ENPM690 MIDTERM

Sahruday Patti

March 2022

1. **Describe at least 5 differences between Local and Global Learning. Identify 3 methods for both, respectively.**

   Local learning algorithms attempt to locally adjust the capacity of the training system to the properties of the training set in each area of the input space. Local learning utilizes a locally distributed weight representation, thus requiring update of only a small subset of all network weights per pattern per iteration.

   Local learning used in learning classifiers from data, tries to employ a subset of input points around the separating hyper plane. It employs part of information critical for the specifically oriented tasks and does not assume a model for the data. For eg in Support Vector Machines, The separating plane is based on the distance between the points on either side of the plane separating and the overall data structure/pattern is not taken into account.

   Local learning experiences no Local Minima and hence, convergence is guaranteed and it converges to global minimum as, these models does not intend to build an accurate model to fit the observed data globally.

   Local learning makes no attempt to model the date globally, and focuses on extracting only that information which is directly related to the task. Because of this, local learning is considered more direct and is computationally cheap

   Global Learning algorithms attempt to produce a model that fits the data globally.Global learning, utilizes a completely distributed weight representation, thus requiring update of all network weights per pattern per iteration.

   Global learning summarizes the data completely and globally, which makes it less direct and computationally expensive

   Global learning experiences Local Minima, and may not converge to global minima.Also, Global learning experiences very slow convergence as it is less direct and computationally expensive.

   Global learning summarizes the data and provides the knowledge on the structure of data, since with the precise modeling of phenomena, the observations can be accurately regenerated and therefore thoroughly studied.

   Local Learning Methods:
   Support Vector Machines (SVM), Gabriel Graph, Neural Networks

   Global Learning Methods:
   Maximum Likelihood Learning,Conditional Learning,Bayesian Average Learning

2. **Describe at least 5 differences between Lazy and Eager Learning. Identify 3 methods for both, respectively.**

   Lazy learning is an algorithm which does not build a model immediately after receiving the training data, it waits till it is provided with an input data to evaluate. i.e Lazy learning is a learning method that delays abstracting from the data until it is asked to make a prediction. Lazy learning simply stores the training data without learning from it i.e it does not build a model immediately after receiving the training data.Lazy learning methods wait to classify data until it receives the test data.

   Lazy learning requires a large space to store the entire training data set as it does not build the model immediately.

   Lazy learning algorithms can create many local approximations. Majority of computation time is spent on predicting the data and not on training it.

   Lazy learning method uses a richer hypothesis space as it uses many local linear functions to form its implicit global approximation to the target function.

   Lazy learning methods:
   Lazy Bayesian rules, K - Nearest Neighbor, Case-Bases Reasoning, Locally Weighted Regression.

   Eager learning is an algorithm which builds a model soon after receiving training data set. Eager learning is a learning method in which the system tries to construct a general, input-independent target function during the training of the data i.e Eager learning methods construct general, explicit descriptions of the target function based on the provided training data.

   In eager learning, given a set of training data, constructs a classification model before receiving new data to classify. So, when training data is received, it does not need to store the training data set and hence, requires much less space.

   Eager learning algorithms take a longer computation time to train the data and less time to predict.

   Eager learning creates a global approximation i.e Eager learning algorithms must commit to a single hypothesis that covers the entire instance space.

   Eager learning methods: Decision tree induction, Multi layer Perceptron Models, Support vector machines, Cerebellar Model Articulation Controller (CMAC).

3. **Explain the Origins of CMAC Architecture for Machine Learning.**

   The Cerebellar Model Articulation Controller (CMAC) was proposed by J. S. Albus in 1975. It is inspired from how the Control Functions in Natural Biological Organisms like Human Brain are able to perform computations quickly and produce the desired movement. CMAC was inspired from trying to learn this mapping. CMAC was emphasized to be understood as a table referring technique that can adaptive to real-time control system and Later CMAC as Neural Network perspective was also presented and a number of algorithms were produced like DCMAC,KCMAC etc.

