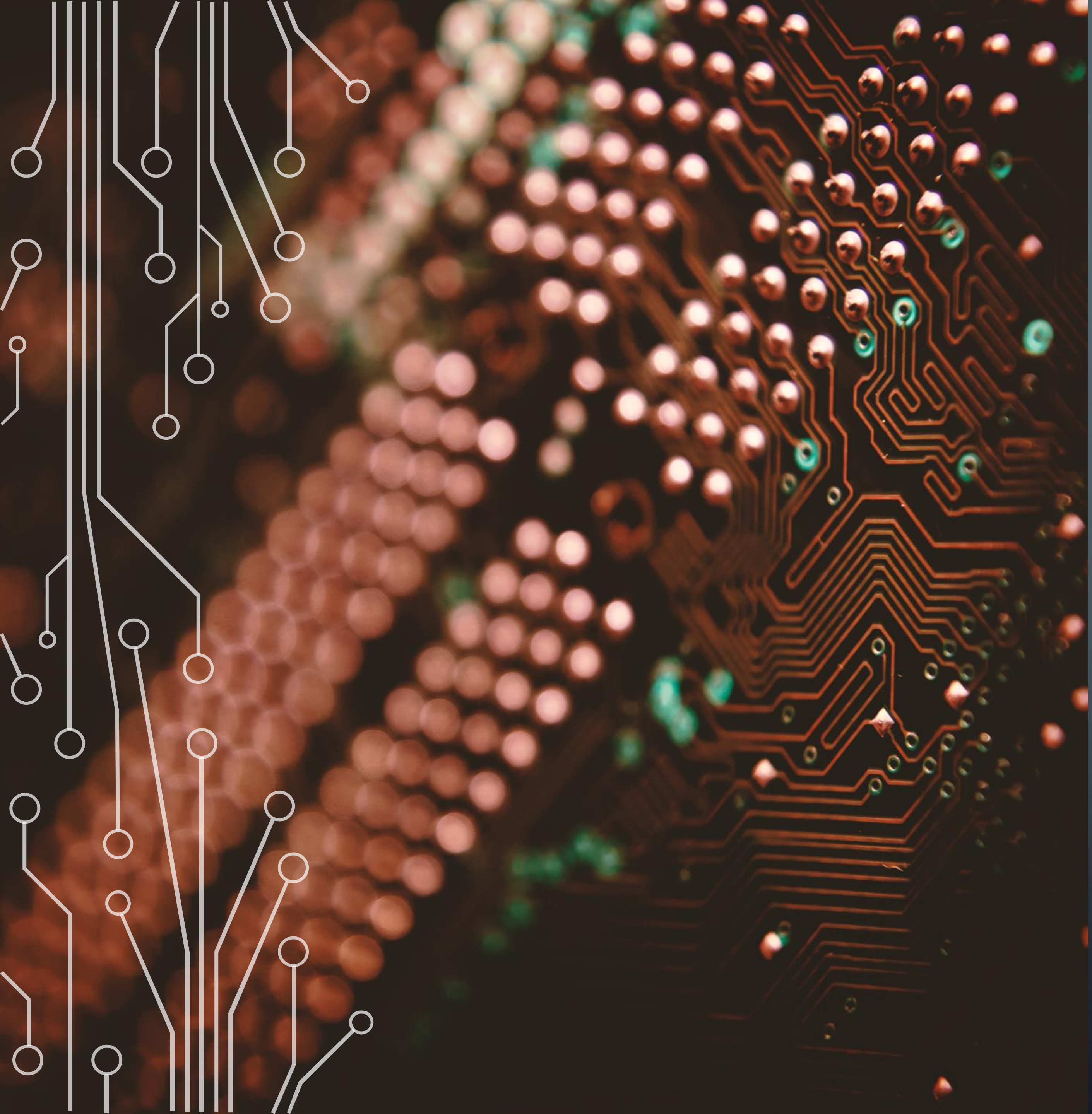
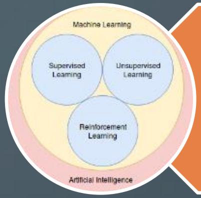
192GEO206T MACHINE LEARNING

