Cognitive EW and Reinforcement Learning

- **Kyle Davidson, PhD, CD**
  - Kyle.davidson@agile-em.com

- **September 8th, 2022**
- **14:00 EST**

Outline

- Introduction
- Basics of Machine Learning
- Reinforcement Learning
- Cognitive EW Systems
- Conclusion

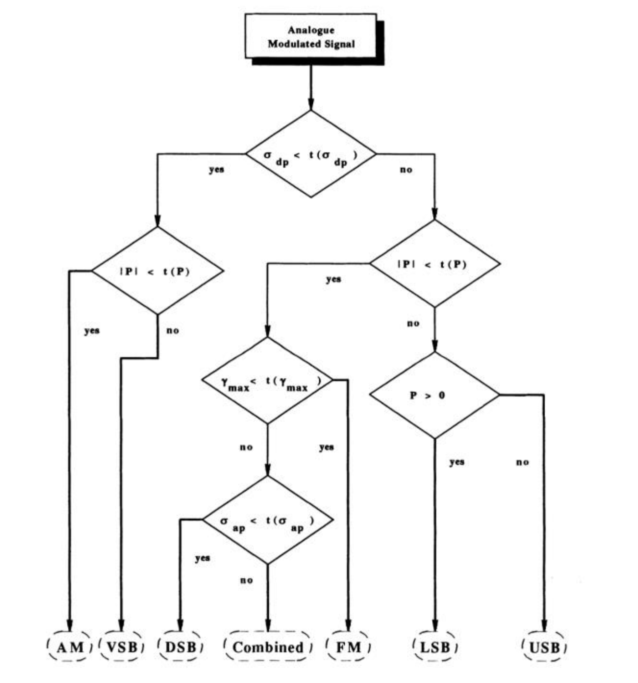
Introduction to Machine Learning

Cognitive EW and Reinforcement Learning

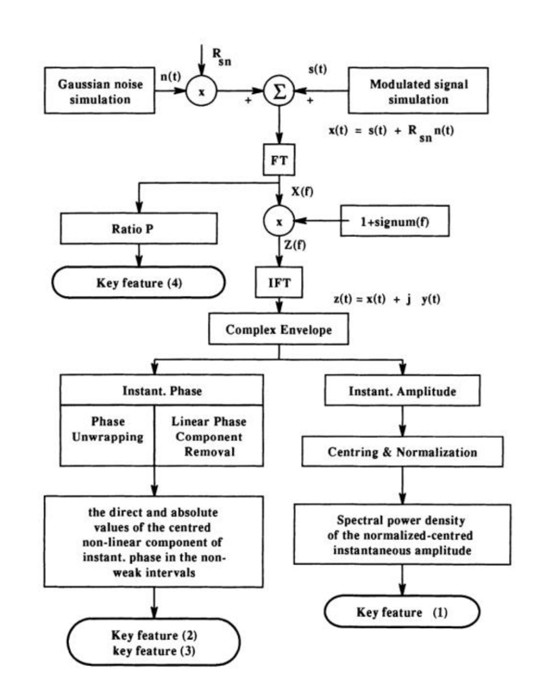
ASSOCIATION OF OLD CROWS

[1] E.E. Azzouz, and A.K. Nandi, Automatic Modulation Recognition of Communications Signals, Springer, 1996

Traditional Signal Classification

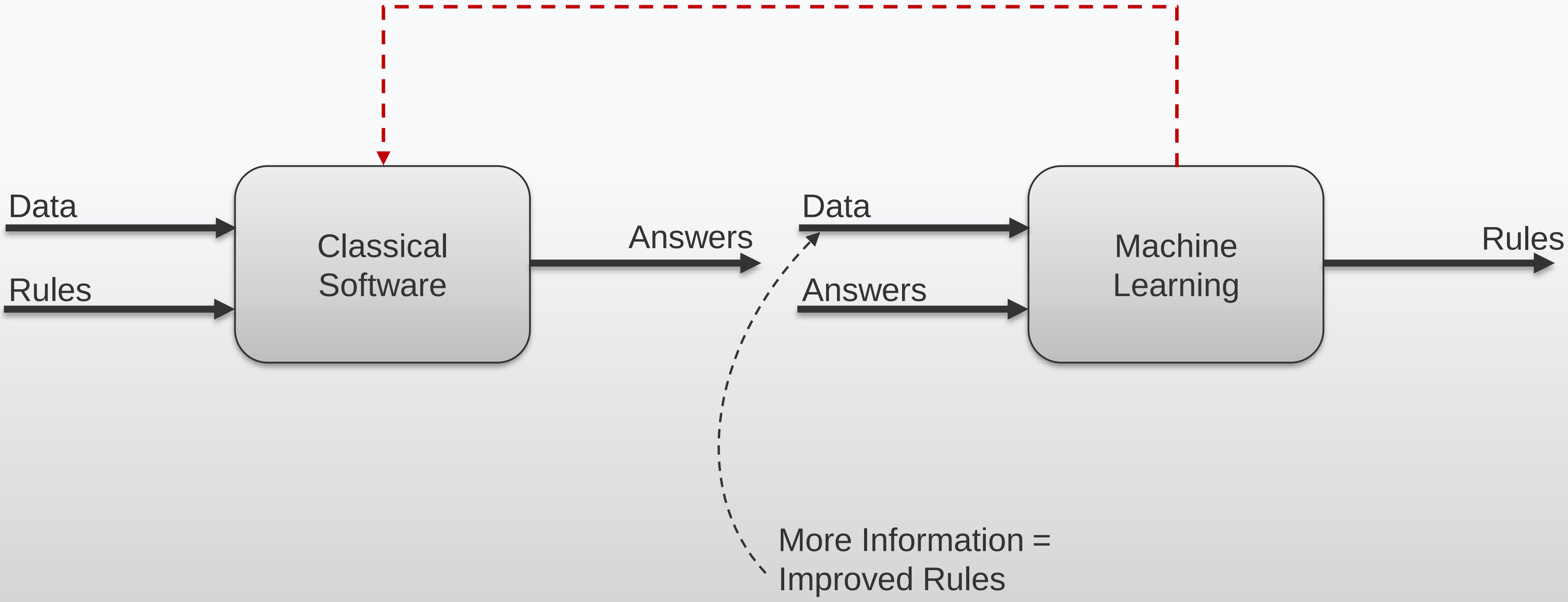


Functional Flowchart for key feature extraction in analogue modulations [1].

