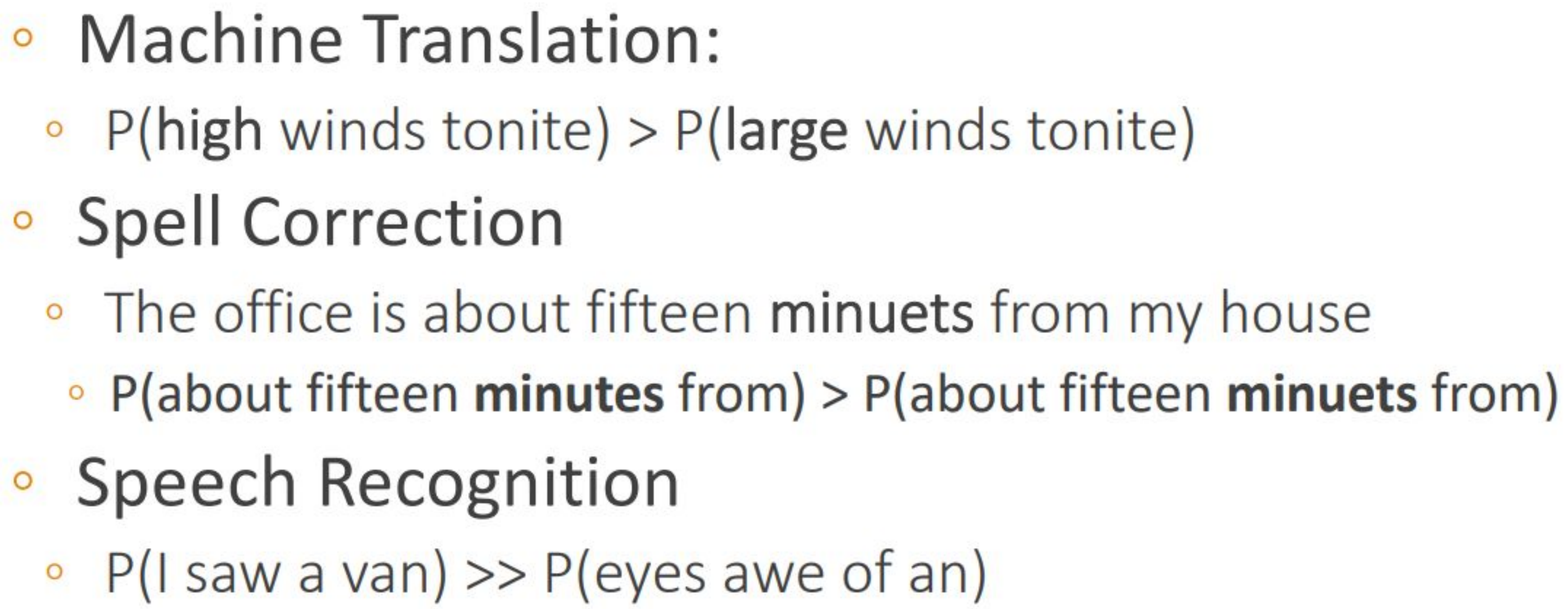
….

fried rice 0.0001

….

The goal of language modelling is to predict the most likely next word based on previous words

Language Modelling: Sentence Probabilities



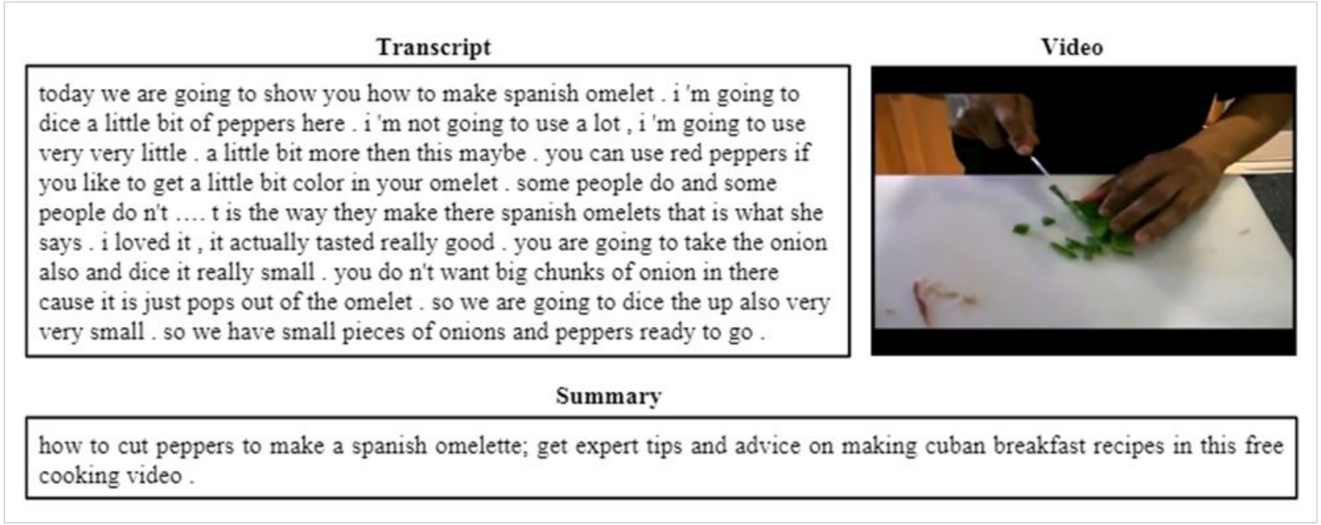
- Machine Translation:
  - P(**high** winds tonite) > P(**large** winds tonite)
- Spell Correction
  - The office is about fifteen **minuets** from my house
    - P(about fifteen **minutes** from) > P(about fifteen **minuets** from)
- Speech Recognition
  - P(I saw a van) >> P(eyes awe of an)

Example: Speech Recognition

- Hybrid Architecture with a Hidden Markov Model and MLP based acoustic model, language model, pronunciation model. Widely used in industry
- Connectionist Temporal Classification models with a speech encoder a per-frame token classification
- Attention-based models with a speech encoder, language decoder and optional language model, where attention is used to learn the mapping between speech frames and language tokens

Example: Speech Summarization

Given input speech, the goal is to produce an abstractive textual summary of the spoken content embedded.



Transcript

today we are going to show you how to make spanish omelet . i 'm going to dice a little bit of peppers here . i 'm not going to use a lot , i 'm going to use very very little . a little bit more then this maybe . you can use red peppers if you like to get a little bit color in your omelet . some people do and some people do n't .... t is the way they make there spanish omelets that is what she says . i loved it , it actually tasted really good . you are going to take the onion also and dice it really small . you do n't want big chunks of onion in there cause it is just pops out of the omelet . so we are going to dice the up also very very small . so we have small pieces of onions and peppers ready to go .

Video

Summary

how to cut peppers to make a spanish omelette; get expert tips and advice on making cuban breakfast recipes in this free cooking video .

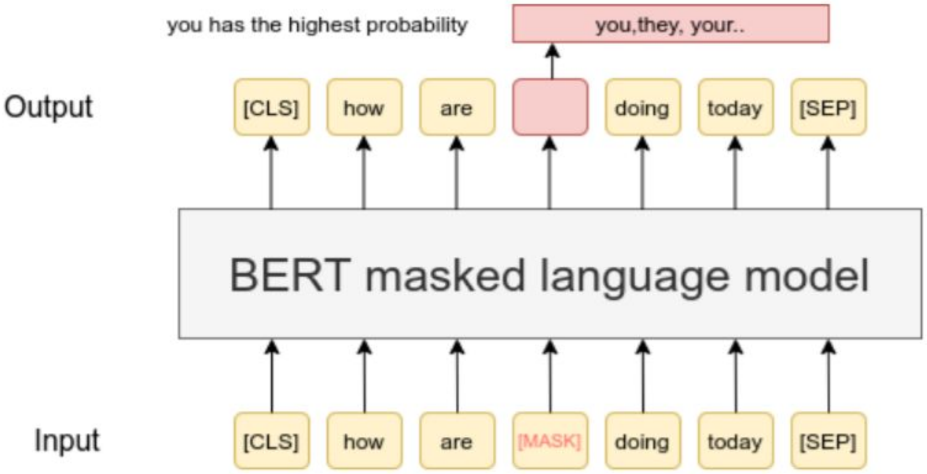
Current Research Trends

- Novel Tasks: Long Document QA, Speech Summarization, Structured Learning
- Novel Models: Transformers, Conformers, Cosformers, Longformers ......
- Novel Learning Methodologies: Self-Supervised Learning, Continual Learning, Curriculum Learning
- Additional Inputs: Multimodal, Multi-lingual, Code-switching, Multi-microphone, Low Resource, Multiple Simultaneous Talkers
- Novel Feature Extraction Approaches: STRFs, Byte Pair Encodings,

Language Understanding from Speech: "There is more in Speech than what is transcribed"

Self-Supervised Learning: A brief introduction

- A form of unsupervised learning where the data provides the supervision
- In general, withhold some part of the data, and task the network with predicting it
- The task defines a **proxy loss**, and the network is forced to learn what we really care about, e.g. a semantic representation, in order to solve it



Recommended CMU Courses in this Area

- 18781/ 11751- Speech Recognition and Understanding
- 11785 - Introduction to Deep Learning
- 11711- Advanced Natural Language Processing

Combinatorial Multi-armed Bandits

Jinhang Zuo

CMU ECE

Overview

- What is multi-armed bandits (MAB)
- Why combinatorial multi-armed bandits (CMAB)
- Applications & research directions

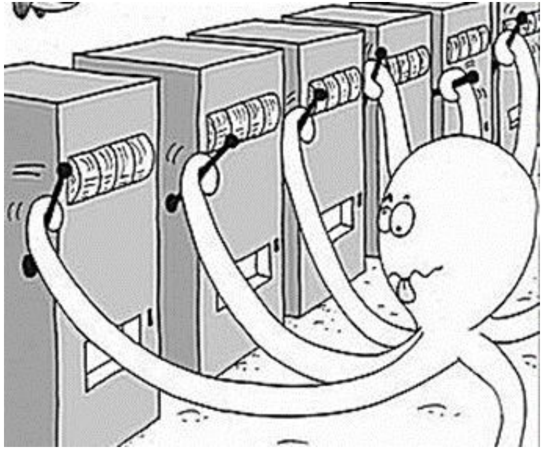
Multi-armed Bandits (MAB)

- There are K arms
- Arm i has unknown reward distribution with unknown mean μ_i (best arm $\mu^* = \max \mu_i$)
- In each round, the player selects one arm to play and observes the reward
- Performance metric: Regret

  - Regret after playing T rounds $= T\mu^* - \sum_{t=1}^{T} \mu_{A(t)}$
- Objective: minimize regret in T rounds
- Balancing exploration-exploitation

  - exploration: try new arms

  - exploitation: keep playing the best arm so far



Carnegie Mellon University

UCB (Upper Confidence Bound) Algorithm

1. Play each arm once
2. In each round $t > K$, play arm $\operatorname*{argmax}_{i \in [K]} \mathrm{UCB}_i(t)$, where:

$$\mathrm{UCB}_i(t) = \overset{\text{exploitation}}{\boxed{\hat{\mu}_i(t)}} + \overset{\text{exploration}}{\boxed{\sqrt{\frac{2 \log t}{n_i(t)}}}}$$

$\hat{\mu}_i(t)$ is the average reward obtained from arm i

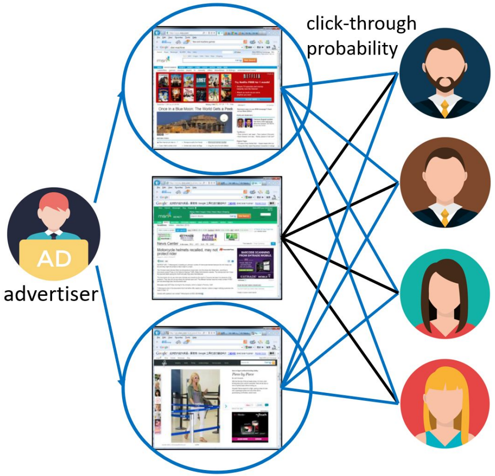
$n_i(t)$ is the number of times arm i has been played so far

Known result: regret upper bound $O(\sqrt{KT \log T})$

Carnegie Mellon University

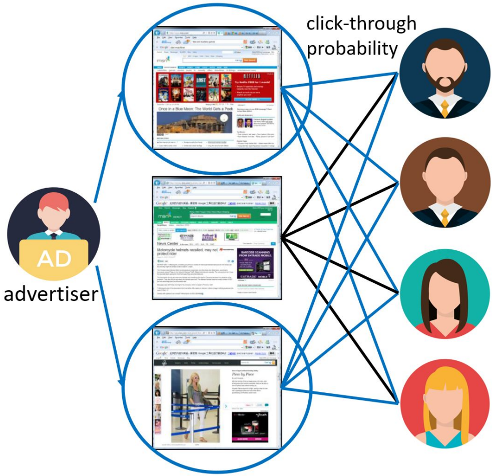
CMAB Example: Ad Placement

- An advertiser puts ads on web pages to attract user clicks
- Each user has a click-through probability on a certain page
- Offline problem:
  Given click-through probabilities, find *k* pages to maximize total # of user clicks (can be solved approximately)
- Online problem:
  With **unknown** click-through probabilities, find *k* pages to maximize the total # of user clicks
- Falls into the CMAB framework