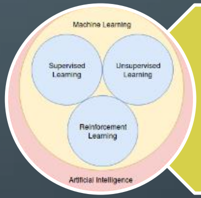
UNIT -1



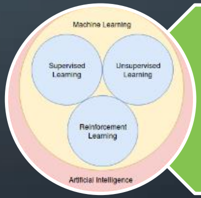
TYPES OF MACHINE LEARNING



Supervised Learning



Unsupervised Learning



Reinforcement Learning

SUPERVISED LEARNING

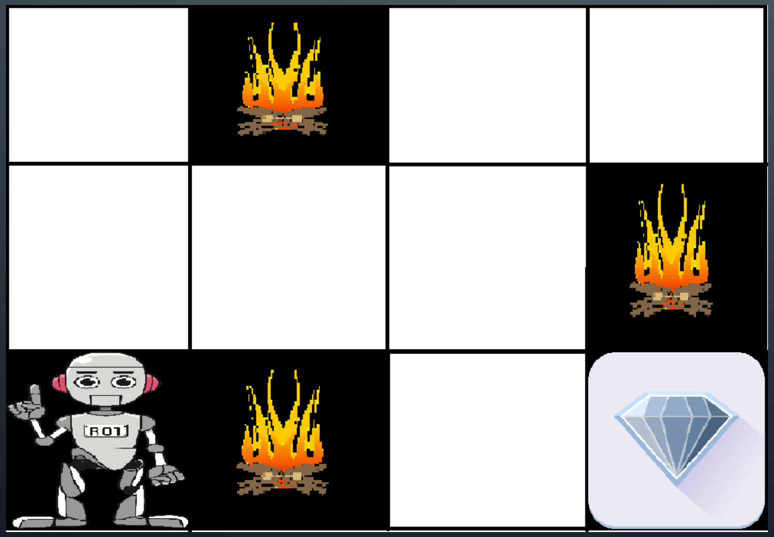
- Supervised learning is a technique in which we teach or train the machine using data which is well labeled.

UNSUPERVISED LEARNING

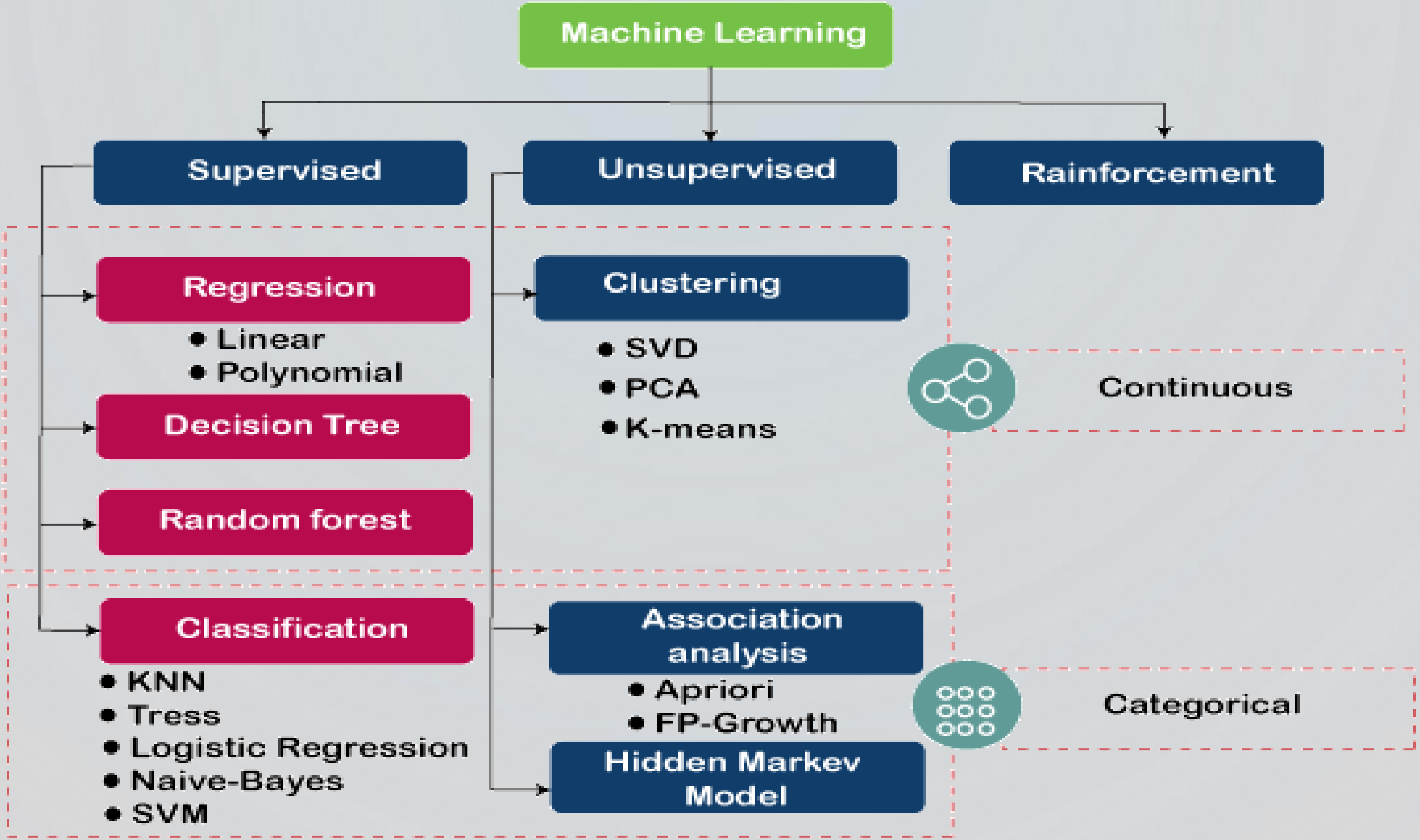
- Unsupervised learning involves training by using unlabeled data and allowing the model to act on that information without guidance.