   The origins of the CMAC architecture for Machine Learning lie in two parts. First part is emulating the fundamental properties of human brain. The part of the brain that is involved in motor control process is the cerebellum. The CMAC is based on the,Then understanding of the human brain formulated as : The input to brain constitutes an address, the contents of which are the appropriate signals to carry out a desired moment. At each point on the trajectory the state of the limb is sent to the cerebellum as input and cerebellar memory responds with actuator signals which drive the limb to the next part of trajectory and the process is repeated. The mathematical concepts of how the cerebellum structures input data, how it computes the address of where control signal are stored, how the memory is organized and how the control signals are generated. These are the basic principles organized into a control system CMAC(Cerebellar Model articulation Controller).

The second part is that, at that time and even today Perceptron is considered as analogous to cerebellum. The control function is rather a smooth continuous function. For every point in input space a small neighbourhood produces similar output and different outputs widely separated. The Perceptrons ability to generalize over a small neighbourhood and having good dichotoimizing properties for points well separated would be a good controller. The CMAC architecture utilizes both of these parts in order to produce a control system which is analogous to biological systems thus in a way providing an insight to control systems real time.

The output of a CMAC is the algebraic sum of the weights in all the memory cells activated by the input point which is similar to the Perceptron model but with an additional mapping. Thus, CMAC took a different perspective in learning by this additional mapping which made memory requirement proportional to number of cells used. The CMAC also showed improvement in the error as the number of Iterations of data increased and there are various versions of CMAC leaning algorithms that has been developed.

In essence all Machine Learning algorithms are generating a function which given some inputs produces the desired output Hence, Machine learning Fundamentals says that, the data we are using is Identically Independently distributed and is generated from a Target function which we are trying to approximate using ML algorithms. Thus, although CMAC is first proposed for Control systems the essence lies in generating a target function which is the problem of Machine learning. During the past decades it has been proved effective in robotic arm control, piezoelectric actuator control, mechanic system, signal processing, intelligent driver warning system and many other fields. The additional mapping which made CMAC analogous to bilogical systems which can be attributed to the fast learning and stable performance of CMAC models made it superior. But, i feel The Key highlight of CMAC is that it showed we can have different types of learning rules. Unlike the Neural Networks which can use different learning rules in different parts, CMAC provided us with the encoding and decoding like a bilogical organism.

4. **Identify and Describe at least 3 key features,2 advantages and 2 limitations of CMAC for Machine Learning**

In essence all Machine Learning algorithms are generating a function which given some inputs produces the desired output Hence, Machine learning Fundamentals says that, the data we are using is Identically Independently distributed and is generated from a Target function which we are trying to approximate using ML algorithms. The essence of CMAC also lies in approximating the target function.

The Key Features of CMAC for ML :
1. The CMAC algorithm key is to generalize over a small neighbourhood of space and to be able to dichotomize for inputs that require different outputs. It derives this ability from the Perceptron Dichotomizing Properties which is extensively used in Machine Learning.
2. If we have enough memory requirements proportional to the complexity of the problem at hand we can approximate the optimal function of behaviour based on the inputs.
3. The other unique feature of CMAC is the mapping algorithm which converts the distance between input vectors(Hamming Distance) into the degree of overlap between sets of addresses where functional values are stored. CMAC is thus a memory management technique.
4. The learning algorithm of CMAC is based on back propagation. But this can be modified to learn the mapping differently and there are various types of learning that has been used here including Reinforcement Learning. Thus, it paves way for research in new types of Machine Learning algorithms.

The advantages of Using CMAC :

1. Multi Layer Perceptron model is fully connected but CMAC restricts the association in a certain neighboring range. This property of mapping significantly accelerates the learning process of CMAC,

which is considered a main advantage of it comparing to other neural network models.

