MSU College of Engineering Request

What are you up to?

Report your job, internship or co-op employment for a chance to win an amazon gift cardl

SCAN HERE

- Today M7 – Machine Learning
- Next class – TT4 and a select Advanced Topic
  - Printed Circuit Board (PCB) Layout advanced design guidelines
- Questions?

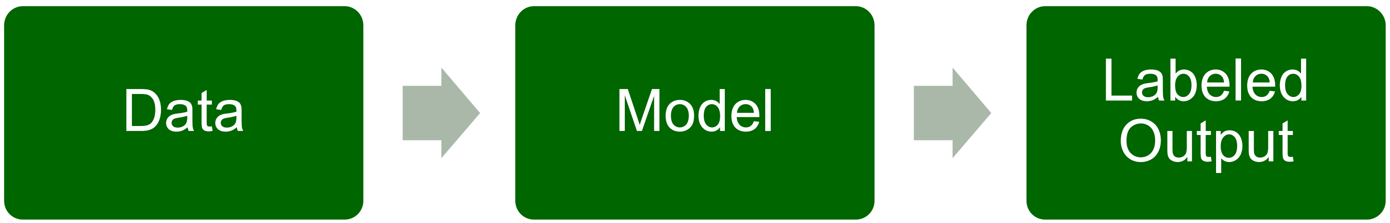
- Lab 6
  - Lab6B **optional** for **extra** 40 points
  - Lab6A and 6B worksheets are due Wed April 23rd at 11:59p
- Please complete your class survey: https://msu.bluera.com/msu/

Milestones: Important events:

- *Wed Apr 23, TT4 (April 8th signal processing, M7) , Open notes, book*

Supervised Learning (review)

- Trained on labeled datasets
- Learn mapping from inputs to outputs
- Predict outcomes for new, unseen data

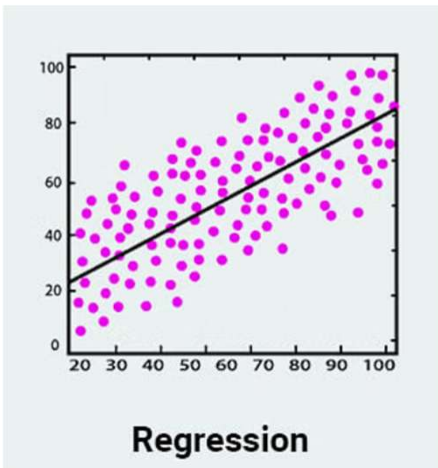


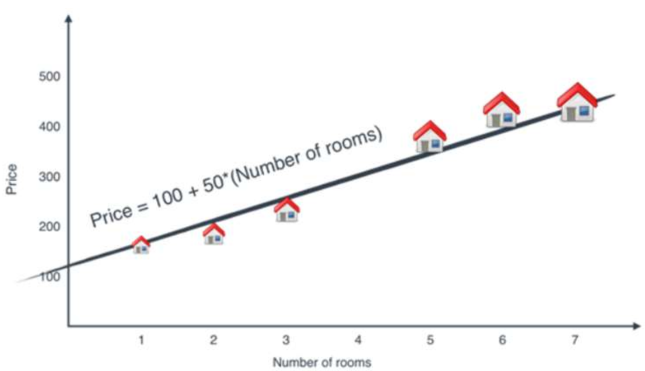
- Applications of Supervised Learning
  - Spam Detection
  - Image Recognition
  - Voice Assistants
  - Predictive Maintenance
  - Credit Scoring

Regression vs. Classification

Regression

- **Definition:** Regression is a type of supervised learning used to predict continuous numerical values.
- **Goal:** The aim is to model the relationship between input features and the output variable.
- **Examples:** Predicting house prices, forecasting stock prices, estimating temperature changes.
- **Output:** Continuous values (e.g., 3.5, 150.75).





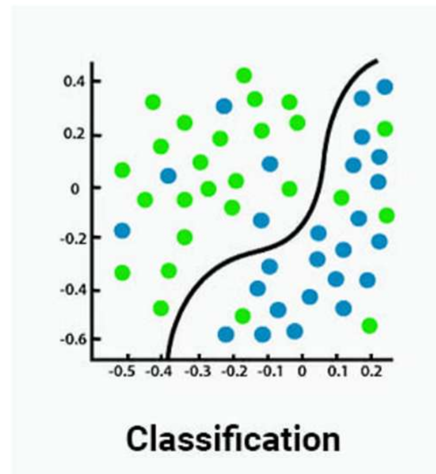
Classification

Definition: Classification is a type of supervised learning used to predict categorical labels.

Goal: The aim is to assign input data into predefined categories or classes.

Examples: Email spam detection, image recognition, medical diagnosis.

Output: Discrete labels (e.g., spam/not spam, cat/dog).