REINFORCEMENT LEARNING

- Reinforcement Learning is a part of Machine learning where an agent is put in an environment and he learns to behave in this environment by performing certain actions and observing the rewards which it gets from those actions.



TYPES OF PROBLEMS



THE BRAIN AND THE NEURON

- Humans perform complex tasks like vision, motor control, or language understanding very well.
- One way to build intelligent machines is to try to imitate the (organizational principles of) human brain.
- The brain is a highly complex, non-linear, and parallel computer, composed of some 1011 neurons that are densely connected (~104 connection per neuron).
- Some of the neural structure of the brain is present at birth, while other parts are developed through learning, especially in early stages of life, to adapt to the environment (new inputs).

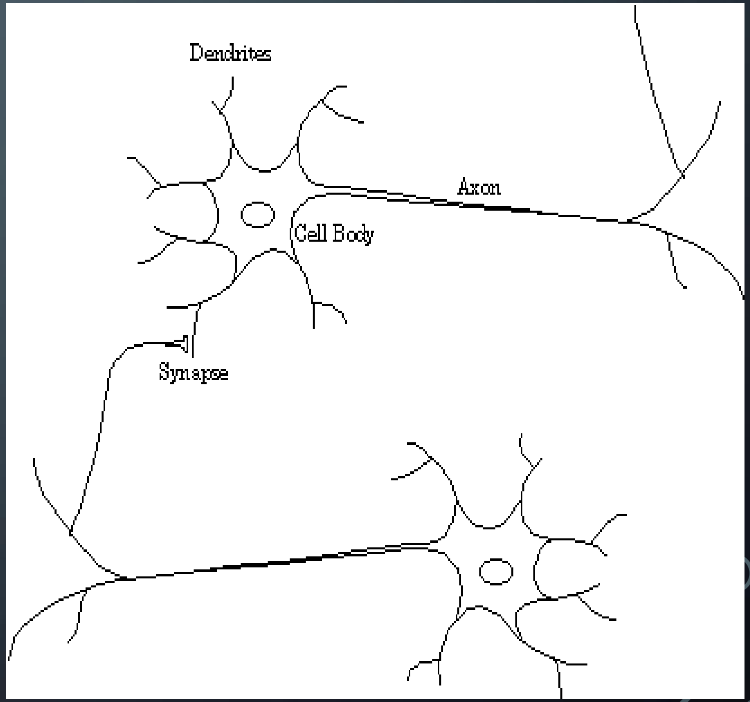
BIOLOGICAL NEURON

- A variety of different neurons exist (motor neuron, on-center off-surround visual cells…), with different branching structures.