2. If we have enough computation power and hardware, with the help of CMAC we might be able to learn and approximate the target function quicker,more accurate than other ML algorithms.

The Limitations of CMAC:

1.CMAC algorithm is dependent upon the Physical memory available to allocate the association cells. As we increase the memory size more and more weights can be allocated in turn increasing the dichotomizing ability of CMAC. As we increase the memory size, the computation time also increases.Also, CMAC can approximate the function closely but its accuracy is definitely a concern. say,we are simulating a control function for an autonomous car, we need to be more precise in our output motor commands in order to be practically useful.

2. Local generalization mechanism may cause some training data to be incorrectly interpolated, especially when data is sparse considering the size of CMAC.

3. Many more weight parameters are needed comparing to normal MLP model

5. **Describe a form of Machine Learning discussed in class that uses no weights, generate a Pseudo code and explain how it works. Describe the limitations of such methods.**

Some of the types of Machine Learning that does not use weights for learning is Reinforcement Learning(RL) and Evolutionary Algorithms. RL is learning what to do - how to map situations to actions so as to maximize a numerical reward signal. We formalize the problem of RL using ideas from dynamical systems theory specifically as the Optimal control of incompletely known Markov-Decision Processes. The Markov chain states that probability of a certain state given the present state is equal to the probability of certain state occurring given all the past states. That is future is independent of the past given the present. We also use RL in POMDPS(partially Observable Markov Decision Processes) with a slightly different modification as the Model of the environment is Unknown.

There are four main sub elements of an RL problem, a Policy, a Reward Signal, a Value Function and optionally a Model of the environment.Reinforcement learning is about learning from interaction how to behave in order to achieve a goal. The reinforcement learning agent and its environment interact over a sequence of discrete time steps. The specification of their interface defines a particular task: the actions are the choices made by the agent; the states are the basis for making the choices; and the rewards are the basis for evaluating the choices. Everything inside the agent is completely known and controllable by the agent; everything outside is incompletely controllable but may or may not be completely known. A policy is a stochastic rule by which the agent selects actions as a function of states. The agent's objective is to maximize the amount of reward it receives over time. We use a Discount Factor in the Expectation of rewards in order to control how far into the future we look into.

There are various forms of RL which involve model free and model based formulations like Dynamic Programming algorithms or Model free algorithms like Q learning, SARSA etc. Here i am describing one such algorithm called SARSA. Consider an Example of Autonomous Driving Using Reinforcement Learning using SARSA. The Pseudo Code is as follows:

```
Initialize Q(S,a) arbitrarily for s $\epsilon$ S, a $\epsilon$ A and Q(terminalstate)=0
Repeat(for each episode)
Initialize S
Choose A from S using Policy derived from Q(e.g. $\epsilon$ greedy)
for each step of episode do
Take action A, observe R, $s^{'}$
Choose action $A^{'}$ from $s^{'}$ using policy derived from Q (e.g. $\epsilon$ greedy)
$Q(S,A) \leftarrow Q(S,A) + \alpha[R + (\gamma Q(s^{'}, A^{'}) - Q(S,A))]$
$S \leftarrow s^{'}, A \leftarrow A^{'}$
```