CMAB Example: Ad Placement

- **Base arm**: edges between pages and users, with unknown click-through probabilities.
- **Super arm**: *k* pages that trigger a set of base arms
  – Challenge: exponential combinations of pages
- **Semi-bandit feedback**: outcomes of all triggered base arms are observed (users click the ad or not).
- **Objective**:
  find *k* pages to maximize the total # of user clicks



CUCB Algorithm [1]

1. Observe each base arm $i \in [K]$ once
2. In each round t, play super arm $S_t = \underset{S}{\operatorname{argmax}}\, r(S, UCB)$,

   where:

$$\mathrm{UCB}_i(t) = \overset{\text{exploitation}}{\boxed{\hat{\mu}_i(t)}} + \overset{\text{exploration}}{\boxed{\sqrt{\frac{2\log t}{n_i(t)}}}}$$

$\hat{\mu}_i(t)$ is the average outcome of base arm i (semi-bandit feedback)

$n_i(t)$ is the number of times base arm i has been played

Known result: regret upper bound $O(K\sqrt{T\log T})$

Applications & Research Directions

- Applications

Cold-start problem for recommender systems (e.g., learning to rank [2])
Online resource allocation for networked systems (e.g., dynamic channel allocation [3])
Online social influence maximization (e.g., ad placement, viral marketing [1, 4])

- Recent Trends:

Large-scale structured problems: metadata-based bandits [5], CMAB with correlated arms
Multi-player problems: competitive CMAB [3, 4], multi-player CMAB [6]

Reference

[1] Wei Chen, Yajun Wang, Yang Yuan, and Qinshi Wang. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. JMLR, 2016.
[2] Branislav Kveton, Csaba Szepesvari, Zheng Wen, and Azin Ashkan. Cascading bandits: Learning to rank in the cascade model. ICML, 2015.
[3] Jinhang Zuo, Xiaoxi Zhang, and Carlee Joe-Wong. Observe before play: Multi-armed bandit with pre-observations. AAAI, 2020.
[4] Jinhang Zuo, Xutong Liu, Carlee Joe-Wong, John Lui, and Wei Chen. Online competitive influence maximization. AISTATS, 2022.
[5] Runzhe Wan, Lin Ge, and Rui Song. Towards Scalable and Robust Structured Bandits: A Meta-Learning Framework. arXiv preprint, 2022.
[6] Chengshuai Shi, Wei Xiong, Cong Shen, and Jing Yang. Heterogeneous Multi-player Multi-armed Bandits: Closing the Gap and Generalization. NeurIPS, 2021.

Introduction to Robotic Learning

Yafei Hu
CMU ECE & RI

Carnegie
Mellon
University

What are robotic research topics all about?

- **Perception**
  - From camera: object detection/tracking/segmentation
  - From LiDAR pointcloud: 3D object detection/tracking/segmentation

- **Localization and Mapping**
  - Compute the location and rotation (pose) of the robot
  - Build a 3D map

- **Motion Planning**
  - How to design a sequence of actions for a certain task, e.g. go from point A to B

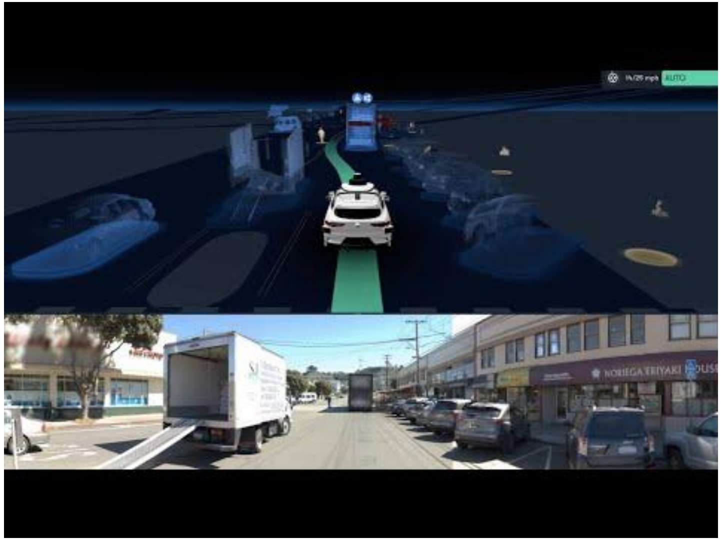
- **Control**
  - How to smoothly execute actions, e.g. stabilize the robotic arms, legs

Looks complex?

Let first take a look at a few vivid examples

Perception: Let the robot know the environment

Often a sensor fusion (from both image and LiDAR) approach is used

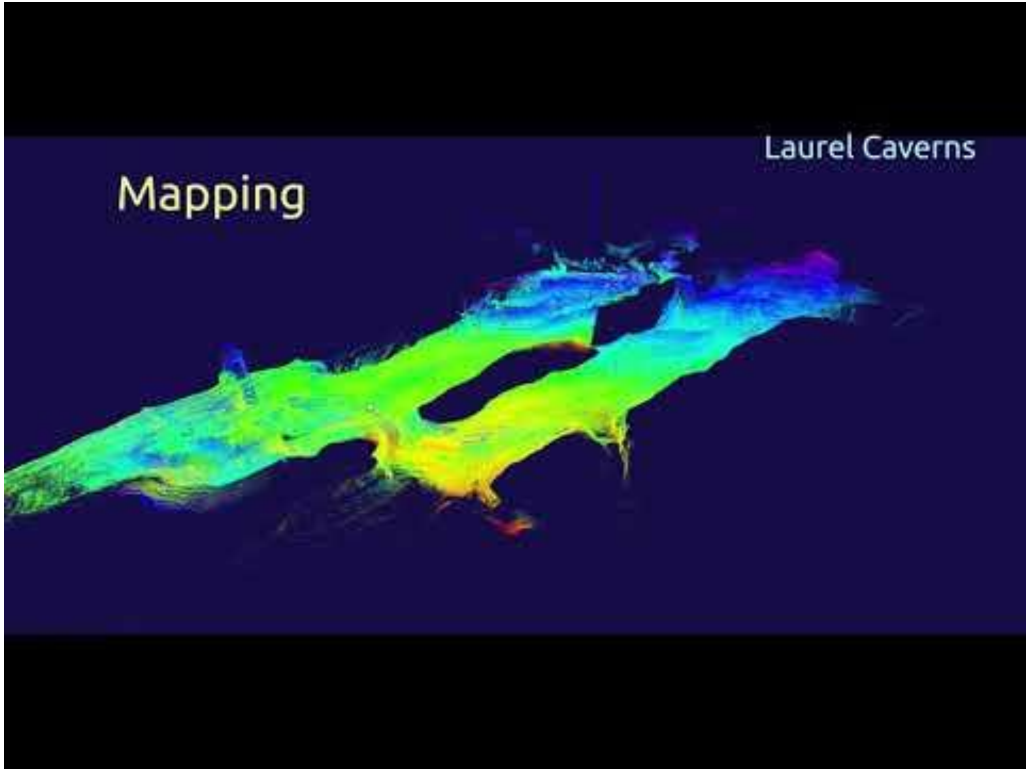


Perception of a autonomous vehicle from Waymo

Robotic perception is now largely a computer vision problem.

Now dominated by deep learning!

Localization and Mapping: Let the robot know where it is



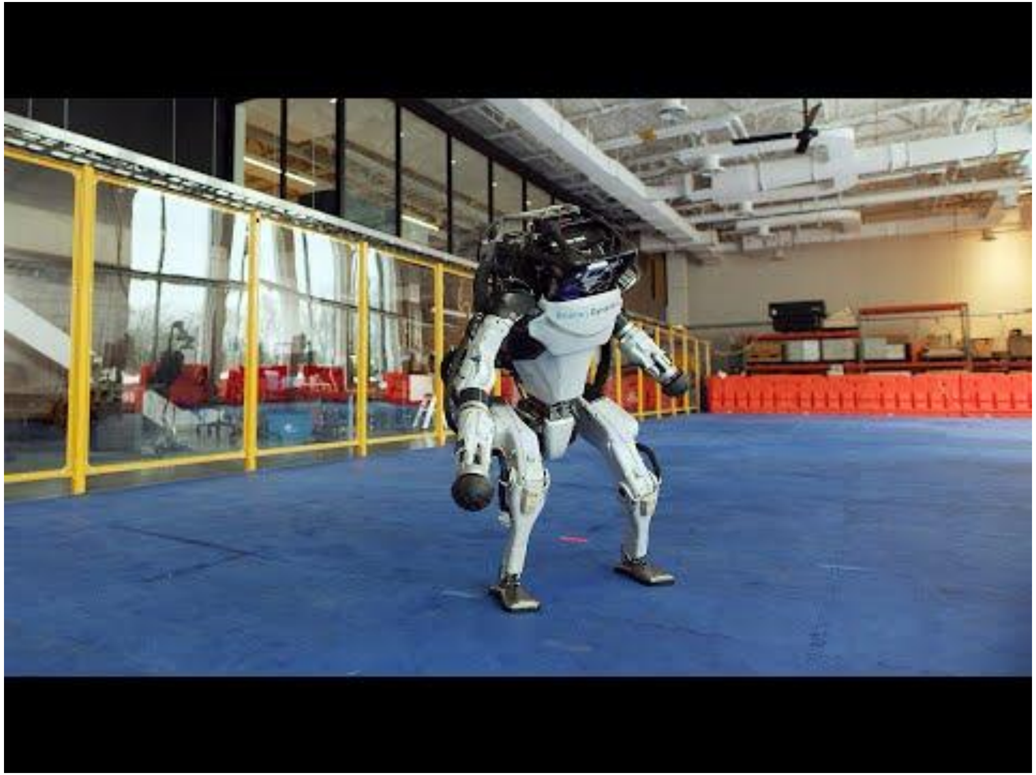
LiDAR localization and mapping for an aerial vehicle by our lab at DAPPA Subterranean challenge

Carnegie Mellon University

Planning: Let the robot know how to finish a task

Robot plan to assemble a birdhouse, from search-based planning lab of CMU RI

(Optimal) Control: Let the robot smoothly execute actions



Boston Dynamics Quadruped robot dog Spot Mini and humanoid robot Atlas

How can machine learning help with these topics?

- **Perception**
    - Nearly everything! Right now deep learning dominate this area.

- **Localization and Mapping**
    - Deep network for pose regression
    - Deep learning based 3D reconstruction, e.g. NeRF

- **Motion Planning and Control**
    - Reinforcement learning (RL) or imitation learning (IL) to learn motion planning policy, robot grasping, legged robot adaptive walking

Few advantages over traditional methods:

- Easy to integrate with vision
- We can learn world transition model (recall the last RL lecture)
- Robot can improve from experience

How can machine learning help with these topics?

- **Perception**
  - Nearly everything! Right now deep learning dominate this area.
- **Localization and Mapping**
  - Deep network for pose regression
  - Deep learning based 3D reconstruction, e.g. NeRF
- **Motion Planning and Control**
  - Reinforcement learning (RL) or imitation learning (IL) to learn motion planning policy, robot grasping, legged robot adaptive walking

Few advantages over traditional methods:

- Easy to integrate with vision
- We can learn world transition model (recall the last RL lecture)
- Robot can improve from experience

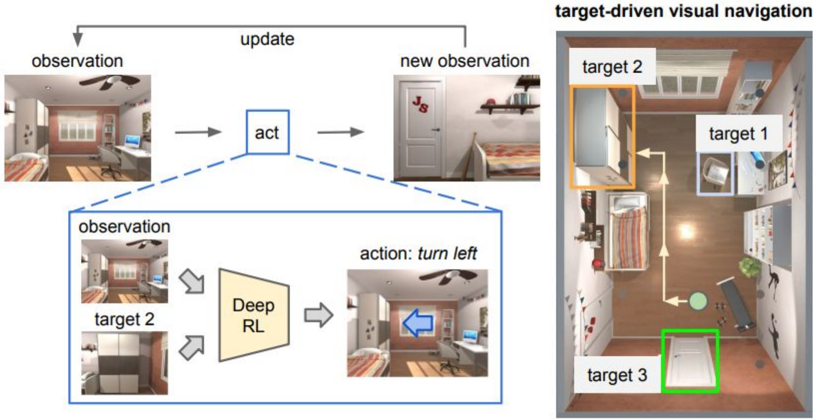
We will mainly focus on learning for planning and control

We will (briefly) introduce a few modern classic works, including:

- Robot Learn to Navigate
  - Yuke Zhu et. al. Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning, ICRA 2017
- Robot Learn to Manipulate Objects
  - Anusha Nagabandi et. al. Deep Dynamics Models for Learning Dexterous Manipulation, CoRL 2019
- Robot Learn to Walk
  - Jemin Hwangbo et. al., Learning Agile and Dynamic Motor Skills for Legged Robots, Science Robotics 2019
  - Xue Bin Peng et. al., Learning Agile Robotic Locomotion Skills by Imitating Animals, RSS 2020 (will not present today, but also fun to read)

Robot Learn to Navigate

System overview:



How to formulate the MDP?