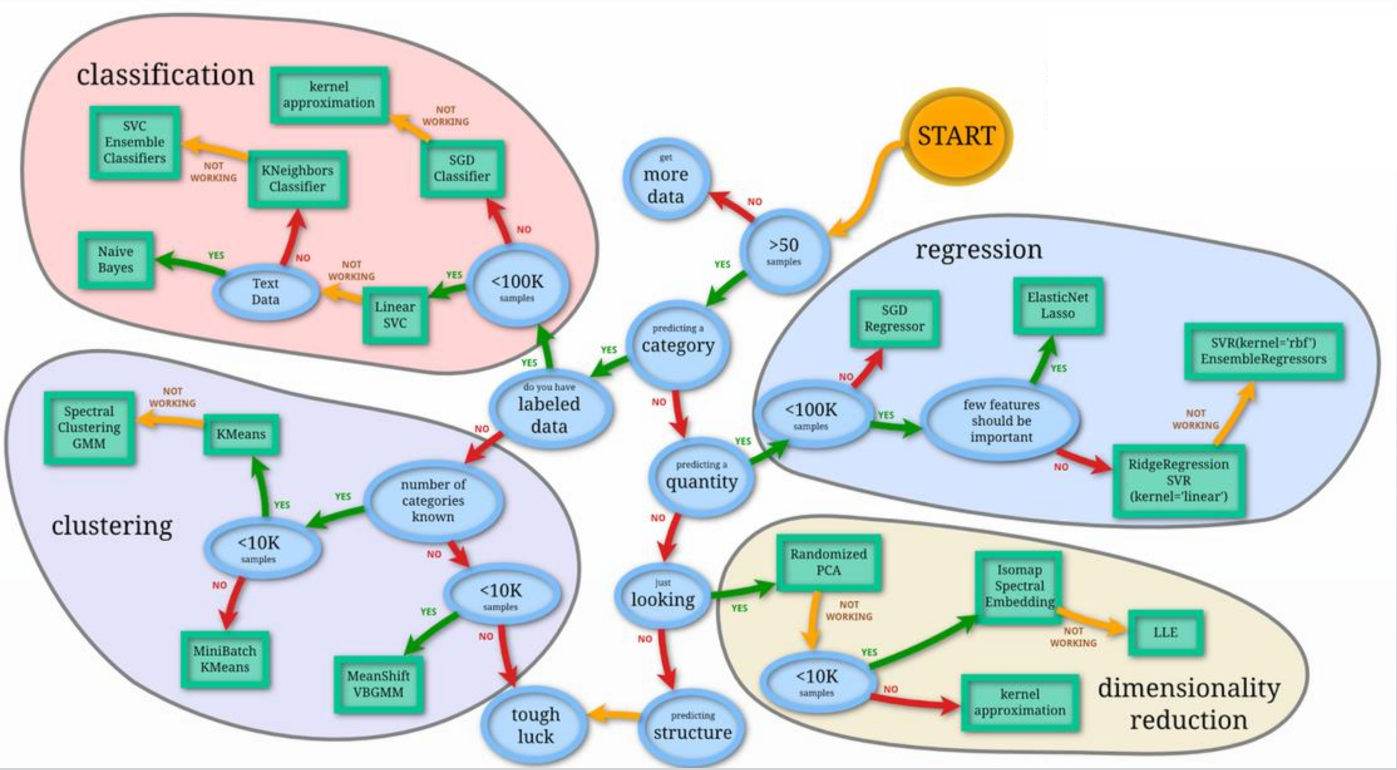


Functional flowchart for modulation classification [1].