until S is terminal

The SARSA algorithm is an on policy algorithm which involves updating the state action pairs and improving the policy over time. First we initialize the State action pairs look up table. this look up table is like if you are in a particular state and you take a particular action what are the probabilities that you end up in the next state. So, in out driving case, say there is an obstacle in front, this is the state. You take an action right what is the probability that you end up in the next state which can be another obstacle state or a free state with a probability associated to it. This is how we characterize look up table. we can initialize this arbitrarily say we initialized to zero. Now, for an episode, where episode is one full simulation run till the terminal state is reached. So, for every episode, For every single time step, time step is like the car is given the data and it should give a response, an action to take say left or right or straight. For every single time step, given a particular state take an action based on the policy and observe the reward and the next state. Now choose the action for the next state based on the epsilon greedy policy. This is the exploration vs exploitation problem. We want see all the actions from a particular state but also we want to take actions which have maximum reward but we dont know what action to take from particular state until we explore all the actions. This is termed as Exploration vs Exploitation Problem and we act Epsilon Greedily at every state. Now, after choosing the action, we update the Action value in the direction of my target function. here alpha is the step size, gamma is the discount factor. And we repeat this entire process for every single time step of the episode for all the episodes. In terms of Autonomous Driving Problem, when we initialize the simulation run, for each time step of the episode we observe the state take an action. After that we observe the next state and current reward for taking that action. Now given the next state we take an action epsilon greedily which maximizes our action value function. maximizing our action value function means we are trying to choose actions which improve my overall reward for the entire episode. Once we choose an action, we evaluate the action value of the state we are presently in and we update the action value of that state. Notice that SARSA is on policy learning, that is if we encounter the same state in the same episode again it will act on the policy that we updated during the episode.

Some of The Limitations of Reinforcement Learning Methods are:

1.One of the challenges that arise in reinforcement learning, and not in other kinds of learning, is the trade-of between exploration and exploitation.

2. This type of learning involves learning by doing. So it is real time learning. This will take a large time to converge but also takes lots of resources to learn.

3. The RL assumes the environment is Markovian which it is not.

6. **Identify and Explain differences between Forward and Inverse Models in Robot control systems.**

Forward model control systems take the input of a motor command to the plant and output a predicted position of the body. In general a forward model is a representation of the future state of a system i.e Given the current state of the motors and the intended action, a forward model allows the system to predict its future state. For example, for a robot manipulator, Forward Models predict the position of the end effector given the current joint angles. Advantage of Forward control theory based techniques is that they can provide necessary and sufficient conditions for a system to be exactly linearizable. This control law utilizes Forward Dynamics of the system. So,in forward models, the next state of the system only depends on the current state and action or previous states and actions.

Inverse models use the desired and actual position of the body as inputs to estimate the necessary motor commands which would transform the current position into the desired one.Inverse models allow the system to determine the motor commands necessary to achieve a desired state. i.e Inverse Models predict the joint angles required to reach a desired end effector position. This control law utilizes the

Inverse Dynamics of the system. So, In Inverse Models, the current action depends on future states. This approach is more intuitive because meaning of the variables remains intact. Also, if for a system with non linear dynamics, if we couldn't find a solution with forward control, we can always try and find a solution using Inverse Control techniques and for a robot an inverse mapping always exists and thus is useful.

7. **What was Demonstrated by Braitenberg with his Vehicles and explain why this was significant in 3 different perspectives.**

   In essence, Braitenberg through his Vehicles Demonstrates that Simple Robots/machines too simple in fact to be interesting from the point of view of Engineering can perform Complex tasks. Simple Machines can develop Love, aggression, Ego, Memory, will to make decisions, foresight, Knowledge, train of thoughts, machines with intelligence etc.

   Some of his Demonstrations where the thoughts of why are:

   1.Synthesis is easier than analysis in the sense that it is not possible to guess what is happening inside the machine without knowing what is inside the machine as there will be many different mechanisms that exhibit the identical behaviour. One of his vehicles had no conscious engineering at all and analyzing this type of systems gives a feeling of supernatural existence.

   2. Realization of an idea is much less important than the Idea Itself in the sense that an idea can be implemented in many different complex ways and there is also or might be a simple way to do it. - Let the problem of the mind, Dissolve in your mind.

   3. The Ideas that were introduced unknowingly may become wide spread in the long run. This comes from an analogy beautiful, marvelous machines can be made out of a simple trick where analysis doesn't make sense.