- The connections of the network and the strengths of the individual synapses establish the function of the network.

BIOLOGICAL NEURON

- dendrites: nerve fibres carrying electrical signals to the cell
- cell body: computes a non-linear function of its inputs
- axon: single long fiber that carries the electrical signal from the cell body to other neurons
- synapse: the point of contact between the axon of one cell and the dendrite of another, regulating a chemical connection whose strength affects the input to the cell.



ARTIFICIAL NEURAL NETWORKS

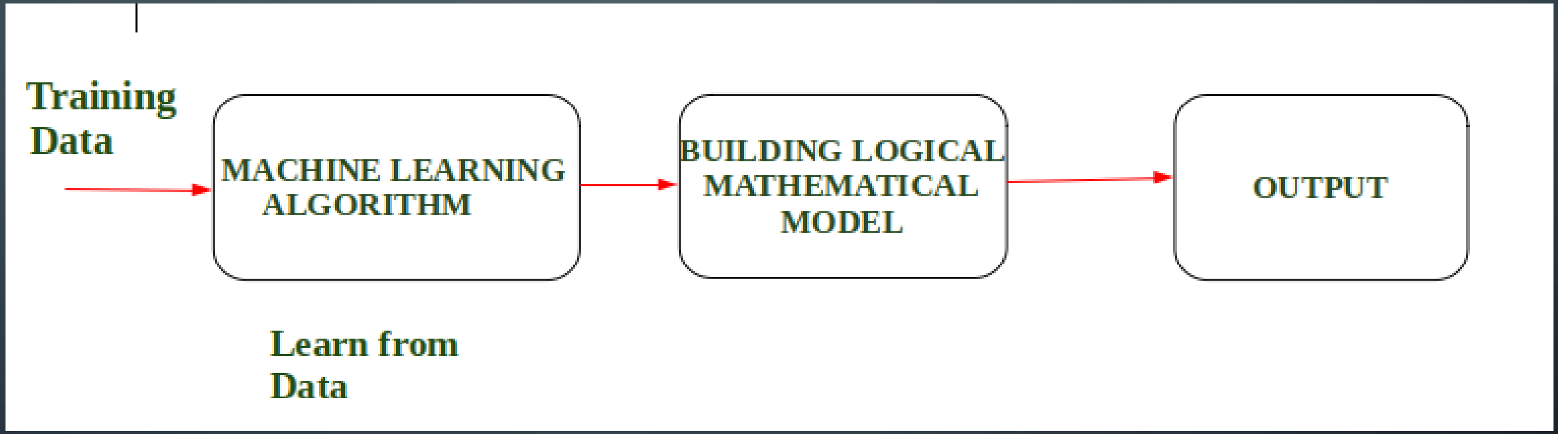
- Computational models inspired by the human brain:
  - Massively parallel, distributed system, made up of simple processing units (neurons)
  - Synaptic connection strengths among neurons are used to store the acquired knowledge.
  - Knowledge is acquired by the network from its environment through a learning process

HEBB'S RULE

- computes the weighted sum of its input wi
- An adder
- An activation function

DESIGNING A LEARNING SYSTEM

- According to Arthur Samuel “Machine Learning enables a Machine to Automatically learn from Data, Improve performance from an Experience and predict things without explicitly programmed.”



According to Tom Mitchell, “A computer program is said to be learning from experience (E), with respect to some task (T). Thus, the performance measure (P) is the performance at task T, which is measured by P, and it improves with experience E.”

EXAMPLE

Example: In Spam E-Mail detection,

Task, T: To classify mails into Spam or Not Spam.

Performance measure, P: Total percent of mails being correctly classified as being “Spam” or “Not Spam”.