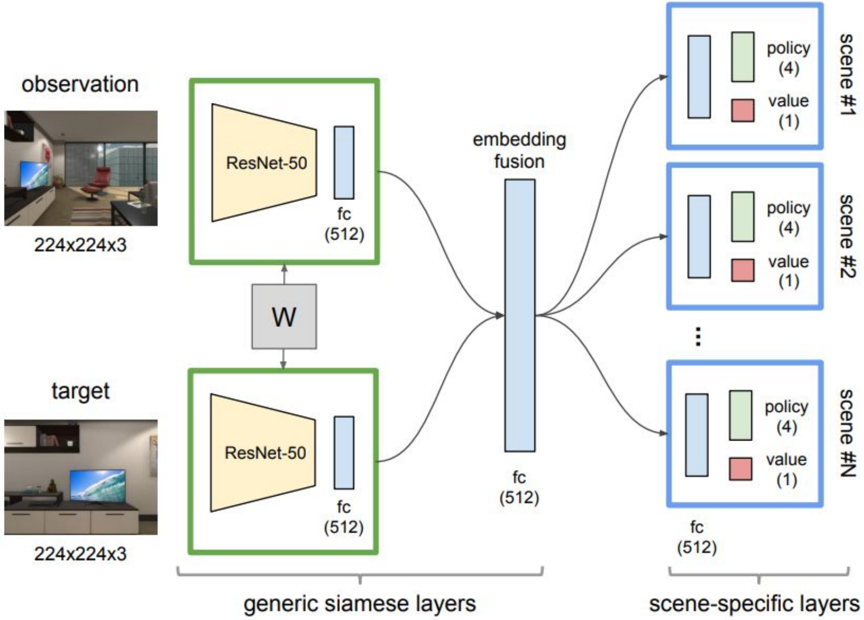
- State: image observations
- Action: moving forward, moving backward, turning left, and turning right
- Reward: 10.0 if goal reached, -0.01 as immediate reward to panelize redundant moves
- State transition model: Unknown

Yuke Zhu et. al. Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning, in ICRA 2017

Robot Learn to Navigate

Policy and Value network:

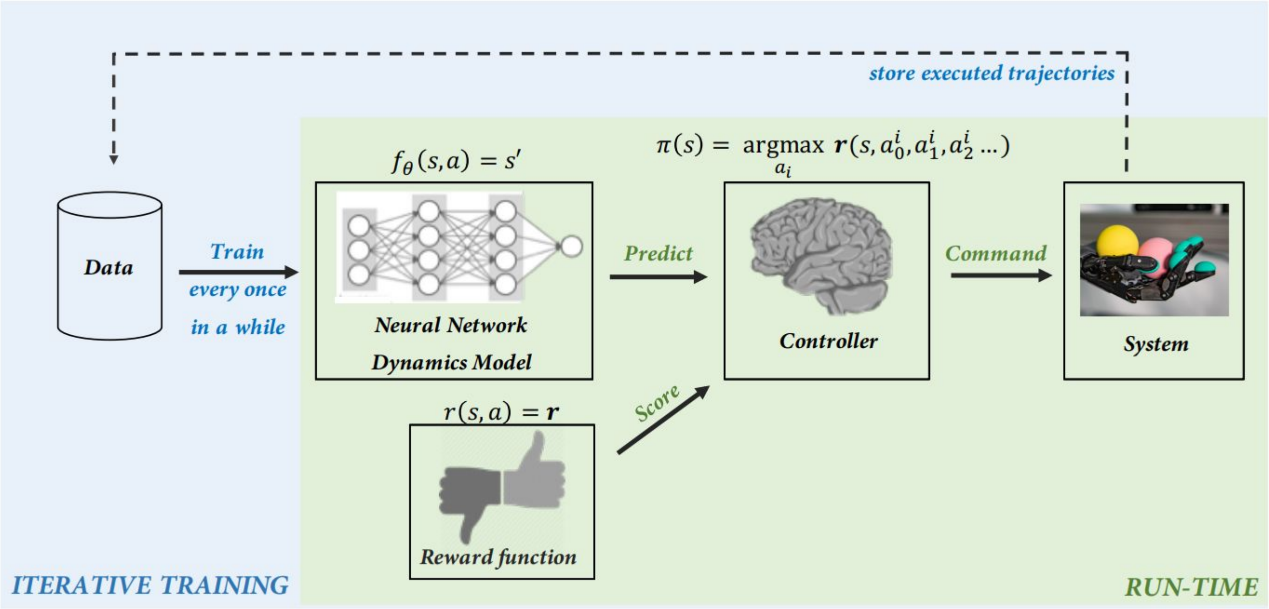
Actor-critic model, meaning it learns both value function and policy



Yuke Zhu et. al. Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning, in ICRA 2017

Robot Learn to Manipulate Objects

System overview:



This work combines model learning and optimal control. It belongs to **model-based reinforcement learning**.

Anusha Nagabandi et. al. Deep Dynamics Models for Learning Dexterous Manipulation, CoRL 2019

Robot Learn to Manipulate Objects

State observations:

Positions of the ball tracked from a RGB camera, positions of the torques of the robot hand

Actions:

position commands to the actuators of the robot hand, 24 DoF

Rewards:

Varies. But in short, the distance from the target

Model representation:

$$\hat{p}_\theta(s'|s,a) = \mathcal{N}(\hat{f}_\theta(s,a), \Sigma)$$

θ denotes model network weights

Policy :

Not from RL, but from model-predictive control

Anusha Nagabandi et. al. Deep Dynamics Models for Learning Dexterous Manipulation, CoRL 2019

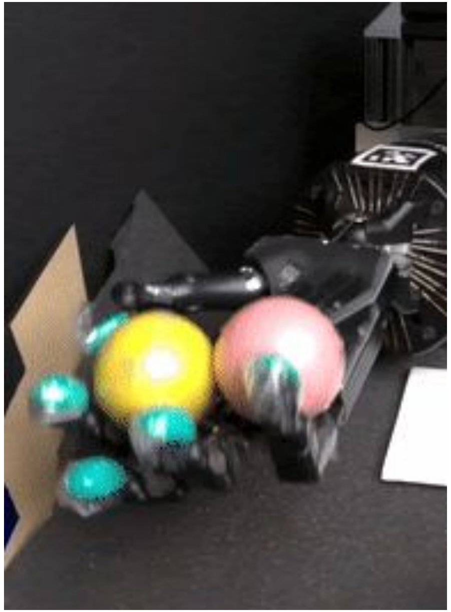
Robot Learn to Manipulate Objects

Some fun results:

0-0.25 hours training

0.25-0.5 hours training

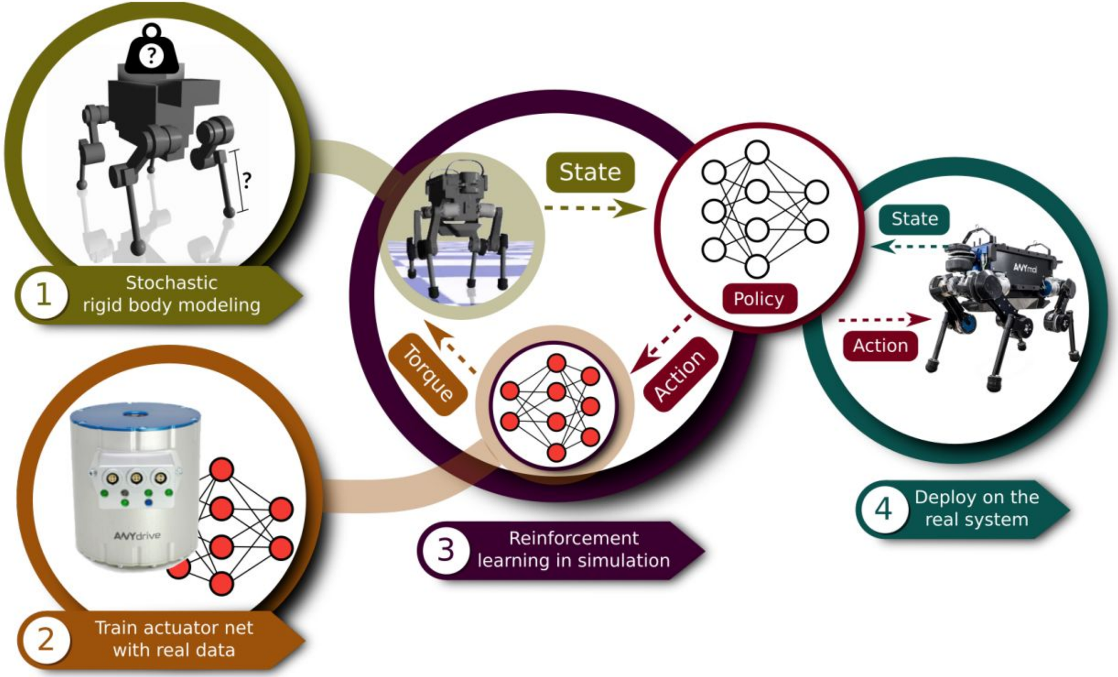
2 hours training



Anusha Nagabandi et. al. Deep Dynamics Models for Learning Dexterous Manipulation, CoRL 2019

Robot Learn to Walk

System overview:



Jemin Hwangbo et. al., Learning Agile and Dynamic Motor Skills for Legged Robots, Science Robotics 2019

Robot Learn to Walk

State observations:

The joint angles, velocities, and body twist

Actions:

position commands to the actuators

Rewards:

Torque, joint speed, acceleration costs, etc.

Policy optimization objectives:

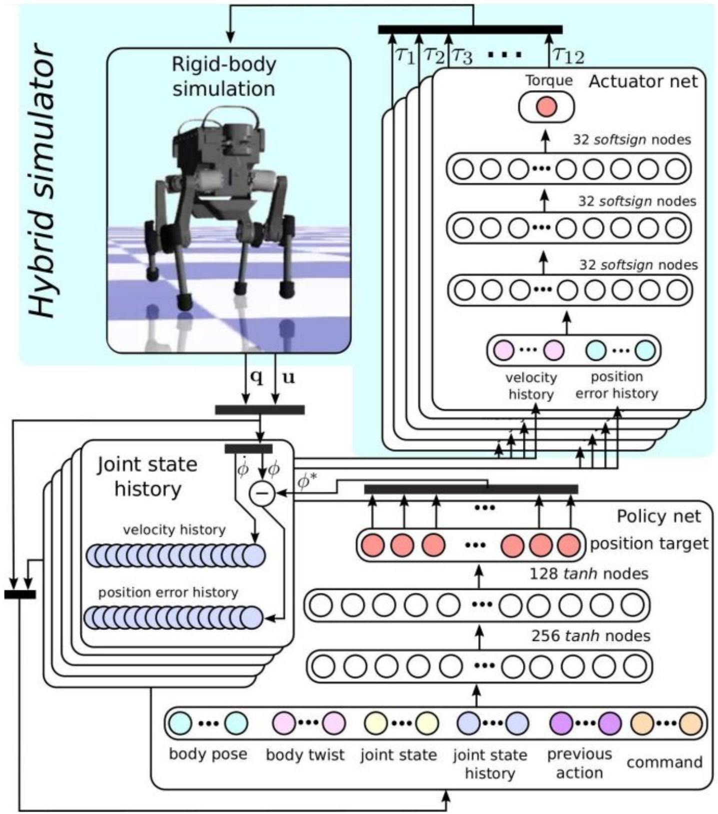
$$\pi^* = \underset{\pi}{\operatorname{argmax}} \mathbb{E}_{\boldsymbol{\tau}(\pi)} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]$$

Reinforcement learning model:

A policy gradient algorithm: Trust Region Policy Optimization (TRPO)

Jemin Hwangbo et. al., Learning Agile and Dynamic Motor Skills for Legged Robots, Science Robotics 2019

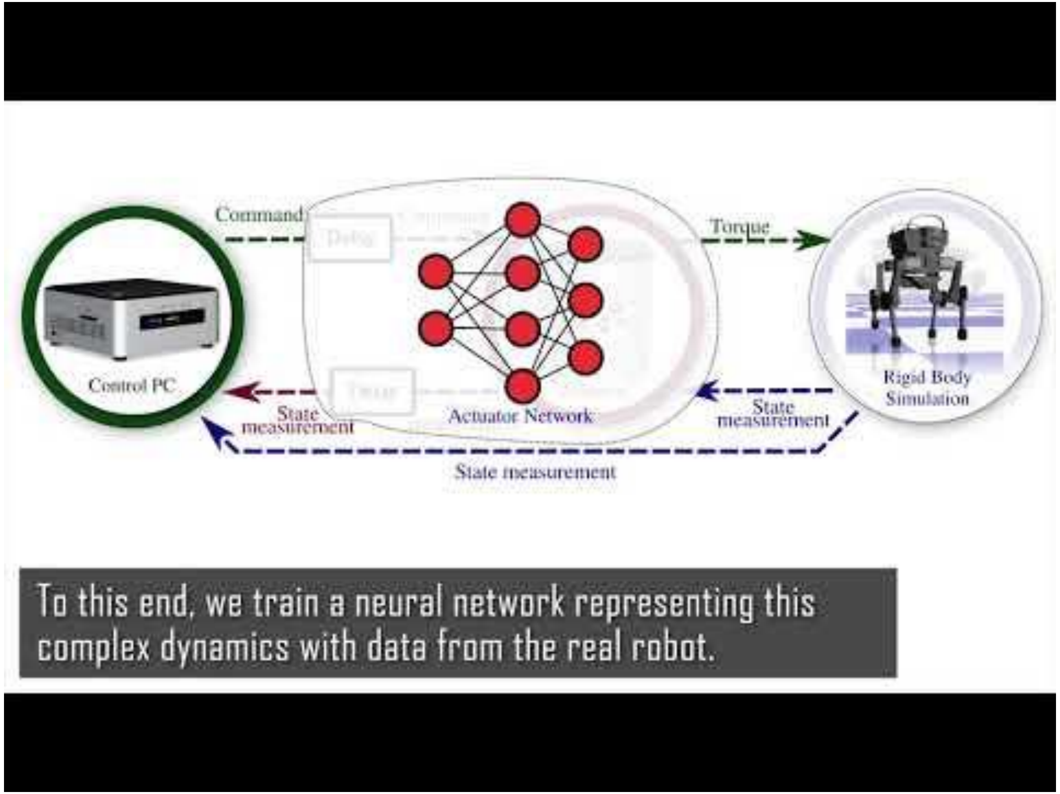
Robot Learn to Walk



Jemin Hwangbo et. al., Learning Agile and Dynamic Motor Skills for Legged Robots, Science Robotics 2019

Robot Learn to Walk

How it works?