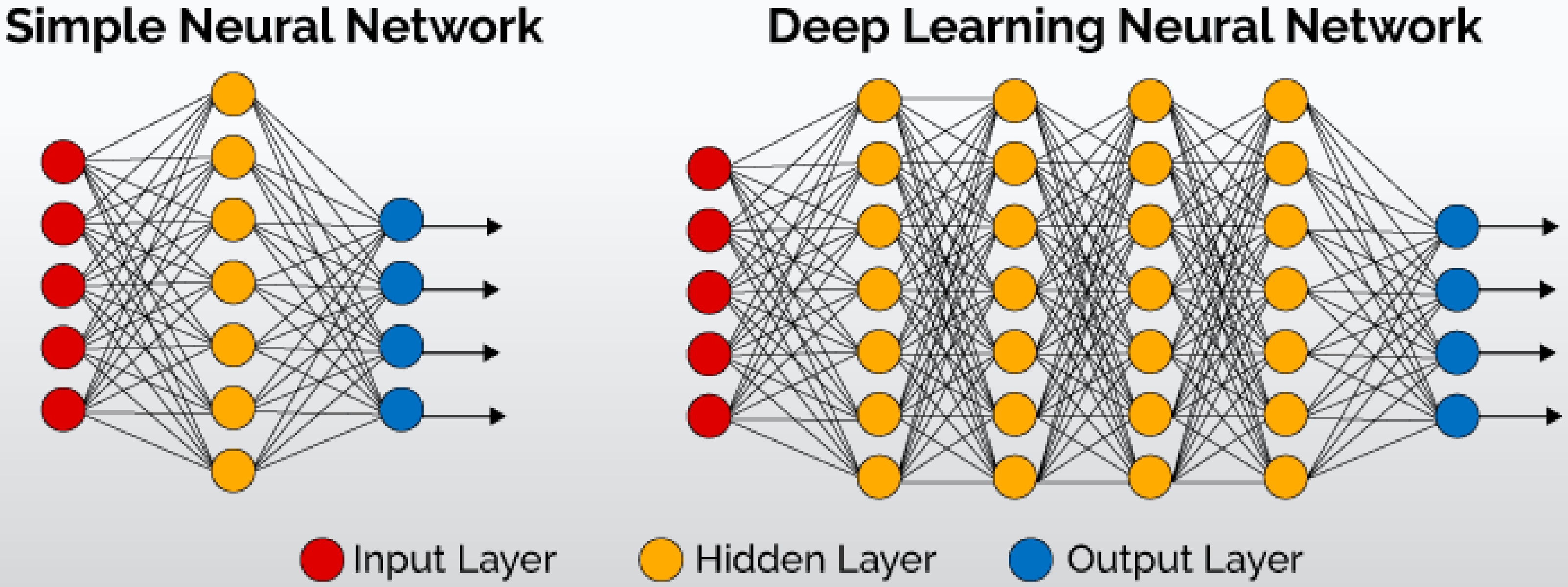
Training vs. Programming



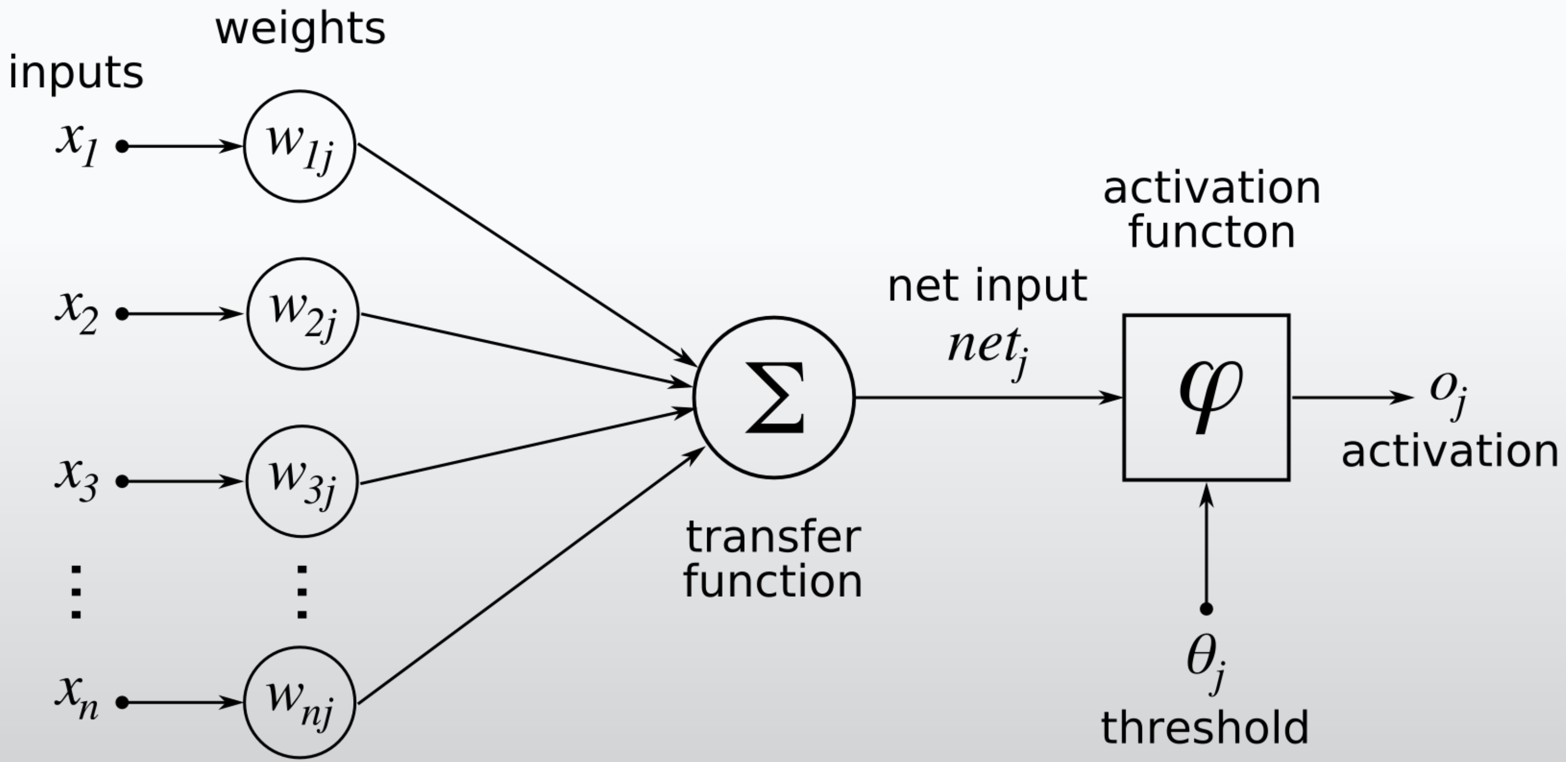
Machine Learning



Deep Learning

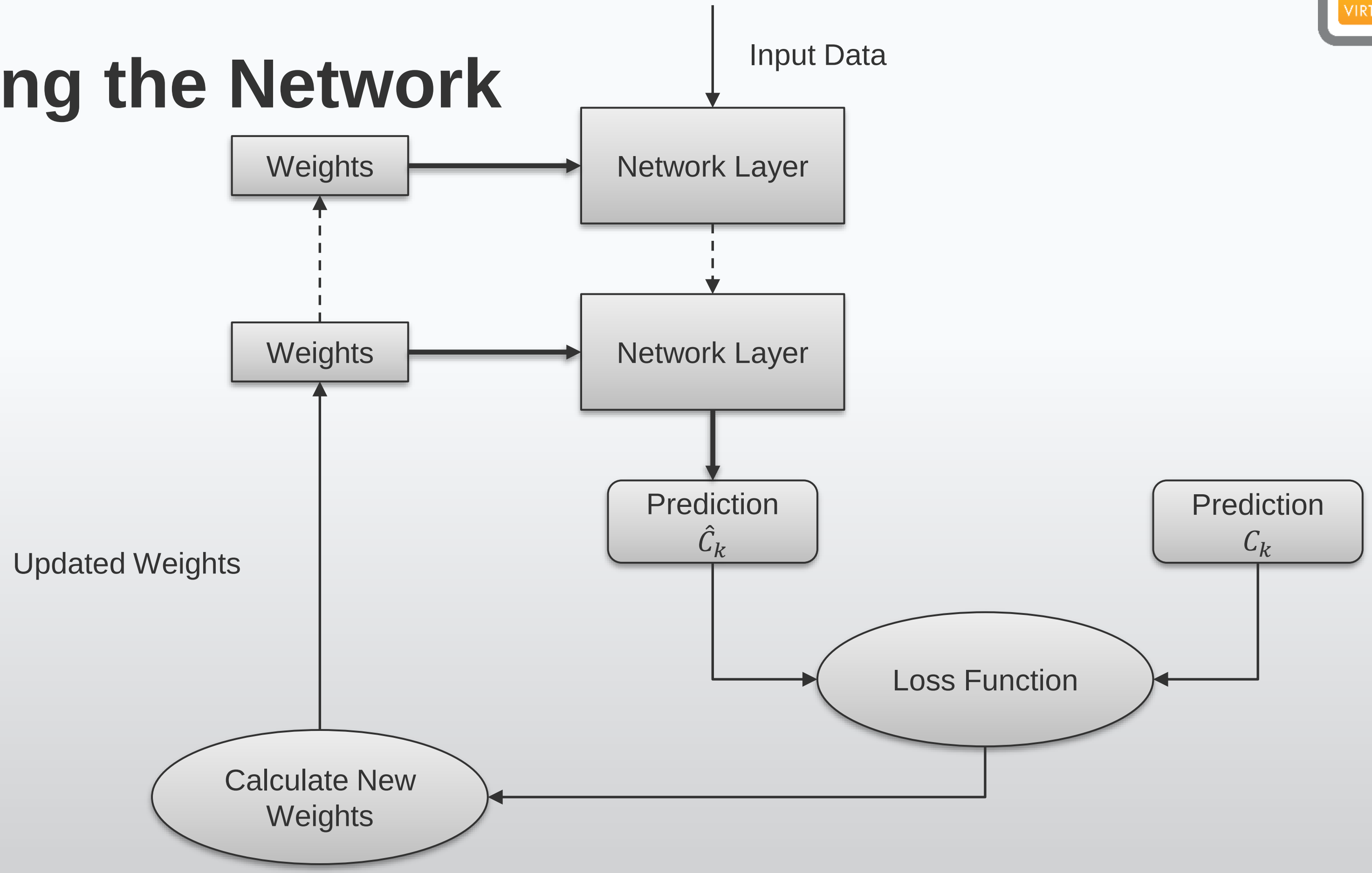


Neuron

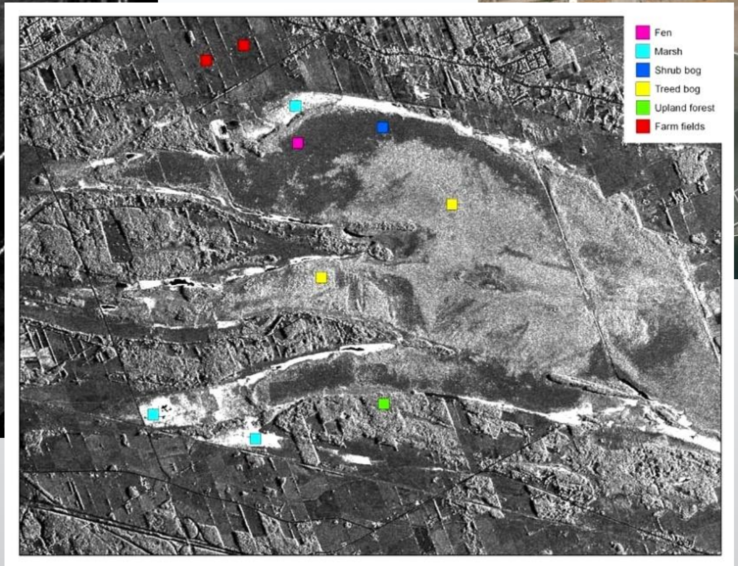


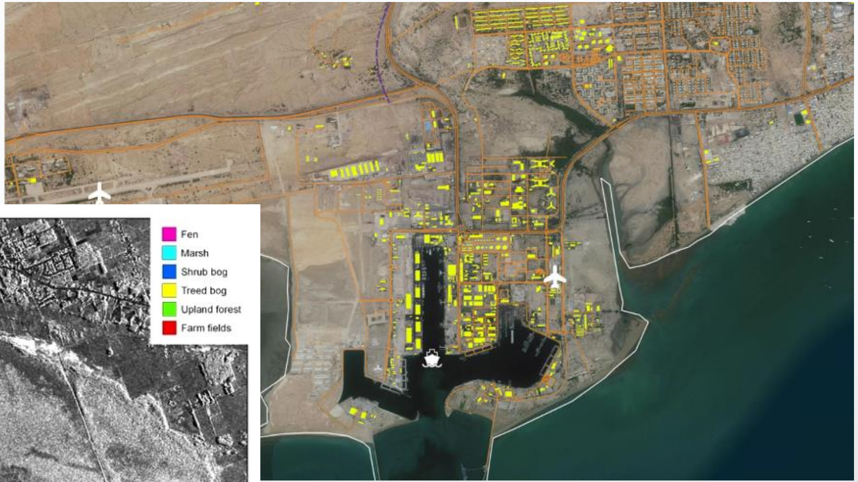
weights
inputs
x_1
w_{1j}
x_2
w_{2j}
x_3
w_{3j}
x_n
w_{nj}
Σ
transfer function
net input
net_j
activation functon
φ
o_j
activation
θ_j
threshold

Training the Network



Where do we see this applied?





Where is the technology now?

- Image Net Challenge
  - 1000 categories
  - 1.4 million colour images
  - 2011 winner = 74 %
  - 2012 winner = 84 % (Hinton & U of T)
  - 2015 winner = 96.4 %
- Deep learning is the prevalent method.
- Deep learning does not require feature engineering.
- Classification is a mature technology.
- Rapidly changing field.
- It has major limitations.

Unsupervised Learning

- This variant of machine learning uses a set of unlabeled data.

$$\{\boldsymbol{x}_i\}_{i=1}^{N}$$

- The objective of unsupervised learning is to create a model that uses the feature vector to solve a practical problem or transforms it into another vector.
- Typical applications include:
  - Clustering
  - Dimensionality reduction
  - Outlier detection
- All are applicable tasks to EW systems.