Experience, E: Set of Mails with label “Spam”

DESIGNING A LEARNING SYSTEM

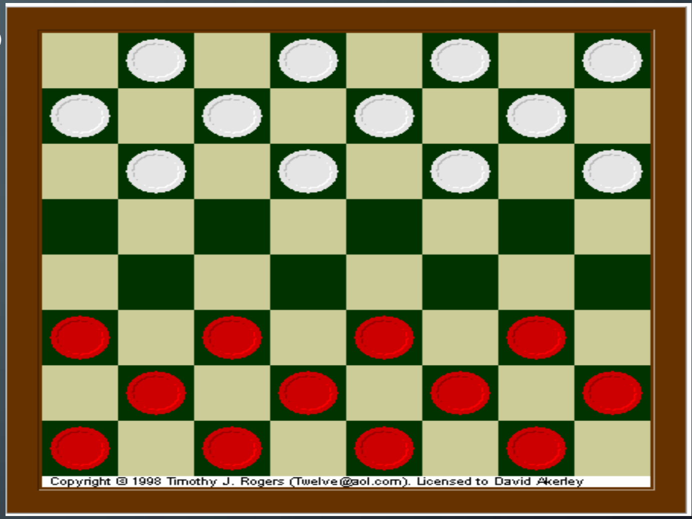
1. Choosing the training experience

- Type of training experience from which our system will learn.

- The type of training experience plays an important role in the success or failure of the learner.
  - **One key attribute** is whether the training experience provides direct or indirect feedback regarding the choice made by the performance system.
  - **The second key attribute** of the training experience is the degree to which the learner controls the sequence of training example.
  - **The third key attribute** of the training experience is how well it represents the distribution of examples over which the final system performance P must be measured.

DESIGNING A LEARNING SYSTEM-CONT'D

- In order to define the training experience, we must cho
  - The exact type of knowledge to be learned.
  - Representation of this target knowledge.
  - Learning mechanisms.



Task **T** : playing checkers

Performance measure **P**: % of game won against opponents

Training experience **E** : playing practice game against itself

DESIGNING A LEARNING SYSTEM-CONT'D

2. Choosing the target function

To determine the exact what type of knowledge will be learned and how this will be used by the performance program.

Eg:

Let's begin with the legal moves a bot can take. Legal moves are the moves our bot(the model) can take which are correct. Now the bot needs to learn to choose the best moves among these legal moves in situations.

DESIGNING A LEARNING SYSTEM-CONT

- Let's call this function **ChooseMove**, which chooses the best moves for the bot.

ChooseMove : M→B

which takes input, set of legal moves M and outputs the best moves B

- To make ChooseMove performance P better with experience E, we set a numerical score as **TargetFunction(V)**.

TargetFunction (V): B → R

- V maps any best move to some real value R, and intend for this target V to assign higher scores to better board states.

i.e,

if b is the final state, **won**, V(b) = 100

if b is the final state, **lost**, V(b) = -100

if b is the final state, **draw**, V(b) = 0

if b is the final state, V(b) = V(b')

- where b' is still the best state that can still be achieved.

DESIGNING A LEARNING SYSTEM-CONT'D

3. Choosing representation for the target function

- Represent V using a collection of rules that match against features of legal moves or a quadratic polynomial function of predefined moves or an artificial neural network.
- Thus our learning program can represent V^(b) as a linear function:

$$\hat{V}(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

w = numerical coefficient

x = legal moves

DESIGNING A LEARNING SYSTEM-CONT'D

4. Choosing a function Approximation Algorithm

- Each training example is an ordered pair of the form **< b,V train (b)>**

5. **Estimating training values**

- Assign the training values of <V train (b)> for any intermediate board state b to be <V^(successor(b))>, where V^ is bot's correct approximation to V.
  - successor(b), next move following b.

  which can be summarised as :