   One of the perspective is yes, simple machines are powerful and can be created and are enough to perform a Complex tasks. For example, when we look at an organism say frog and it sees a snake it runs away from it, but stays closer to snake homes in search of food. It look a complex task but the simple Vehicle exhibited this phenomenon. Say, we want to incorporate multitude of behaviour such as staying in the oxygen and running away from high temperatures, a vehicle with multiple sensors is able to achieve this. It is hard to comprehend that simple ideas and process produce complex tasks but all things that we cant comprehend are not false. When we try to build a complex machine to perform a task it shows our lack of understanding of the fundamental concepts to solve the problem. if we can reason every minute detail we could construct a simple machine to solve the task by having a simple idea. Another example is how does computers perform computations when there is an overflow? but a simple machine is able to perform the computations without being able to comprehend it. So, our idea of machines that machines can handle what they can handle might be wrong. we can handle machines differently such that it can perform the complex task without complex ideas in it. If we put a Thresholding behaviours to our inputs and we have a range of sensors we might not be able to comprehend what the logic of the machine is but it still works the way we want it to. its just not comprehensible. One of the analogy is, when we try to prove anything Mathematically, every other cross terms or terms that we cant reason nicely cancels out in the theorem and we would get the desired result. Is it because we wanted it to cancel out or the solution is that simple? may be there is a Single simple theory that unifies all the physical process and we are just not able to comprehend it or our brains are not made to comprehend just like how we couldn't imagine more than 3 dimensions which we know can exist mathematically. Believing Simple is true is the way forward to achieve complex tasks because more problems we can solve by simple machines the more complex tasks can be achieved that much more simply.

   The other perspective may be on the lines that, abstraction is the power of human mind. Simple vehicles might perform well for a particular task but might not be able to adapt to other things. Also, performing complex tasks with simple machines might be the ideas which we can think of a reasoning. while constructing the vehicles we are defining its architecture and the process of how it should work. It also responds in a way which we dont intend to but this always happens in any fundamental machine or a biological organism. Sometimes organisms need more fine tuning and it eventually comes and behaves as the way we expect similarly with the machines. I want to use an analogy here of Occam's Razor which states simplest model is better. But the No free lunch theorem Contradicts it. It says there is no one model and all models perform equally well when they are averaged over all possible problems. In order to find a right algorithm for a particular problem you can choose the simplest model but there will always be problems which one simple model couldn't generalize to. So, the ability to keep this memory demonstrated by Braitenberg vehicles is a significant aspect.

The other perspective may be there is little bit of ideology and perspective in designing machines. we design a machine to perform a task that is given and we also expect it to behave as we want to. To make the machine do as we expect it to we need to control every other aspect that is controlling the machine and this would require doing complex engineering. Knowing what to expect out of a machine is crucial So may be the simple model worked for a particular case because of some faulty generalization of the simple machine but may not be able to behave as we expect it to. The train of thoughts, the idea of getting ideas for simple machines , the free will these can be incorporated into simple machines but these also should occur with our expectations. This is a significant aspect demonstrated by Braitenberg.

8. **Identify 3 additional influential Pioneers in the development of Behaviour-based Robotics discussed in class, and explain how their approaches and methodologies differed.**

Traditional AI relies on a centralised world model for verifying sensory information and generating actions in the world.To overcome the problems encountered in Traditional AI when designing real robotic systems, Brooks proposed a completely different methodology. He refused the centralized symbolic world model and proposed a decentralized set of simple modules which reacted more rapidly to environmental changes. Since then many different approaches to behavioural robotics are developed and mostly differ in their architecture.

The Subsumption architecture was designed by Rodney Brooks in the 1980s at the Massachusetts Institute of Technology. His work opened the field of Behaviour based Robotics. One of the principles of the Subsumption architecture is independence from the layers. Individual layers work on individual goals concurrently and asynchronously and all the layers have direct access to the sensory information. Layers are organised hierarchically allowing higher layers to inhibit or suppress signals from lower layers. This hierarchy of layers with the suppression and inhibition constitute the Competitive coordination. Advantages of the Subsumption approach are robustness, modularity and simplicity of tuning the behaviours.