Reinforcement Learning

- This is a field of machine learning where the system can perceive the state of the environment.
- The machine executes actions that are state dependent.
- Different actions lead to different rewards and can change the state of the environment.
- The goal of machine learning is to learn a **policy**.
- Think of a radar jammer and ASM missile
  - The jammer will typically perform a sequence of attacks which depend on the state of the engagement.
  - The goal of a jammer using reinforcement learning would be to learn the optimal “policy” for these various attacks based on the state of the engagement.
- Reinforcement learning is outside the scope of this course.

Introduction to Reinforcement Learning

Cognitive EW and Reinforcement Learning

ASSOCIATION OF OLD CROWS

Introduction

- A computational approach to learning from interaction.
- Compared to other approaches in machine learning, Reinforcement Learning (RL) is more focused on goal directed learning from interactions.
- RL is learning what actions to take so as to maximize a numerical reward.
  - If a missile is tracking my aircraft, what EA action do I take?
  - What if I've already launched chaff? What if the missile is in a trailing engagement or head-on?
- The learner is not told what actions to take, but must discover the actions that maximize the reward.
- In more complex cases, the action(s) maximize the reward in the current state, but all future states.
- Two important characteristics here:
  - Trial and error
  - Delayed reward

Reinforcement Learning

- A learning agent in RL must be able, to some extent, sense its environment and take a some action to affect the state of the environment.
  - An RWR or ESM system senses the environment.
  - The jammer takes actions to affect the state of the environment.
- The agent must also have goals relating to the state of that environment.
  - Maximizing the miss distance during an engagement.
- Any ML method that contains these three attributes is a RL method:
  - Sense
  - Act
  - Goal

Reinforcement Learning

- A challenge in RL is the trade-off between:
  - Exploration
  - Exploitation
- We want to maximize the goals but at the same time explore new options.
- These new options may produce better rewards but will take time to refine.
  - Maximize a noise jamming attack – simple and easy.
  - Explore a cross-polarization attack – effective but requires precise signal generation.
- There's no simple answer to this problem, which continues to be studied.

Example – Chess

- Chess is a good discrete example.
- The board as a state.
- The actions are defined.
- Goals are quantified
  - Pieces are worth points
  - Checkmate

Example – Jamming

- More analog version…
- Goal is to maximize miss distance during a missile engagement.
- State of aircraft, missile, jammer, etc.
- Actions?
  - Jammer technique
  - Aircraft maneuver
  - Dispense chaff

Elements of Reinforcement Learning

Cognitive EW and Reinforcement Learning

Elements – Policy

- There are four elements of a reinforcement learning system:
  - Policy
  - Reward Signal
  - Value Function
  - Model of the Environment (optional)

- Policy
  - Governs behavior
  - Given state A, do action B
  - Stimulus response rules
  - Policies may be stochastic, associating a probability with each action.

Elements – Rewards

- Rewards signals define the goal of the learning problem.
  - Checkmate
  - Maximize miss distance
  - Prevent search radar detection
- Each time step the model provides a single number, the reward.
- The goal is to maximize the reward over time.
- Reward functions may be also stochastic functions of the state.

Elements – Value Function

- Reward functions give immediate feedback.
- Value functions focus on long term.
- The value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state.
- Near vs. long term thinking
  - Turn hard now reduces available kinetic energy for maneuvering later, when it may be of more value.

Elements – Model of the Environment

- Mimics the behavior of the real environment.
- Allows inferences to be made about how the environment will behave.
- Models are used for planning
  - If I make this move in chess, what are the opponent's available moves, etc.
  - If I perform this EA technique, can the radar track me afterwards or will miss distance be too great.

Finite Markov Decision Processes

Reinforcement Learning and Electronic Warfare

ASSOCIATION OF OLD CROWS

K-Armed Bandit Problem

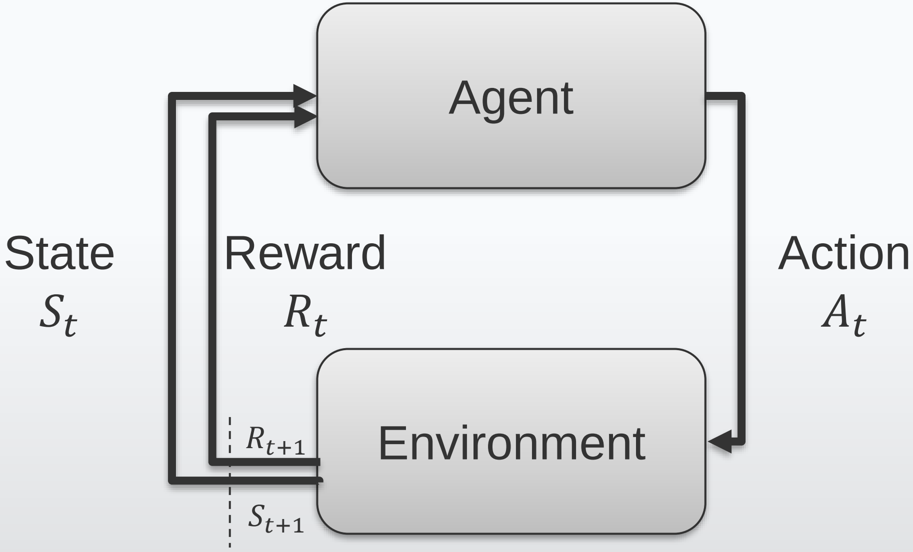
- You have a fixed amount of resources.
- Those resources must be allocated between competing options.
- The goal is to allocate resources to maximize gain (reward).
- The outcome of the allocation is only partially known – probability.
- The outcome will be better understood with time by allocating resources to an option.
- Commonly associated with gambling:
  - What are the machine PDFs
  - Learn the PDFs by playing
  - Allocate choice based on learned PDFs

Finite Markov Decision Processes

- a.k.a. Finite MDP
- Builds on the multi-armed bandit problem.
- Instead of learning just evaluating feedback, we also have an associative aspect.
  - The outcomes will change depending on the state.
  - Think playing cards, and learning who is skilled and who isn't – multi-armed bandit.
  - Now we're learning players are skilled differently in different situations – Finite MDP.
- In a k-armed bandit problem we estimate the values of each action $q_*(a)$.
- In a finite MDP problem we estimate the values of each action in each state, $q_*(a, s)$.
- Alternatively, we can estimate the value of each state, $v_*(s)$, assuming a set of optimal actions.