To this end, we train a neural network representing this complex dynamics with data from the real robot.

Jemin Hwangbo et. al., Learning Agile and Dynamic Motor Skills for Legged Robots, Science Robotics 2019

Thank you for attending!

All questions and comments are welcome!

Domain adaptation in computer vision

Yu-Jhe Li
CMU ECE

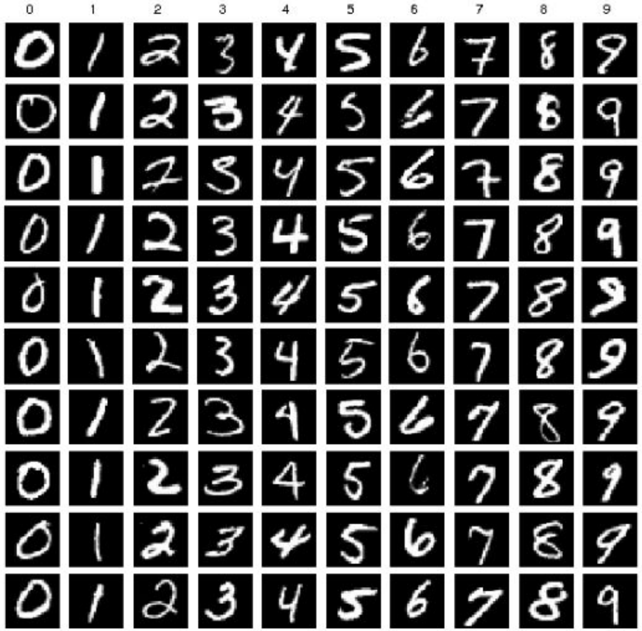
Current topics in Computer Vision

- 2D/3D object recognition
- Action and behavior recognition
- Adversarial learning, adversarial attack and defense methods
- Biometrics, face, gesture, body pose
- Computational photography
- Efficient training and inference methods for networks
- Explainable AI, fairness, accountability, privacy, transparency and ethics in vision
- Image and video retrieval
- Image and video synthesis
- Image classification
- Low-level and physics-based vision
- Machine learning architectures and formulations
- Medical, biological and cell microscopy
- Motion and tracking
- Optimization and learning methods
- Pose estimation
- Representation learning, deep learning
- Scene analysis and understanding
- Transfer, low-shot, semi- and unsupervised learning
- Video analysis and understanding
- Vision + language, vision + other modalities
- Vision applications and systems, vision for robotics and autonomous vehicles

https://cvpr2021.thecvf.com/node/33

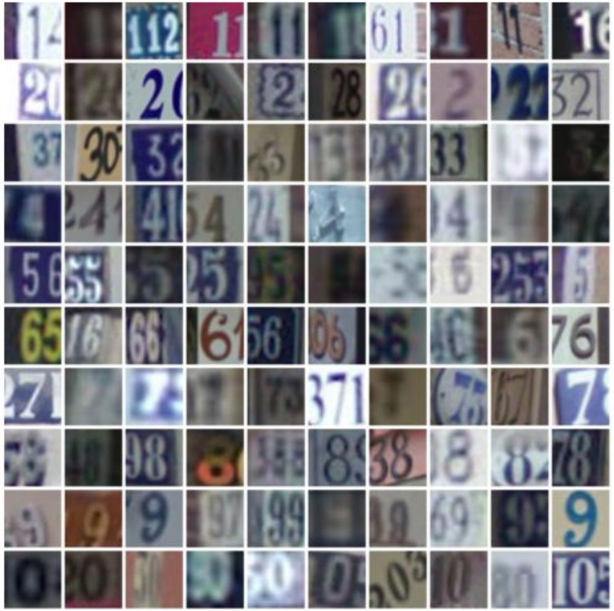
Why domain adaptation

Train:



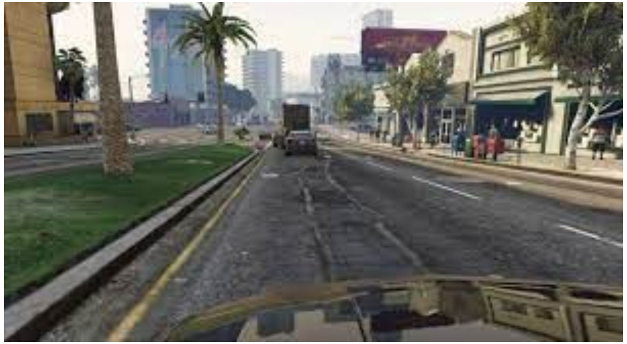
Test:

ACC: 67%

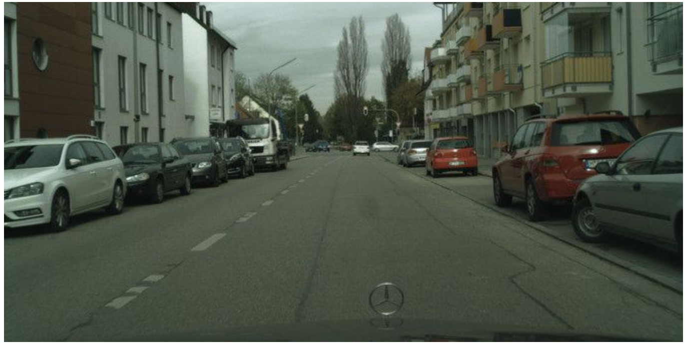


Why domain adaptation?

Simulator (GTA)



Real world (cityscapes)

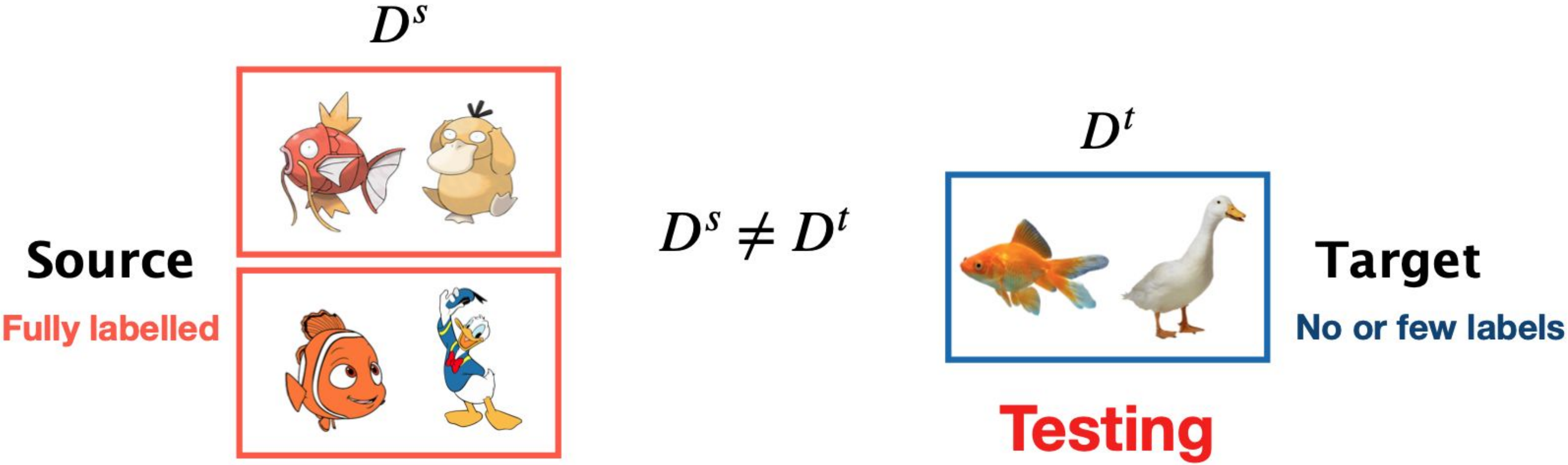


Real photos

Sketches

Domain Adaptation

We do not have labels in target domain for training the models.



D^s

Source

Fully labelled

$D^s \neq D^t$

D^t

Target

No or few labels

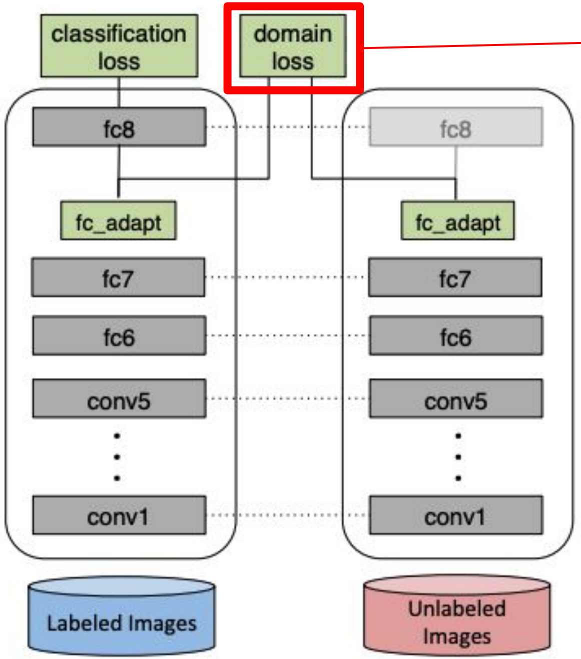
Testing

Approaches

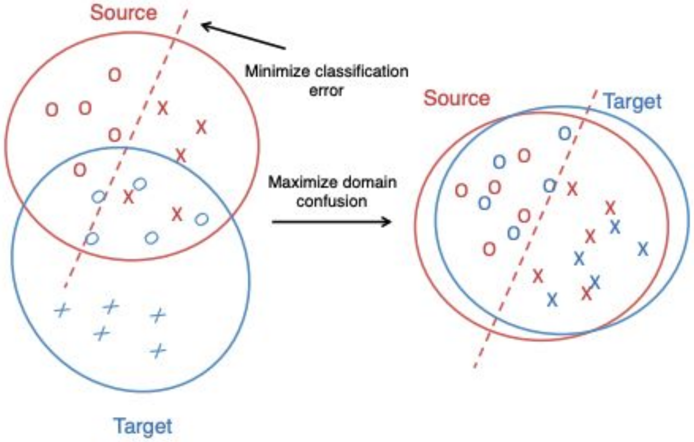
1. Discrepancy-based methods
2. Adversarial-based methods

1. Discrepancy-based method: **Maximum Mean Discrepancy**

Push the mean of two distributions closer



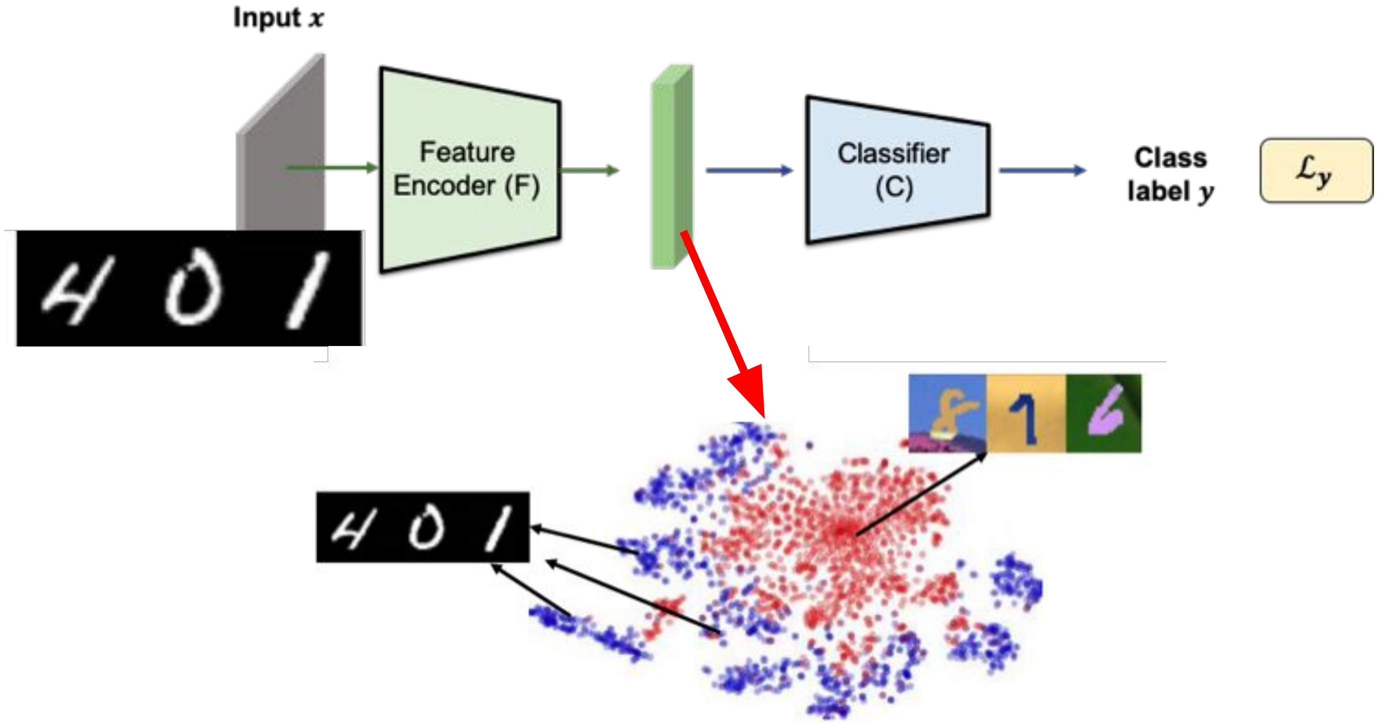
$$\mathrm{MMD}(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \phi(x_t) \right\|$$



Tzeng, Eric, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. "Deep domain confusion: Maximizing for domain invariance." arXiv preprint arXiv:1412.3474 (2014).