$$\mathbf{V\ train(b) \leftarrow V\hat{}(successor(b))}$$

DESIGNING A LEARNING SYSTEM-CONT'D

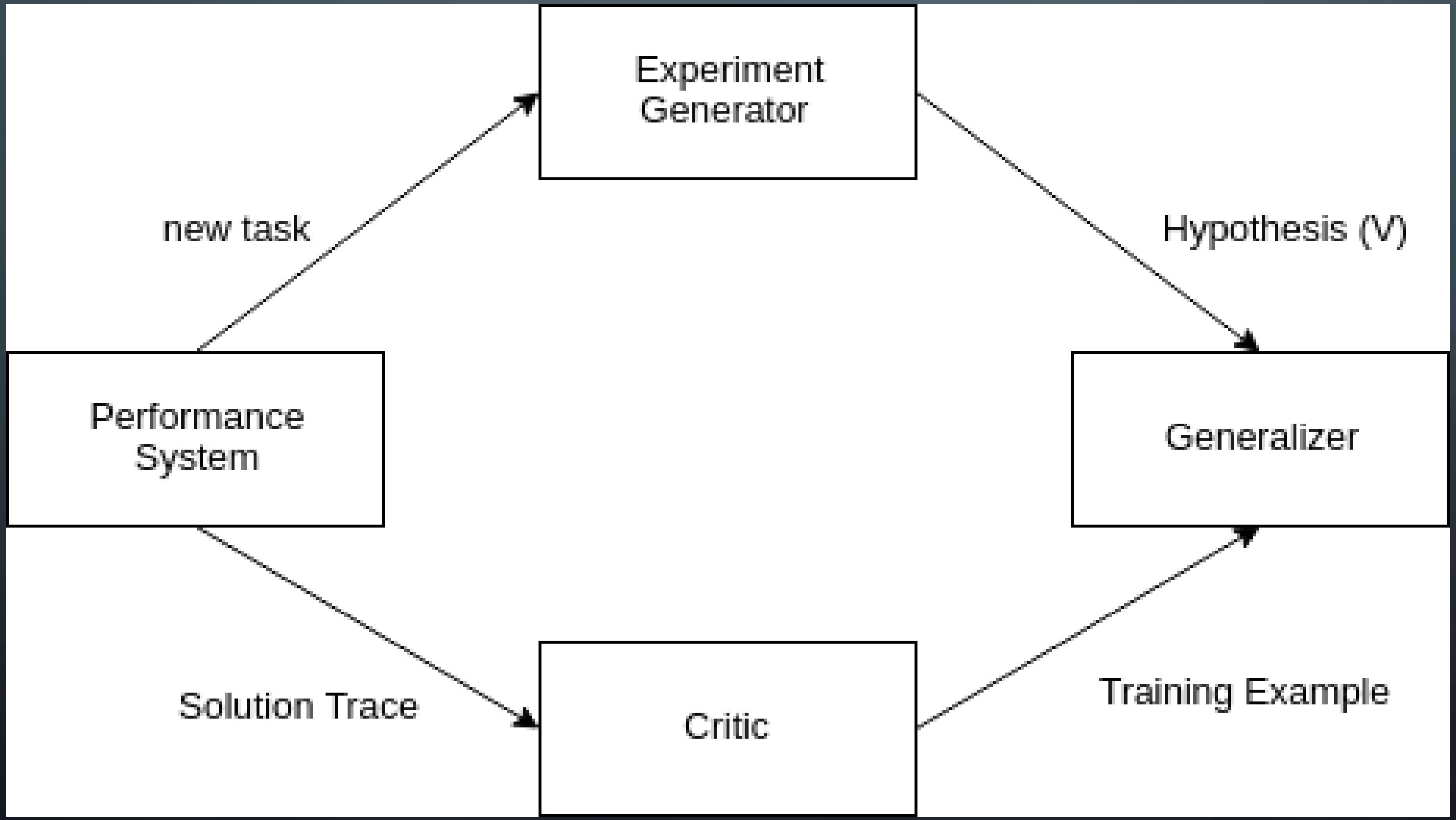
6. Adjusting the weights

- To define the best hypothesis, or set of weights, or approach, is to adjust the weights to minimise the squared error E between the training value and the values predicted by the hypothesis $\hat{V}$.

$$E \equiv \sum_{\langle b, V_{train}(b)\rangle \in \, training\ examples} (V_{train}(b) - \hat{V}(b))^2$$

DESIGNING A LEARNING SYSTEM-CONT'D

Final Design- for Checker problem



PERSPECTIVE & ISSUES IN MACHINE LEARNING

- **Perspective:**

It involves searching a very large space of possible hypothesis to determine the one that best fits the observed data.

- **Issues:**
  - ✓Which algorithm performs best for which types of problems & representation?
  - ✓How much training data is sufficient?
  - ✓Can prior knowledge be helpful even when it is only approximately correct?
  - ✓The best strategy for choosing a useful next training experience.
  - ✓What specific function should the system attempt to learn?
  - ✓How can learner automatically alter it's representation to improve it's ability to represent and learn the target function?

THANK YOU