The Agent-Environment Interface

- MDPs are meant to be a simple framing of the learning problem:
  - Learning from interaction to achieve a goal.
- The learner and decision maker is called an **agent**.
- The **environment** is comprised of everything outside the agent that interacts with it.
- The agent and environment interact continuously.
  - Agent makes decisions
  - Environment responds to decisions
  - Agent makes more decision, etc.



The Agent Environment Interface

- The agent and environment interact at a set of discrete steps.

$$t = 1,2,3, \dots$$

- At each time t, the agent receives a representation of the environment state.
- At each time, based on the state $S_t \in \mathcal{S}$, the agent selects an action $A_t \in \mathcal{A}(s)$.
- One time-step later, the agent receives a reward from its action, $R_{t+1} \in \mathcal{R}$.
- The system then finds its new state, S_{t+1}.
- The MDP is referred to as finite, as there only a finite number of states, actions, and rewards.
  - Finite doesn't mean small!

The Agent Environment Interface

- The state and rewards have discrete probability distributions.
- The distributions depend only on the previous state and action.
- For random variables s' and r at a time t the probability is:

$$p(s', r|s, a) = \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

- In a Markov decision process, the probabilities given by p characterize the environment's dynamics.
- Using this framework we can compute any number of statistical relationships.
- For example, the state-transition probabilities are:

$$p(s'|s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_{r \in \mathcal{R}} p(s', r|s, a)$$

The Agent Environment Interface

- The expected rewards for a state-action pair becomes:

$$r(s,a) = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r | s, a)$$

- The expected rewards for the state-action-next-state becomes:

$$r(s,a,s') = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a, S_t = s'] = \sum_{r \in \mathcal{R}} r \frac{p(s', r | s, a)}{p(s' | s, a)}$$

- The MDP framework is abstract and flexible.
- As a result it can be applied to many different forms of problems.

Monte Carlo Methods

Cognitive EW and Reinforcement Learning

ASSOCIATION OF OLD CROWS

Introduction

- Now we want to:
  - Estimate value functions; and
  - Determine optimal policies.
- We will not assume complete knowledge of the environment.
- Monte Carlo Methods only require experience.
  - Sample sequences of states, actions, and rewards.
  - Derived from actual or simulated data.
- Dynamic programming – which we skipped over – requires complete PDFs of all transitions.
- Imagine being trained from:
  - A large number of chess games.
  - Data from simulated EW-missile engagements.

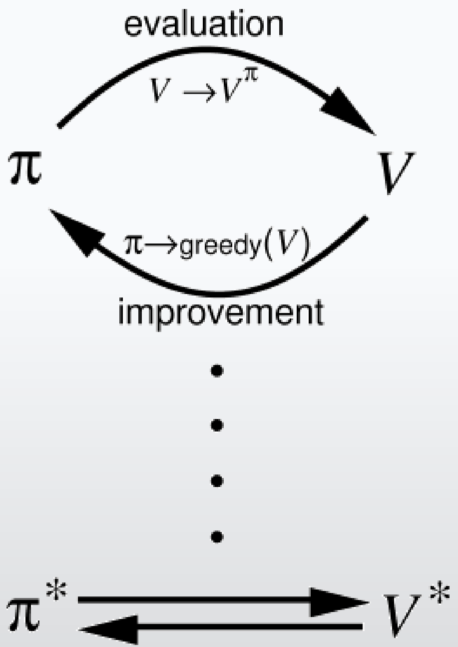
Monte Carlo Methods

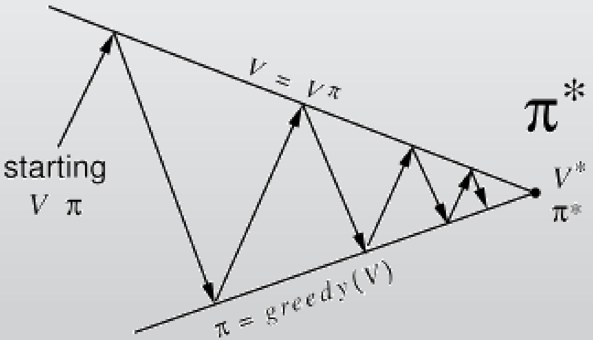
- Monte Carlo methods are often used broadly for estimation methods for any estimation method with a random component.
- We will use MC for averaging complete returns.
- MC methods sample and average returns for each state-action pair.
- Each reward is then average for the state-action pair.
- This is similar to the bandit problem, but in MC there are now multiple states with the action in one state depending on the many future states.
- Since all the actions selections are undergoing learning, the problem is nonstationary.

Monte Carlo Methods

- General policy iteration (GPI)
  - Involves evaluating the policy and value functions until they are consistent with each other.

- Here we have several parameters:
  - $v(s)$ = the state-value function.
  - $q(s, a)$ = the action-value function.
  - π = the current policy.





Monte Carlo Prediction

- The value of a state is:

“The expected return – expected cumulative future discounted reward – starting from that state.”

- One method of estimating the state-value function is to average returns from experience.
  - What is the average state value from known data?
  - As the number of samples increases this converges on the true value.

$$\lim_{n \to \infty} \hat{v}(s) = v(s)$$

- We now want to find the value of the state with a given policy:

$$v_\pi(s)$$

Example – Blackjack

- Each game is an episode.
- Reward:
  - Win/lose/draw = +1 / -1 / 0
- Actions:
  - Hit / stay
- The player learns to make decisions on:
  - Current sum
  - Dealer's showing card
  - Player's usable ace
- 200 total states
- 500 000 games approximates optimal policy.