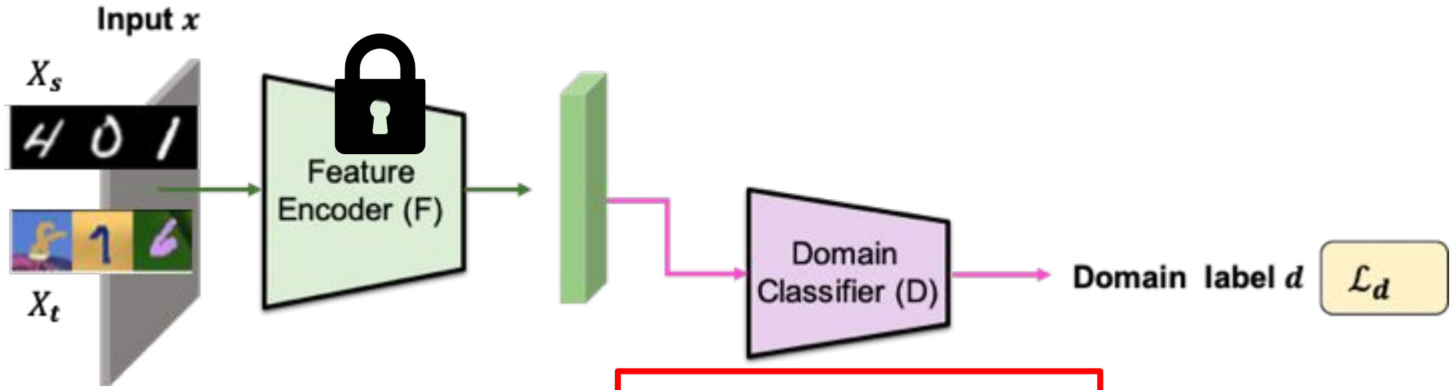
2. Adversarial-based method: Domain Adversarial Learning

Problem with the model train on the source domain:



2. Adversarial-based method: Domain Adversarial Learning

Train a domain classifier

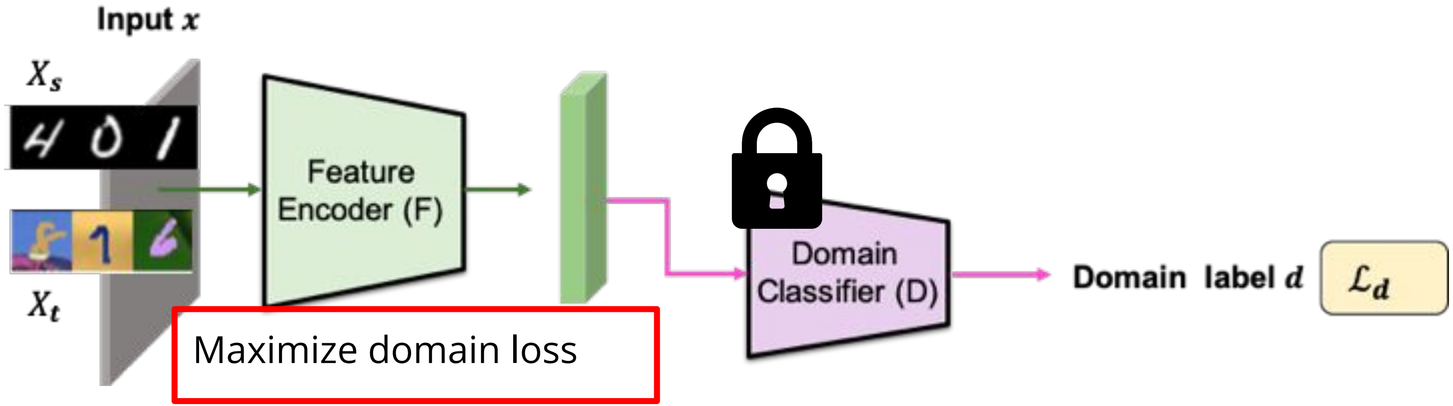


Minimize domain loss

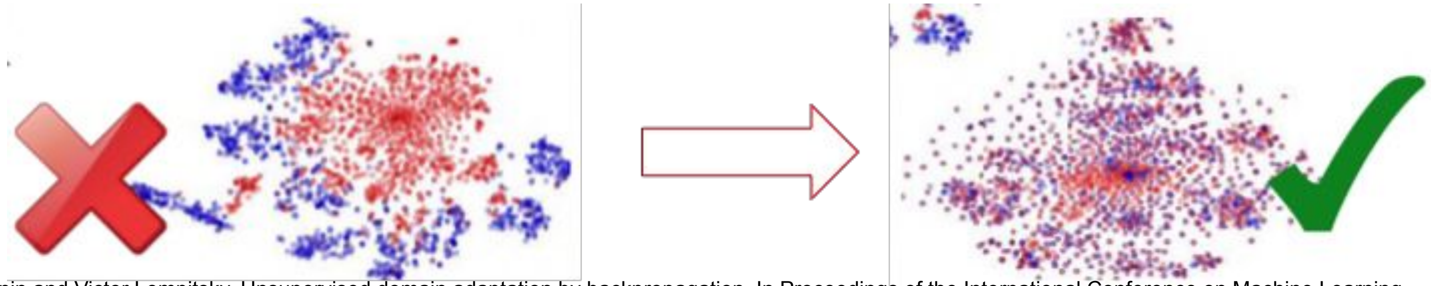
Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In Proceedings of the International Conference on Machine Learning (ICML), 2015.

2. Adversarial-based method: Domain Adversarial Learning

Update the feature encoder to confuse the domain classifier



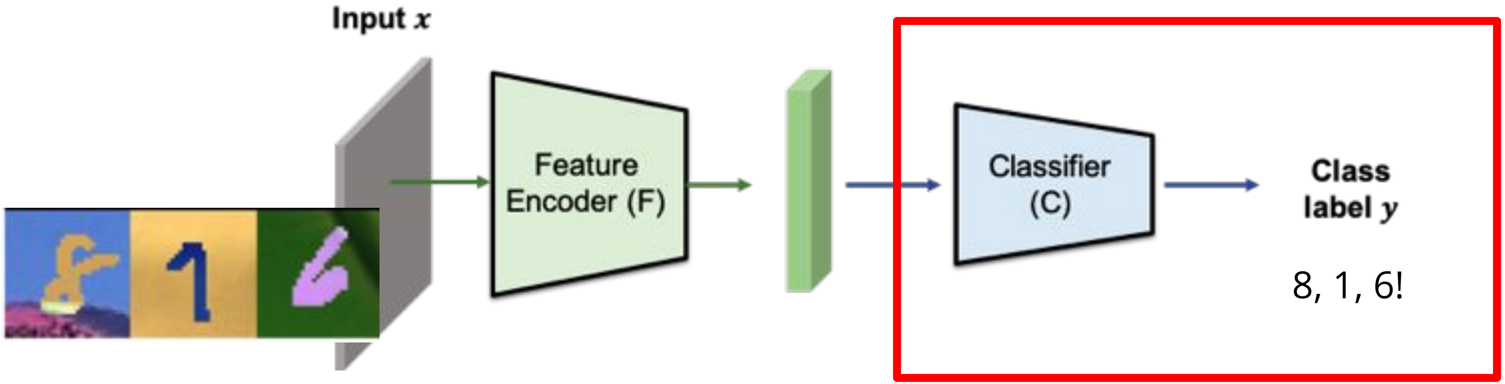
Maximize domain loss



Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In Proceedings of the International Conference on Machine Learning (ICML), 2015.

2. Adversarial-based method: Domain Adversarial Learning

Append the classifier back without training.



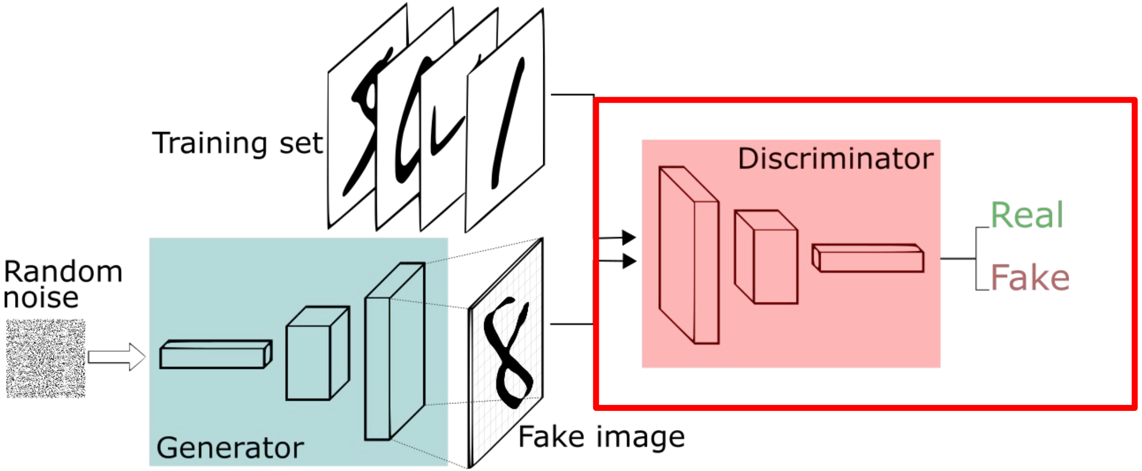
8, 1, 6!

Applications of adversarial learning

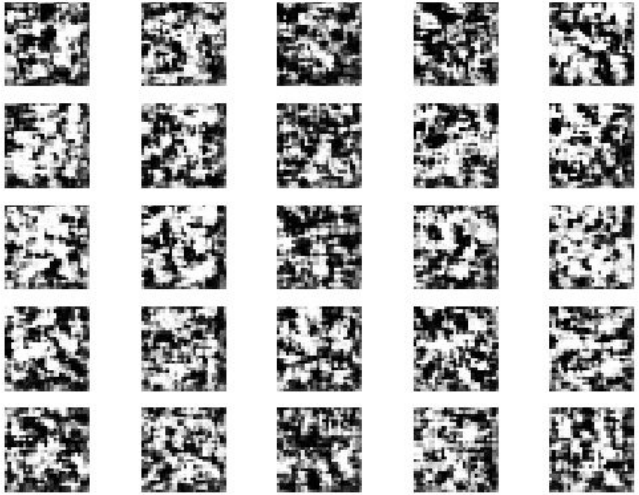
- Generative adversarial network (GAN)
- Style transfer
- Feature disentanglement
- Object detection

Generative adversarial network

Domain adaptation between the distributions from real and fake.



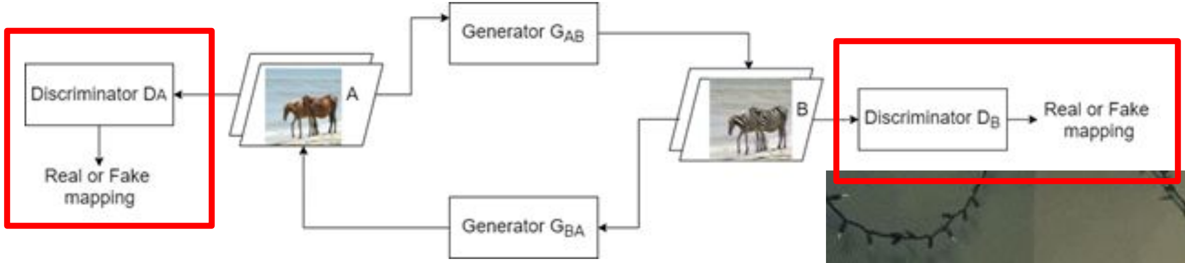
Generated images

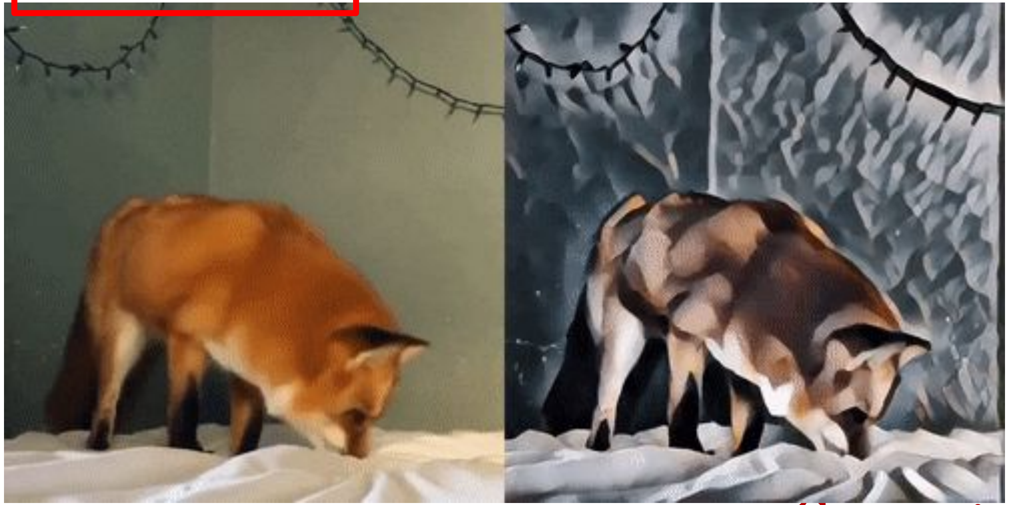


Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems 27 (2014).