Action Selection Dynamics (ASD) is an architectural approach which uses a dynamic mechanism for behaviour (or action) selection. Pattie Maes from the AILab at MIT developed it. Action Selection Dynamics uses a network of nodes to implement the control architecture. Where each node is a behaviour and only one behaviour can be active. Hence the competitive coordination similar to Subsumption architecture and advantages are the same.

Motor dynamic mechanism for behaviour (or action) selection. developed by Ronald Arkin at Georgia Institute of Technology. Arkin proposed Motor Schemas as a new methodology of Behaviour-based Robotics. It assumes motor schema based decision making approach.Here, Motor Schema is a basic unit of behaviour from which complex behaviours are produced. Motor schema react proportionally to sensory information. Here, the cooperation method is cooperative and is a vector summation of all motor schema operators. These vectors are produced using the potential field methods. The advantage of this approach is its simplicity and ease of implementation. It also produces Optimized outputs since its output is summation of all the outputs. Also, similar to subsumption architecture sequential step algorithm is not used.

Schall, Stephan is another pioneer who used imitation Learning for producing desired Behaviours. In this, the behaviours are produced by imitation which is a type of Robot Shaping or reinforcement learning. RL/Robot Shaping is a process where we systematically reinforce successive approximations with positive or negative feedback until a desired behaviour is acquired. Here, the training is difficult and time consuming.

Mark Tilden proposed a style of robotics that primarily uses simple analog circuits, such as comparators, instead of a microprocessor in order to produce an unusually simple design. While not as flexible as microprocessor based robotics, can be robust and efficient in performing the task for which it was designed. It differed in the aspect of trying to reduce computations.

All these styles have more in common that they all used a bottom up approach but differed in the aspects of how they implemented that approach. In all these approaches Valentino Braitenbergs Simple is Better philosophy is embodied.

9. **Give 5 Concrete Examples on how evolutionary computation can be used to develop useful robot behaviours.**

Evolutionary computation has a family of algorithms for global optimization inspired by biological evolution. These algorithms can be categorized into Genetic Algorithms, Genetic Programs, Evolutionary Strategies, Evolutionary Programming and Learning Classifier Systems.

Each of these algorithms have their own methodologies of selecting an individual from the population, creating a cross-over of them, mutating them and calculating the fitness of each individual to generate future generations.

Collision - Free Navigation of vehicles: Collision-free navigation of vehicles can be achieved with the help of evolutionary methods. In mobile robots one of the principle problems is to create a system which is well adapted to its operating environment. Genetic Algorithms is very useful for adaptation and hence can be a solution for this. The basic consept of GA is that individuals within a population which are better adapted to environment can reproduce more than individuals which are not. Using this Darwinian philosophy we can train an Autonomous robot and improve its generations. The vehicle is equipped with various sensors which take input information from its surroundings and learn from that information. As the principle of Evolutionary methods follow they need to be trained at every step till the vehicle can start to navigate on its own. So, the task is broken down for each generation. For example, the first generation of training would be to detect any surface in front of them. Then, it will be trained to sense the distance between the vehicle and the surface. At each step, fitness is calculated which helps in determining the next generation. In this manner, at each step it evolves and learns which will help in reaching the goal i.e to navigate without crashing into anything.

Homing for battery charge: Battery charging for autonomous vehicles without human interaction is the desired behaviour. This can be looked at as the Braitenbergs Example of simple robot where when we connect a light sensor to a simple robot it exhibits an animal like behaviour by running away from light.(Neuroethology)Evolutionary algorithms can be used to train the autonomous vehicle to go back to its charging station a few seconds before the battery is completely drained. Here, the fitness function will be in terms of speed of the vehicle and the light sensor reading. The vehicle will move around in a confined space where a spot will be dedicated for the vehicle to charge. The vehicle will move around the confined space and will remember the spot for charging and a few seconds before the battery is about to be dead, the vehicle will go back to the charging spot to charge its batteries.