Estimation of Action Values

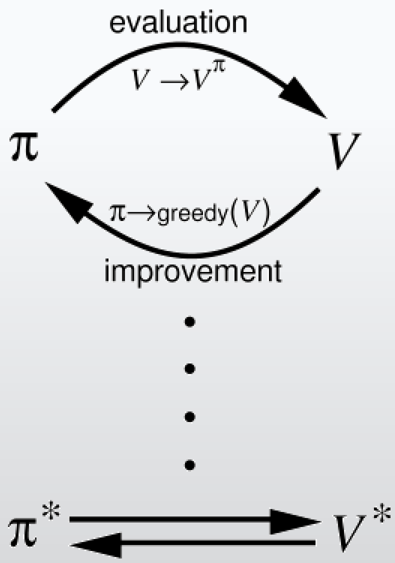
- What if a model is not available for state values?
  - With a model, state values alone are sufficient to determine the policy.
  - You look ahead one step, and determine which action leads to the best combination of reward and next state.
- Then the solution is to estimate action values.
- Our goal is now to find the optimal action, q_*.
- The problem is to estimate $q_\pi(s, a)$.
  - The expected return when starting in state s.
  - Taking action a.
  - Therefore, following policy π.

Estimation of Action Values

- Now how do we find the optimal policy?
- Each state action pair $\{s, a\}$ is said to be visited in an episode if:
  - State s is visited; and
  - Action a is taken.
- Every time $\{s, a\}$ is visited the value is averaged.
- The result is the value of the $\{s, a\}$ pair should converge rapidly.
- We have a problem though, what if the $\{s, a\}$ pair is never visited?
  - Not an uncommon circumstance.
  - Examples include new threat missile or a novel board state in chess.
  - This is the **maintaining exploration** problem.

Monte Carlo Control

- Now how do we use MC to estimate the optimal policy?
- Again, we use Generalized Policy Iteration (GPI)
  - GPI maintains an approximate policy, π, and an approximate value function, q_π.
  - The value function is continuously altered to more accurately approximate the value function for the current policy.
  - Simultaneously, the policy is continuously improved.
  - This creates a moving target with the policy and value function chasing each other.



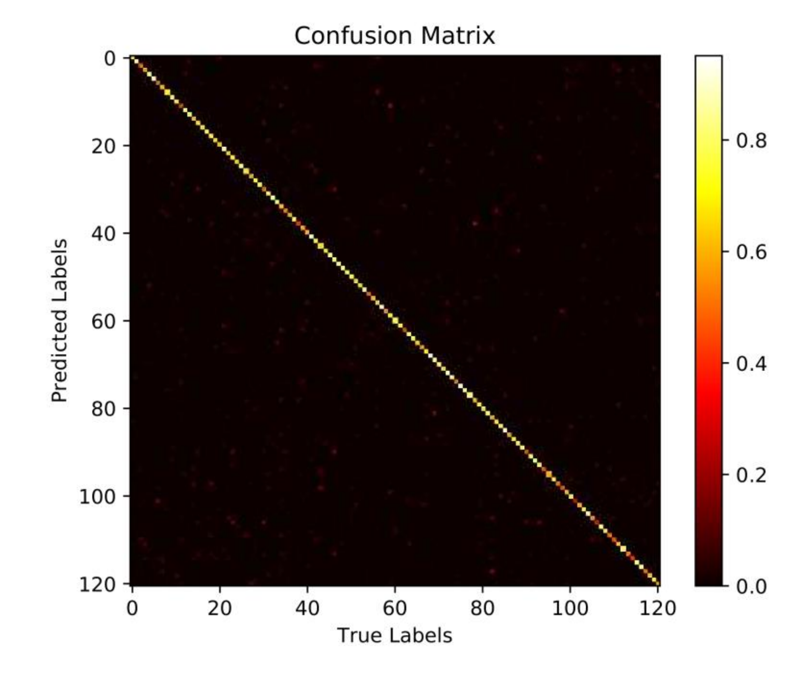
Cognitive EW Systems

Cognitive EW and Reinforcement Learning

ASSOCIATION OF OLD CROWS

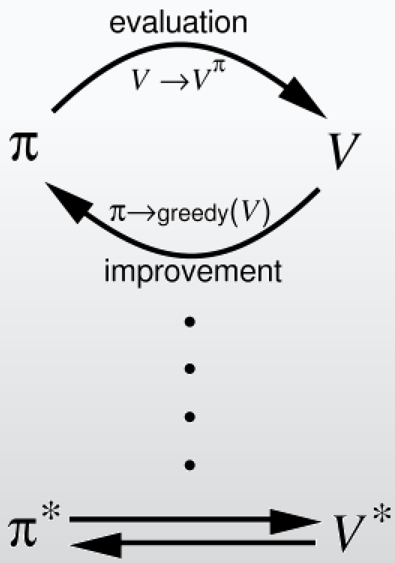
Cognitive EW Systems

- There's a significant difference between EW systems that apply ML and those that are cognitive
- ML is not new to EW
  - Clustering
  - Classification
- Using EW to deal with novel threats in an effective manner is new



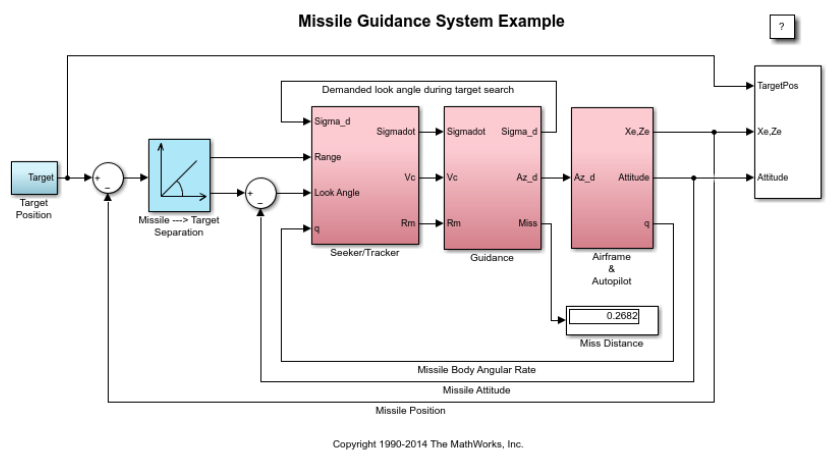
Existing Literature

- There's a few papers on EW and RL.
- Generally of PRC authorship or western authours only describing the problem.
- Largely over simplifications that drastically reduce the problem space.
- It can be applied using existing technologies but need a clear goal and a good model
- EW's problem is the model
  - Lots of interacting systems
  - Goal is not always clear
  - How the action affects the reward can be complex and non-linear
- Need data!