Style transfer (real → comic)

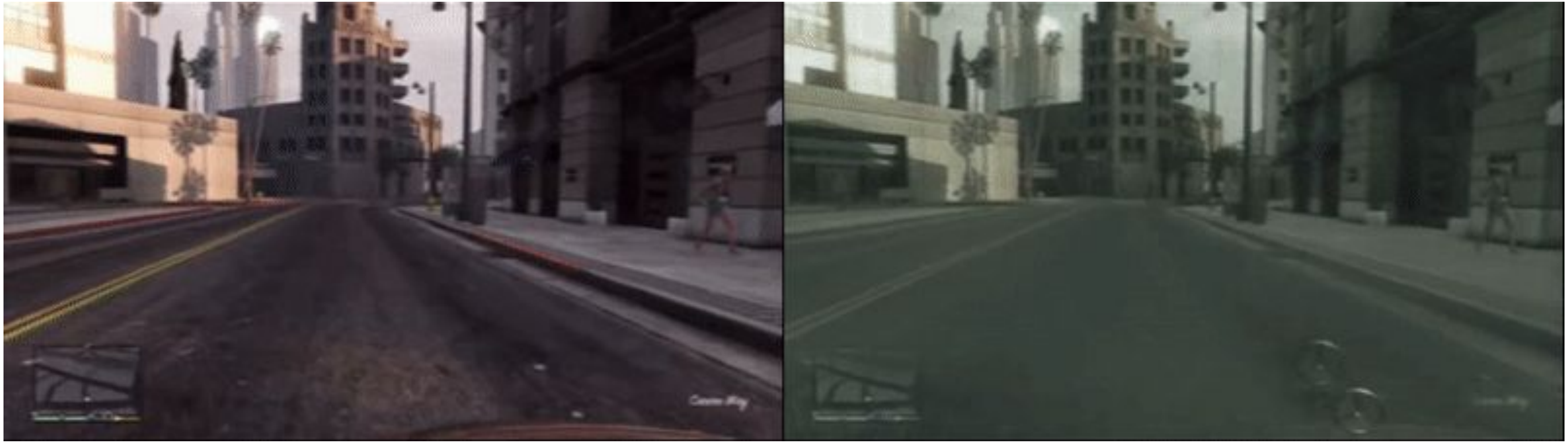
Transfer the style between two domains





Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." In Proceedings of the IEEE international conference on computer vision, pp. 2223-2232. 2017.

Style transfer (game → real)



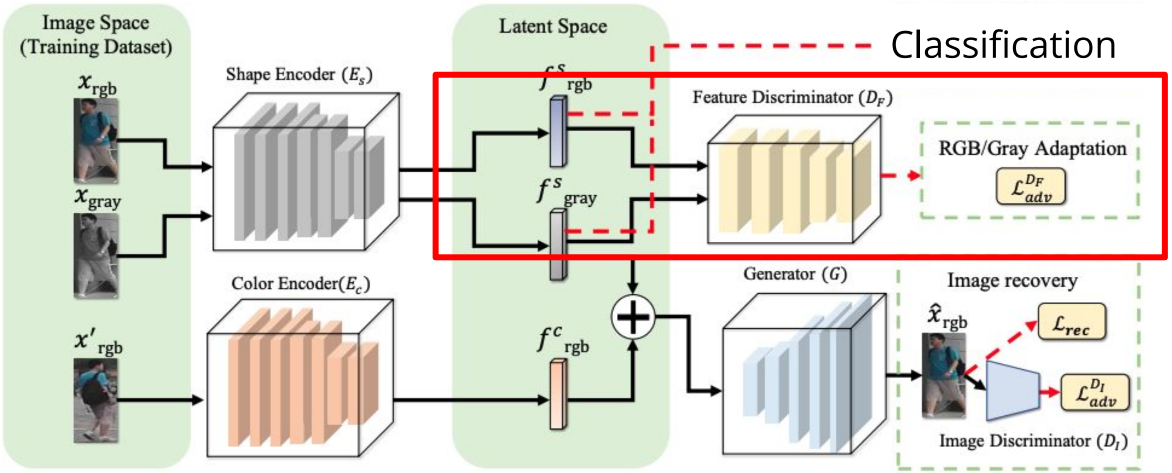
GTA

Real

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." In Proceedings of the IEEE international conference on computer vision, pp. 2223-2232. 2017.

Feature disentanglement

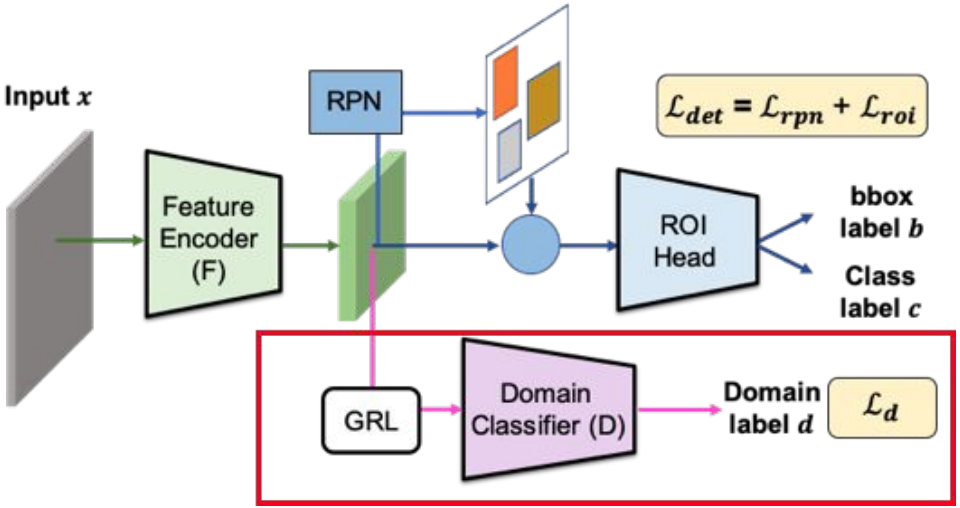
Domain adaptation between human body structures.



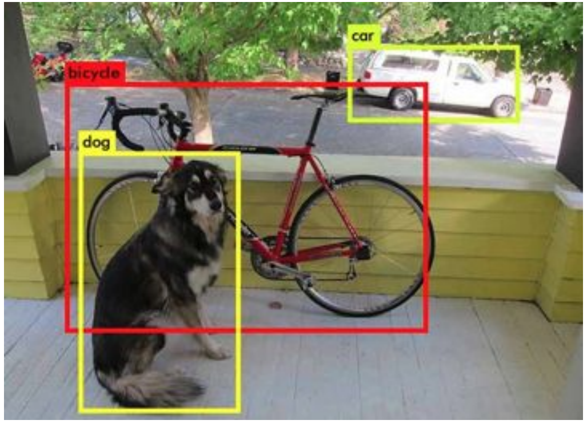
Yu-Jhe Li et al. "Learning shape representations for person re-identification under clothing change." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 2432-2441. 2021

Object detection

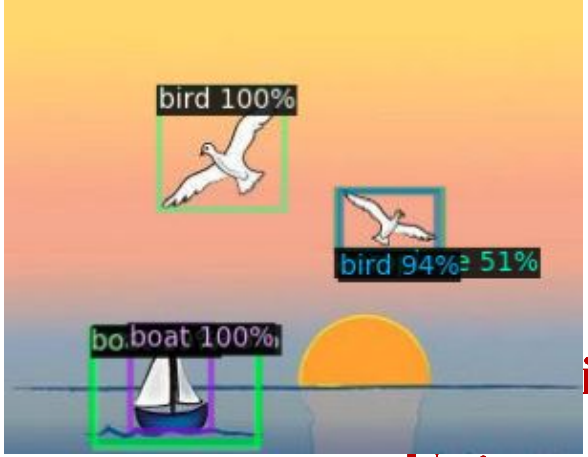
Adding the domain classifier on the feature encoder for object detection!



Source



Target



Yu-Jhe Li et al. "Cross-Domain Adaptive Teacher for Object Detection." Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). 2022.

Q/A

Machine Learning & Hardware

Ching-Yi Lin
CMU ECE

Overview

- Why hardware?
- Customized hardware for machine learning
- Customized machine learning for hardware
- VLSI Symposium 2022: Quantized neural network hardware

Why ML & HW

- ML algorithms have
  - Few kinds of operations: CNN (MUL, ADD, RELU)
  - Repeated multiple times

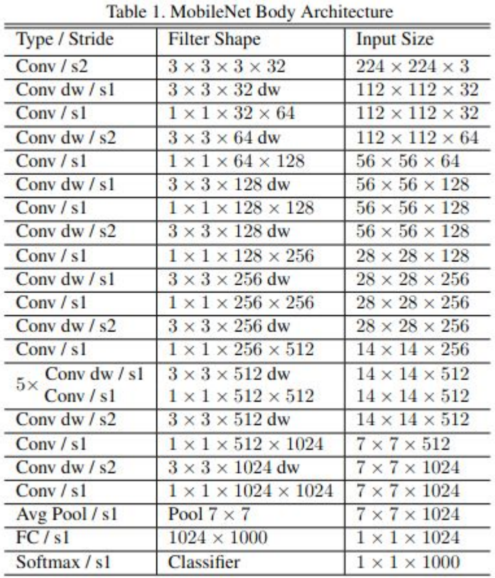


Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1 Conv / s1	$3 \times 3 \times 512$ dw $1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$ $14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

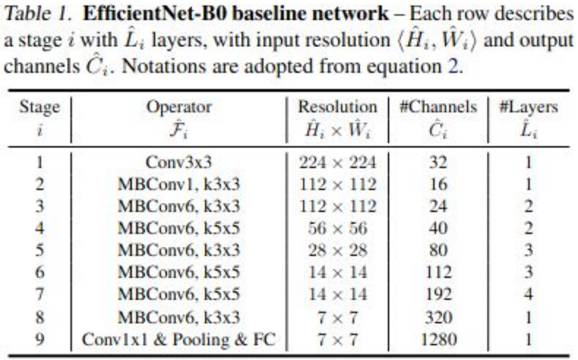


Table 1. **EfficientNet-B0 baseline network** – Each row describes a stage i with $\hat{L}_i$ layers, with input resolution $\langle \hat{H}_i, \hat{W}_i \rangle$ and output channels $\hat{C}_i$. Notations are adopted from equation 2.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

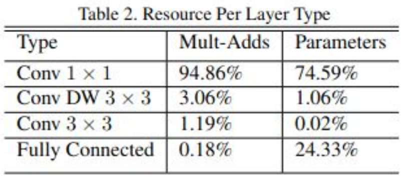


Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

[1] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.
[2] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv:1704.04861* (2017)

Hardware vs. Software: Σwx

- Software uses a generalized template
  - Load -> Compute -> Write
  - Inefficient for repetition
  - 10 cycles
- Hardware speedup
  - Pipelining (6 cycles)
  - Parallelization (5 cycles)

▼ Schedule for software

1	2	3	4	5	6	7	8	9	10
Load (w1)	Load (x1)	MUL	ADD	Write (sum)	Load (w2)	Load (x2)	MUL	Load (sum)	ADD

▼ Schedule for pipelined hardware

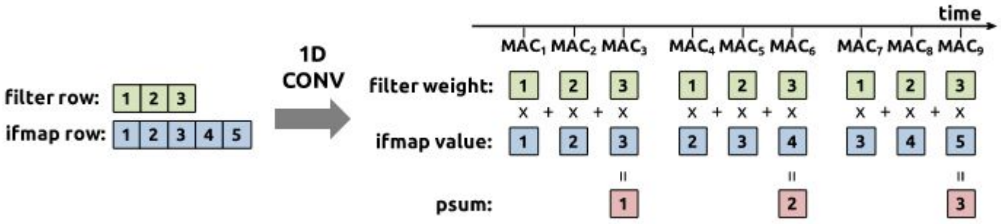
1	2	3	4	5	6
Load (w1)	Load (x1)	MUL	ADD		
		Load (w2)	Load (x2)	MUL	ADD

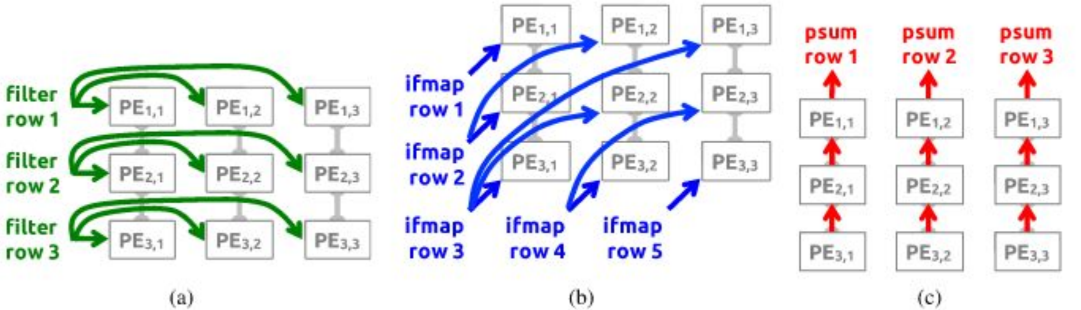
▼ Schedule for parallel + pipelined hardware

1	2	3	4	5
Load (w1)	Load (x1)	MUL	ADD	
	Load (w2)	Load (x2)	MUL	ADD

Customized Hardware for ML

- Convolution
    - Filter row: w
    - Input feature map: x
    - Partial sum
- Systolic array


filter row: 1 2 3
ifmap row: 1 2 3 4 5
1D CONV
time
MAC_1 MAC_2 MAC_3 MAC_4 MAC_5 MAC_6 MAC_7 MAC_8 MAC_9
filter weight: 1 2 3 1 2 3 1 2 3
x + x + x
ifmap value: 1 2 3 2 3 4 3 4 5
=
psum: 1 2 3



(a)

(b)

(c)

Chen, Yu-Hsin, et al. "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks." *IEEE journal of solid-state circuits* 52.1 (2016): 127-138.