Active vision for vehicle navigation: In this example, the goal is to navigate a vehicle using vision information from a camera. The output of the vision system will be the movement of the camera and the wheels of the vehicle. Here, the feedback of the outputs are sent back to the input to decide upon the next behavior. It Involves Genetic evolution of a neural network driven robot.This example follows evolutionary strategy where the information is represented as strings or vectors of real-valued attributes. The Fitness function can be computed using Distance D covered in R number of races on M circuits. Then the population of high Fitness value is nurtured.

Karl Sims - Evolved Virtual Creatures: Developing control systems for robots to do complex and Dexterous tasks is very hard. We need a lot of sensors, processor speed etc. But what if the robot itself learns a control strategy? This was demonstrated by Karl Sims in his work. In this, one of the robot evolved to swim was able to adapt to new conditions of the ground and came up with a way to move on land. Another robot was accurately swimming to the point of light by itself. This shows that given enough time and accessories robots can exhibit far higher capabilities.

Example 3. Ant problem: Here, the problem is the ant has to eat all the food available in the map. The fitness function for this program is to maximize the food eaten. Control programs can be represented as trees. Mutation and crossover acting on trees.

10. **Describe in detail the use of shaping for creating a behaviour based robot controller.**

Behavior-based robotics (BBR) or behavioral robotics is an approach in robotics that focuses on robots that are able to exhibit complex-appearing behaviors despite little internal variable state to model its immediate environment, mostly gradually correcting its actions via sensory-motor links. It is based on Braitenberg demonstrating how simply wired sensor/motor connections can result in some complex-appearing behaviors such as fear and love. The idea is that complex behaviours can be built bottom - up from a set of simple responses. This is where Robot Shaping comes into play. Many different Architectures are in place for a robot to produce complex behaviours and Robot Shaping is one of those methods to produce these behaviours.

A Behaviour is a product of the interaction between an agent and its environment. The possible behaviours are determined by the structure and the dynamics of both the agent and the environment and by the interface between the two.

We can distinguish between different types of behaviours based on the interface as: Stimulus-Response, Dynamic Behaviour and the other is Shaping. Most learning schemes uses set of steps to solve problems, it follows a path based on internal representations of events compared to the behavior-based approach. Rather than use preset calculations to tackle a situation, behavior-based robotics relies on adaptability. Shaping is a tool that is used to teach these behaviours for BBR.

Robot Shaping is a process where we systematically reinforce successive approximations with positive or negative feedback until a desired behaviour is acquired. It adopts the reward system of reinforcement learning but the reward is presented by a trainer unlike the state in RL. The reward structure is entirely decided by the trainer and it must be in such a way that the model should not stop at an intermediate behaviour point.

TASK: A robot must move in a straight line, collect, and return a ball.
STEPS:
I. Fetch → Move towards the ball → rewarded
II. Fetch → Move towards the ball (reward withheld) → locate the ball → rewarded
III. Fetch → Move towards the ball → locate the ball (reward withheld) → pick the ball → rewarded
IV. Fetch → Move towards the ball → locate the ball → pick the ball (reward withheld) → hold the ball firmly → rewarded
V. Fetch → Move towards the ball → locate the ball → pick the ball → hold the ball firmly (reward withheld) → If the ball is dropped → penalty
VI. Fetch → Move towards the ball → locate the ball → pick the ball → hold the ball firmly (reward withheld) → Move back to initial position → rewarded

As seen from the example, a task can be broken into simpler tasks and training the robot to perform these simple tasks eventually makes the robot learn complex Behaviour.This way we can make the robot learn the behaviours we desire(Target Behaviour). If we can reinforce enough behaviours in different environments the robot might take an average of all the situations and act accordingly, and thus exhibit intelligence. This is the approach of BBR. Thus, robot shaping is a crucial element in a Behaviour Based Controller where we can train the robot with our desired behaviours instead of trying to evolve by itself.