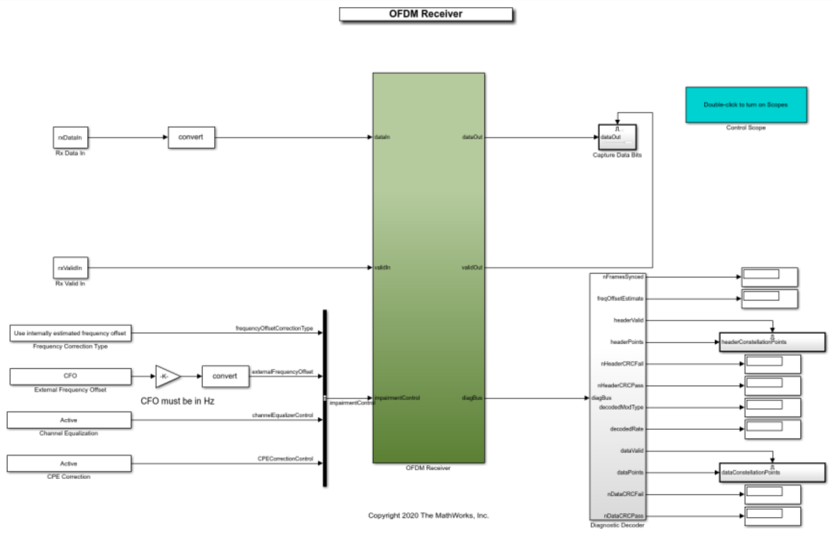
Reinforcement Learning and EW

- What are the model?
  - Missile-target-EA interaction
  - Radio link-EA interaction
- Policy being developed for?
  - Design a cognitive jammer that can deal with novel threats
- Goals for that model?
  - Achieve a desired BER
  - Miss distance in missile engagement
- How do we test the system?



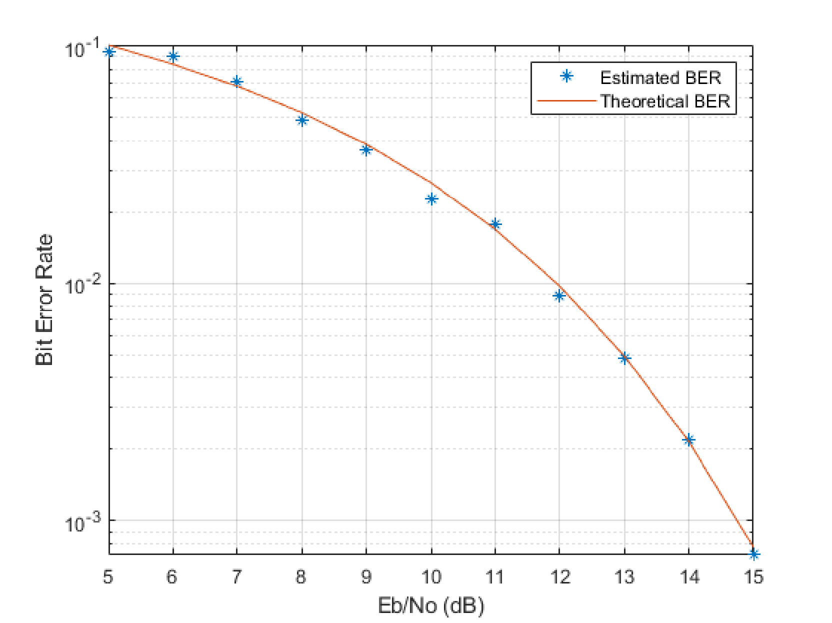
Model

- Actor is the jammer
  - Power
  - Bandwidth
  - Technique
- Need a model for the target and its response
- How to feedback the results into the jammer



Policy and Goals

- Goal is to maximize the BER
- Get the jammer to search for the optimal policy based on its information of the environment
- Limited information is a critical problem



Summary

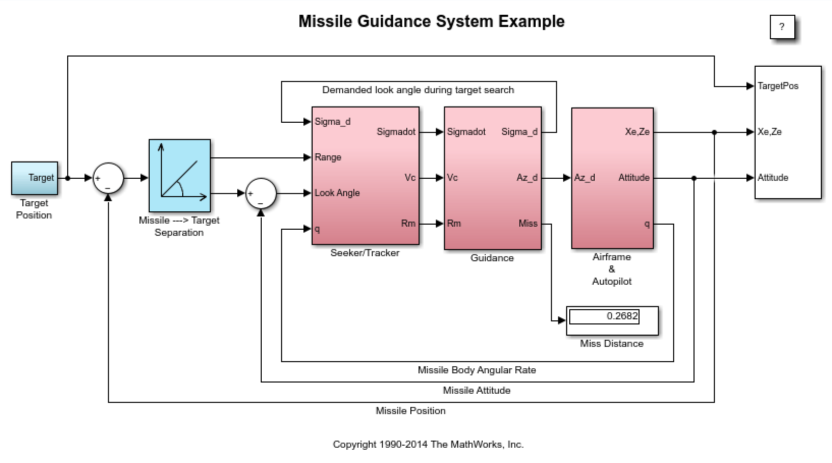
Cognitive EW and Reinforcement Learning

ASSOCIATION OF OLD CROWS

AOC VIRTUAL SERIES

Summary

- Machine learning is maturing in EW
- Main problems are the data and related models
- Reinforcement learning can work well with a valid model, clear goals, and few and discrete interactions



ASSOCIATION OF OLD CROWS

AOC VIRTUAL SERIES

References

[1] E.E. Azzouz, and A.K. Nandi, Automatic Modulation Recognition of Communications Signals, Springer, 1996.

[2] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2011.

[3] I. Goodfellow, Deep Learning, MIT Press, 2016.

[4] P. Pace, Detecting and Classifying Low Probability of Intercept Radar, Artech House, 2009.

[5] R. Sutton, A. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.

[6] L. Graesser, W.L. Keng, Foundations of Deep Reinforcement Learning: Theory and Practice in Python, Addison-Wesley, 2019.

Questions?

- **Kyle Davidson, PhD, CD**
  - Kyle.davidson@agile-em.com
- **Date**
- **Time of Day**

ASSOCIATION OF OLD CROWS