Customized Hardware for ML

- Systolic array for commercial
- Tensor Processing Unit (TPU)
  - 65,536 8-bit MAC matrix multiply
  - 92 TeraOps/second
  - 15-30x faster than GPU

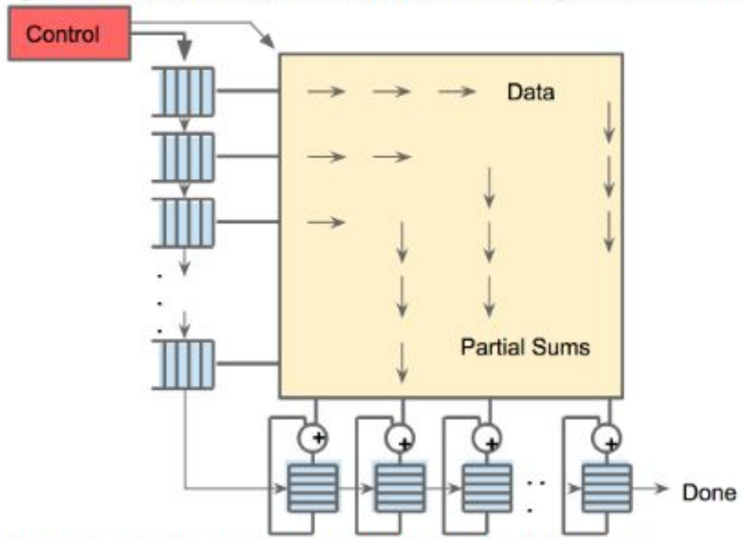


Figure 4. Systolic data flow of the Matrix Multiply Unit.

Jouppi, Norman P., et al. "In-datacenter performance analysis of a tensor processing unit." *Proceedings of the 44th annual international symposium on computer architecture*. 2017.

Customized ML for hardware

$$1.2345 = \underbrace{12345}_{\text{significand}} \times \underbrace{10}_{\text{base}}^{\overbrace{-4}^{\text{exponent}}} .$$

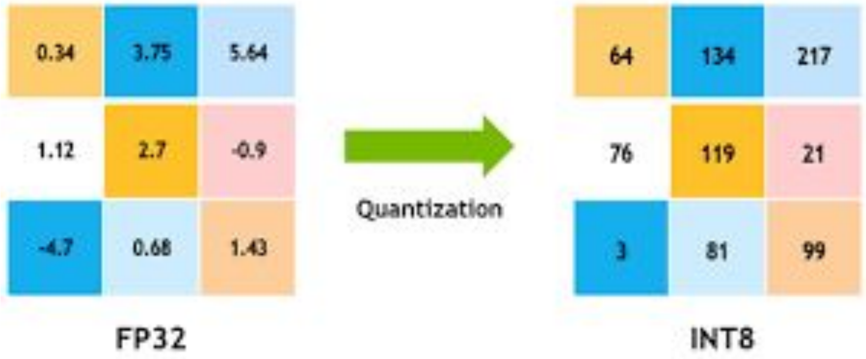
- All you learned are mostly in floating point
  - Arithmetic is expensive
- Quantization is easy? – Back propagation is hard
  - Floating point: 15.1 - 0.2 - = 14.9
  - Fixed point: 15 - 0 = 15(?)

- E.g., 10.375 + 6.34375 = 16.71875

$$\begin{array}{r} 1.0100110 \times 2^3 \\ + 1.1001011 \times 2^2 \\ \hdashline \end{array}$$

$$\boxed{\begin{array}{rr} & 1.0100110 \times 2^3 \\ + & 0.1100110 \times 2^3 \\ \hline & 10.0001100 \times 2^3 \end{array}}$$

$$\boxed{16.75}$$



0.34	3.75	5.64
1.12	2.7	-0.9
-4.7	0.68	1.43

FP32

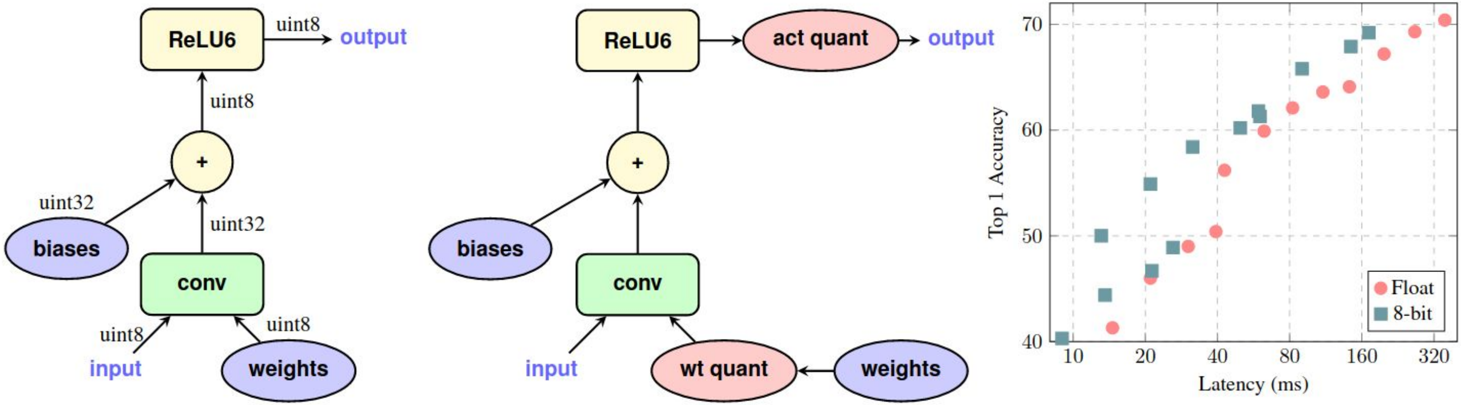
Quantization

64	134	217
76	119	21
3	81	99

INT8

Customized ML for hardware

- Quantization is easy? – Back propagation is hard
- Quantized Neural Network - 8-bit version
  - 8-bit processors are common

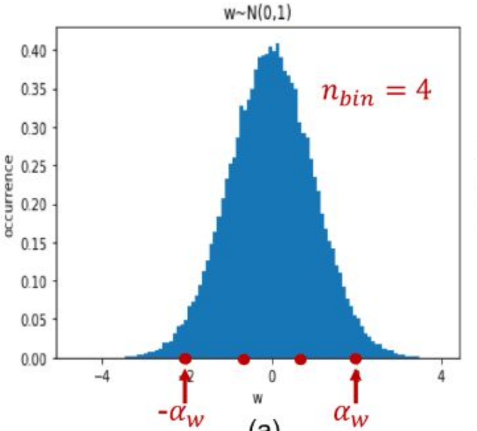


(a) Integer-arithmetic-only inference (b) Training with simulated quantization (c) ImageNet latency-vs-accuracy tradeoff

Jacob, Benoit, et al. "Quantization and training of neural networks for efficient integer-arithmetic-only inference." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Customized ML for hardware

- Quantized Neural Network - 8-bit version
- Quantized Neural Network - 2-bit version
  - Weight: 2-bit



(a)

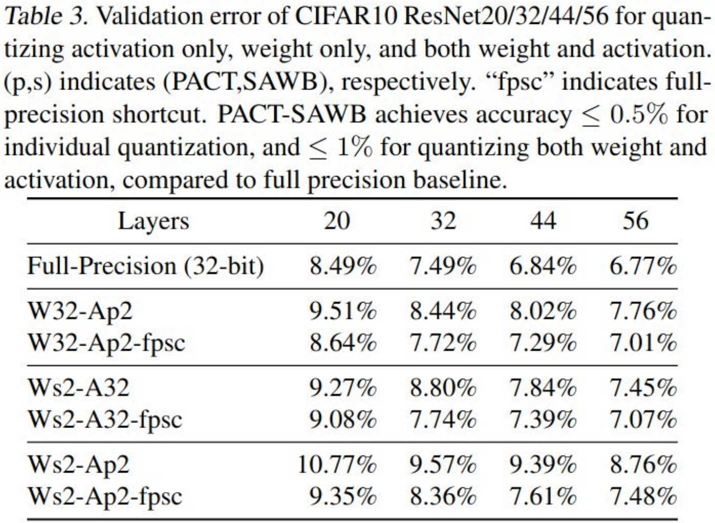


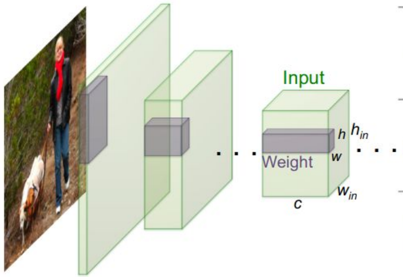
Table 3. Validation error of CIFAR10 ResNet20/32/44/56 for quantizing activation only, weight only, and both weight and activation. (p,s) indicates (PACT,SAWB), respectively. "fpsc" indicates full-precision shortcut. PACT-SAWB achieves accuracy $\leq 0.5\%$ for individual quantization, and $\leq 1\%$ for quantizing both weight and activation, compared to full precision baseline.

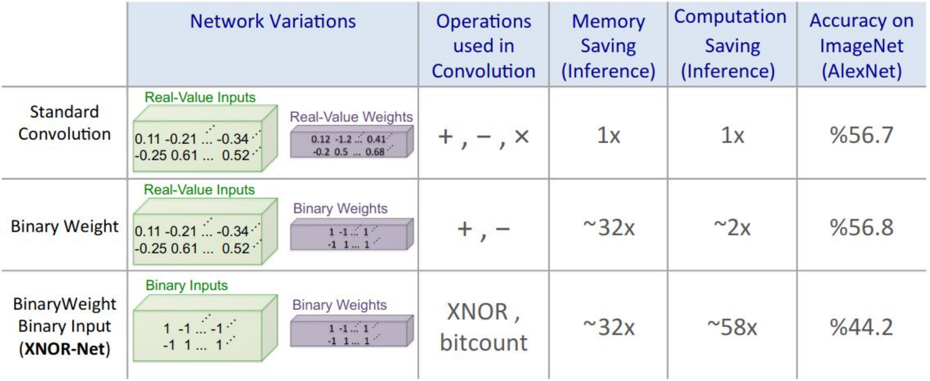
Layers	20	32	44	56
Full-Precision (32-bit)	8.49%	7.49%	6.84%	6.77%
W32-Ap2	9.51%	8.44%	8.02%	7.76%
W32-Ap2-fpsc	8.64%	7.72%	7.29%	7.01%
Ws2-A32	9.27%	8.80%	7.84%	7.45%
Ws2-A32-fpsc	9.08%	7.74%	7.39%	7.07%
Ws2-Ap2	10.77%	9.57%	9.39%	8.76%
Ws2-Ap2-fpsc	9.35%	8.36%	7.61%	7.48%

[1] Choi, Jungwook, et al. "Accurate and efficient 2-bit quantized neural networks." *Proceedings of Machine Learning and Systems* 1 (2019): 348-359.

Customized ML for hardware

- Quantized Neural Network - 8-bit version
- Quantized Neural Network - 2-bit version
- Quantized Neural Network - 1-bit version
    - Input: 1-bit
    - Weight: 1-bit





Network Variations			Operations used in Convolution	Memory Saving (Inference)	Computation Saving (Inference)	Accuracy on ImageNet (AlexNet)
Standard Convolution	Real-Value Inputs 0.11 -0.21 ... -0.34 -0.25 0.61 ... 0.52	Real-Value Weights 0.12 -1.2 ... 0.41 -0.2 0.5 ... 0.68	+ , − , ×	1x	1x	%56.7
Binary Weight	Real-Value Inputs 0.11 -0.21 ... -0.34 -0.25 0.61 ... 0.52	Binary Weights 1 -1 ... 1 -1 1 ... 1	+ , −	~32x	~2x	%56.8
BinaryWeight Binary Input (**XNOR-Net**)	Binary Inputs 1 -1 ... -1 -1 1 ... 1	Binary Weights 1 -1 ... 1 -1 1 ... 1	XNOR , bitcount	~32x	~58x	%44.2

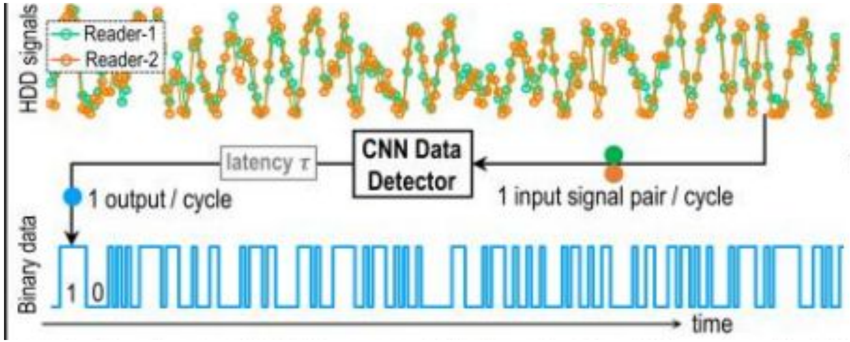
Rastegari, Mohammad, et al. "Xnor-net: Imagenet classification using binary convolutional neural networks." *European conference on computer vision*. Springer, Cham, 2016.

Overview

- Why hardware?
- Customized hardware for machine learning
- Customized machine learning for hardware
- **VLSI Symposium 2022**
  - High-throughput HDD detector
  - Quantized neural network (QNN) algorithm
  - Customized hardware for QNN

Our work – High Throughput HDD Detector

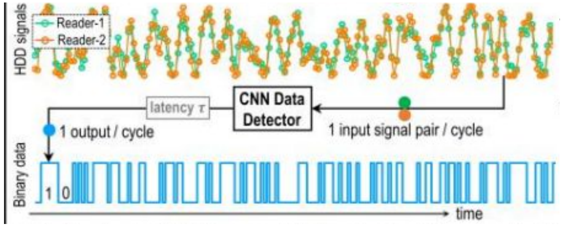
- Objective: Detect hard drive data
  - High throughput: 200Mbit/s - 2Gbit/s
- Customized ML: Quantized MobileNet Architecture
  - 6-bit input
  - 8-bit weight
  - 18-bit partial sum
  - 6-bit activation

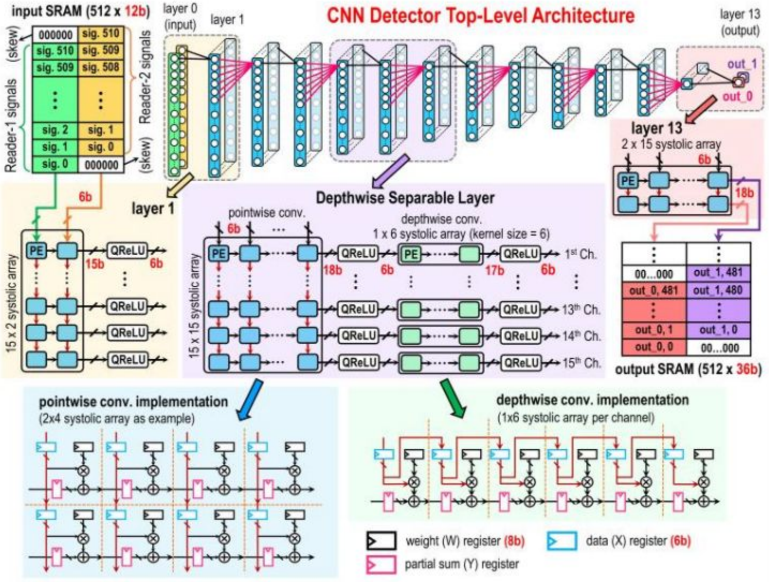


Yuwei Qin, Ruben Purdy Alec Probst, Ching-Yi Lin, Jian-Gang (Jimmy) Zhu, Non-linear CNN-based Read Channel for Hard Disk Drive with 30% Error Rate Reduction and Sequential 200Mbits/second Throughput in 28nm CMOS, 2022 IEEE Symposium on VLSI Technology and Circuits

Our work – High Throughput Detector

- Customized ML
- Customized hardware
    - Low-bit systolic array
    - Flatten CNN layers
- Real tapeout in TSMC 28nm





Yuwei Qin, Ruben Purdy Alec Probst, Ching-Yi Lin, Jian-Gang (Jimmy) Zhu, Non-linear CNN-based Read Channel for Hard Disk Drive with 30% Error Rate Reduction and Sequential 200Mbits/second Throughput in 28nm CMOS, 2022 IEEE Symposium on VLSI Technology and Circuits

Questions?

- Why hardware?
- Customized hardware for machine learning
- Customized machine learning for hardware
- VLSI Symposium 2022: Quantized neural network hardware

Human Physical and Psychological Behavior Economics - The ML Part

Sundar Anand
CMU ECE

Overview

- What is Human Behavior Economics?
- Real Life Examples
  - Physical Behavior
  - Psychological Behavior
- Privacy and Security
- Take Backs

What is Human Behavior Economics?

- We are in a generation where AI is ubiquitous. In recent times, researchers have started exploring different applications for intelligence enabled agents rather than just making them smarter and smarter.
- This gave rise to a new world where humans just don't use AI but they start to live alongside AI.
- Due to this many domains underwent a significant transformation and one of those is Human Behavior Economics.
- In Human Behavior Analysis, agents train the human rather than humans training agents. 😲

How and Why should AI train humans???????

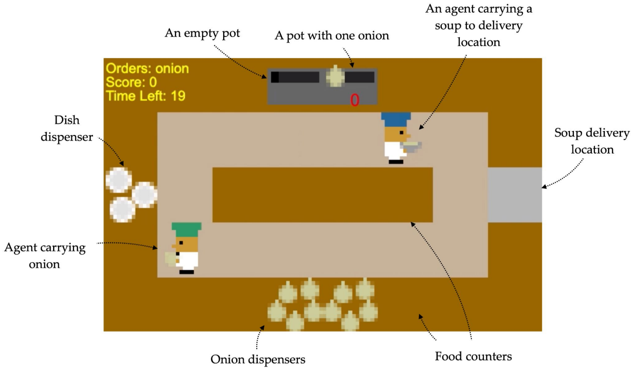
Real Life Examples

Co-bots are designed to work collaboratively with humans to achieve a common goal using Human data.

Still confusing right?

Let's take a few examples

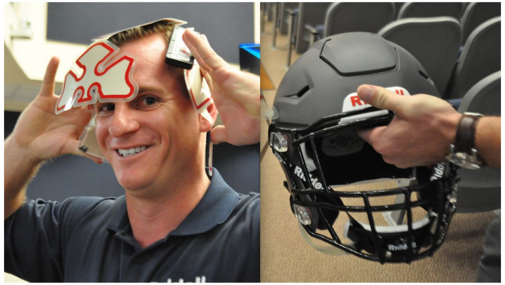
1. Sports Technology and Analysis (Physical)
2. Gaming (Psychological)

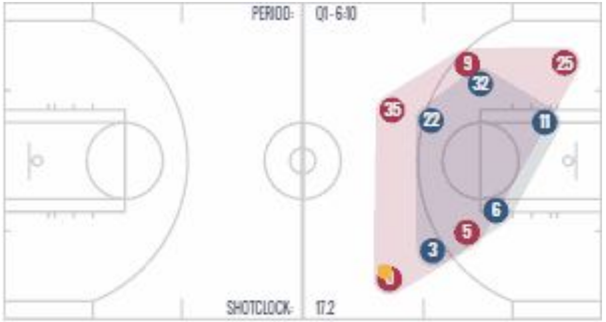


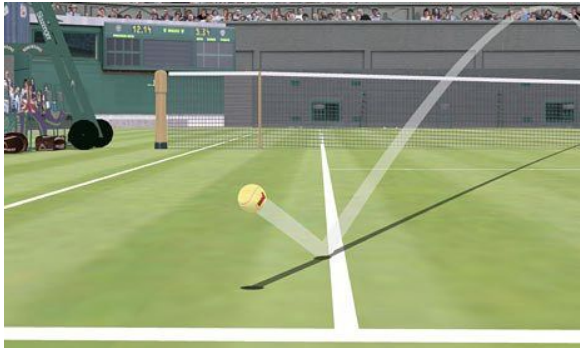
Sports Technology

Machine Learning can be applied to different domains inside Sports

- Player Safety (Smart Helmets)
- Refereeing (Hawk-Eye)
- **Player and Team Performance (Player Tracker)**
- Fan Experience (Win Prediction)





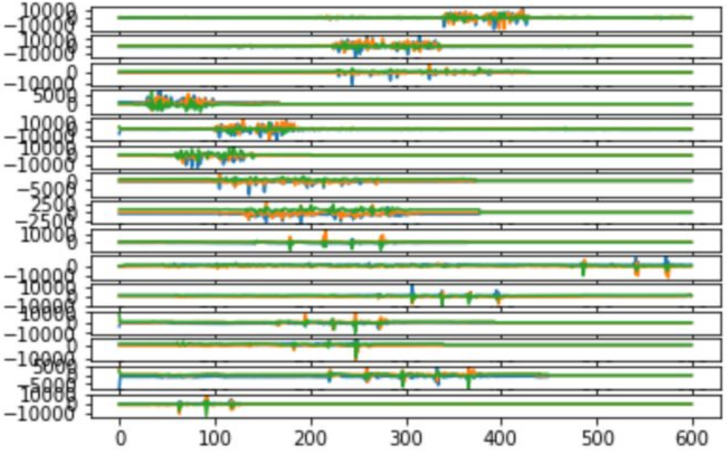


Player and Team Performance

We mentioned we have AI all around us. What if those AI around you is focused on you?

Let me introduce you to one of my research works,
=> Ice-Hockey Player improvements using Computer Vision - Pittsburgh Penguins

- Cameras and sensors all around the rink and players
- Identify the player and activity and hence the player's activity data pattern
- Compare the activity pattern between a professional Player and a U-19 player (Convey Suggestions to Coach)



<u>Use cases:</u>

1. Identify similar playing pattern players (Potential Subs)
2. Better Customised team Strategy
3. Player pattern metric for comparison
4. Identify suitable players for lease or auction

Gaming

Now the psychological part ✨

Some simple yet complicated questions

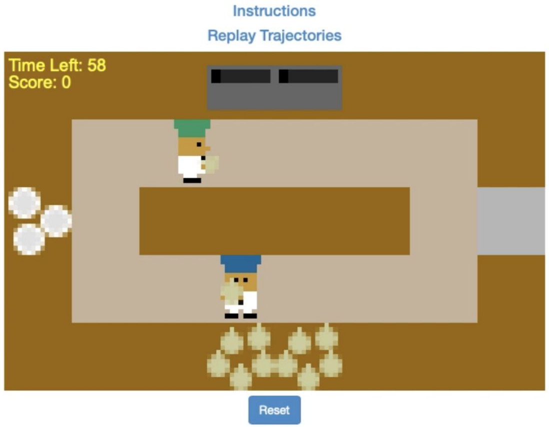
1. What do you think about the Human Computer Relationship?
2. If you are playing a game, will you choose a computer over a human as your teammate?
3. Oh wait, have you ever thought being a sidekick to an agent?
4. Have you ever thought agents can order you?

If you are conventional about these questions, how do you think Robots can perform intricate human surgeries in the future?

Gaming

We tried playing with this psychological (Trust towards AI) part of human brain using a game called Overcooked.

It is a cooking game as simple as it is. But there are different ways to play the game (Scope of Strategy).



We trained three types of agents to play this game

1. Self Play - Can play the game independently
2. Human Aware - Can understand humans strategy
3. Population Based - Has its own strategy

We got many people to play this game with our agents. As expected, they all chose the Human Aware agent and played their own strategy.

When the same set of people were asked to play the same game following the strategy of the Population based, they were able to land in a better score.

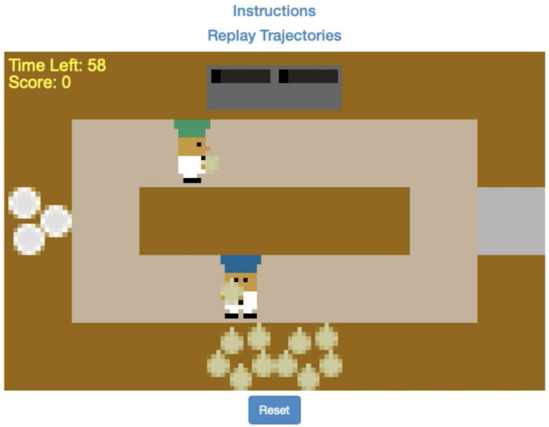
Gaming

Simple logic…

Agents can play the same game billions of times overnight and become experts. Humans on the other hand are extremely fast at grasping the agent's strategy.
=> Perfect Formula

But why is it very hard, answer -> Lack of trust

We are now building some advanced AI methods to analyse all these game logs and to find a pattern in the human play when they have their own strategy and when they follow an Agent's strategy.



We hope this analysis will lead us to bridge the trust gap and hence make the future fantasy a reality! :D

Privacy and Security Breach

Here comes the question that everyone asks today…

Let’s take physical behavior first

What all can be potential stop signs

You don’t want

1. The opponent team to know how your player plays
2. Your player to become all conscious about tracking
3. These metrics to affect the player’s contract
4. The nature of the game to be changed

Well sports was an example. What other things are there

You don’t want

1. Your navigation system to track your every movement.
2. Apple watch to know when you go out for walk and when you come back home

Now the Psychological part

You don’t want

1. Your thought and actions to be predictable
2. Opponent to predict your strategy
3. Your decision making thought process to be exposed

Outside games

You don’t want

1. Companies to know what you gonna shop next
2. Online movie streaming to know what type of person you are
3. Your phones to know what words you gonna type in next

Carnegie Mellon University

Take Backs

In this course you would have learnt a lot about ML, right from basic mathematics to Neural Networks and we have tonnes and tonnes of data lying all around us.

As a Machine Learning Engineering student, it is our duty to make the maximum out of it for tech and mankind evolution without disturbing anyone's privacy and security.

In today's modern world, with everyone's house equipped with IoT devices, it is easy to predict one's life of pattern (ie when he unlocks his door, when he turn on the lights and fans, when he opens his bedroom, when he leaves the house, etc) - But should we do it....

So be it in your future projects or products, make sure to be extra careful when you play with Human Behavioral data!!

OPEN FOR QUESTIONS NOW