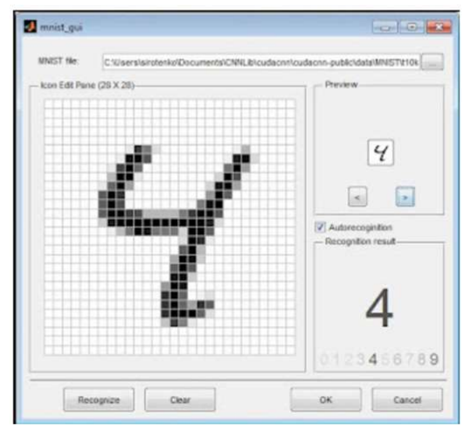


Image Source: https://kindsonthegenius.com/blog/what-is-the-difference-between-classification-and-regression/

Overfitting. Vs. Underfitting

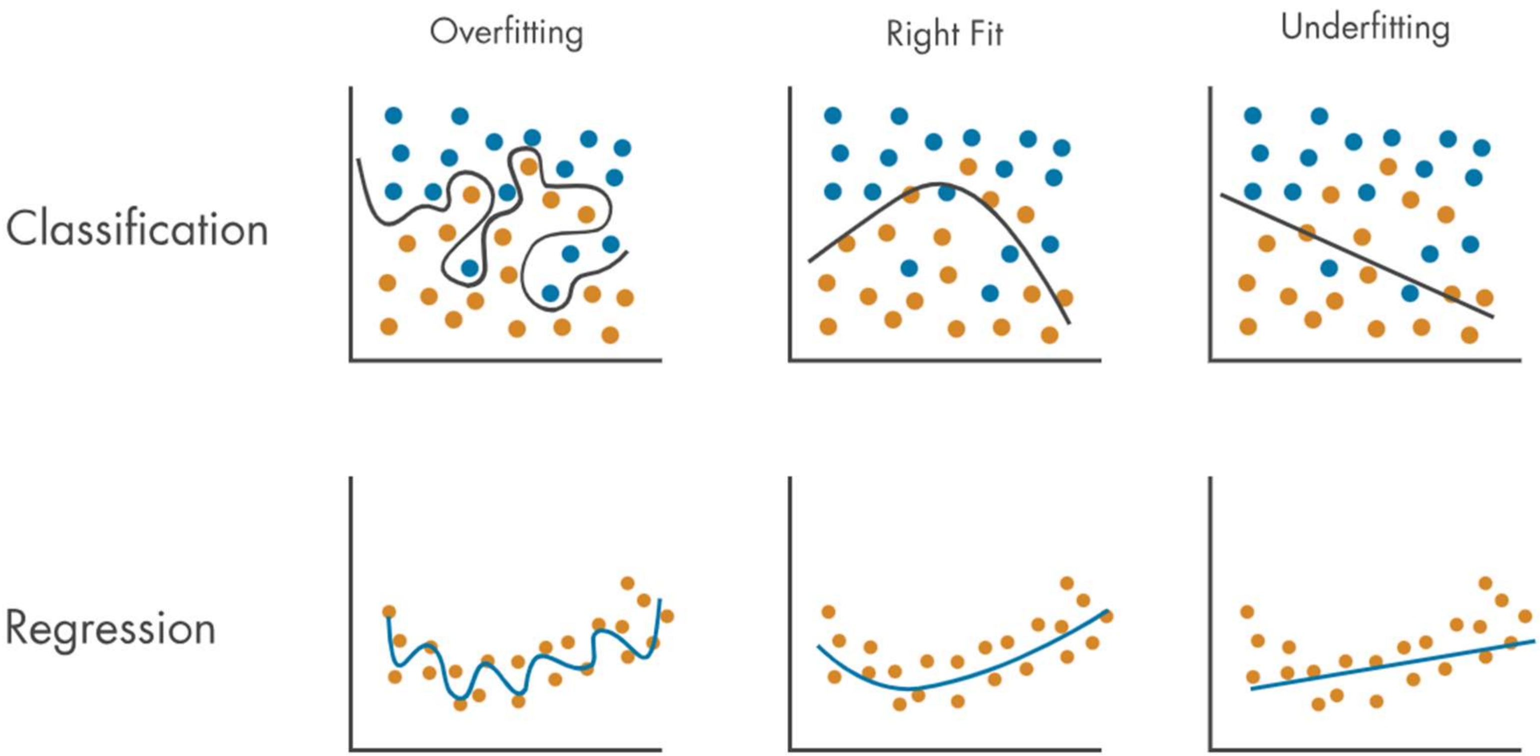


Image Source: MathWorks

Error	Overfitting	Right Fit	Underfitting
Training	Low	Low	High
Test	High	Low	High

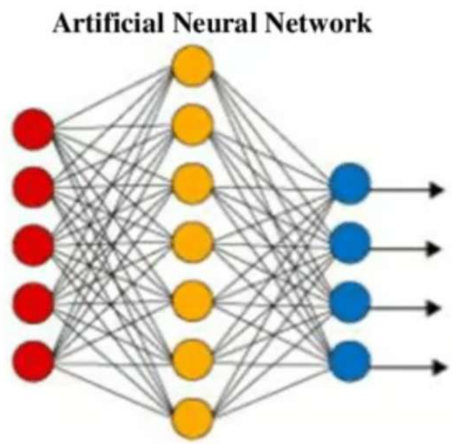
Neural Networks

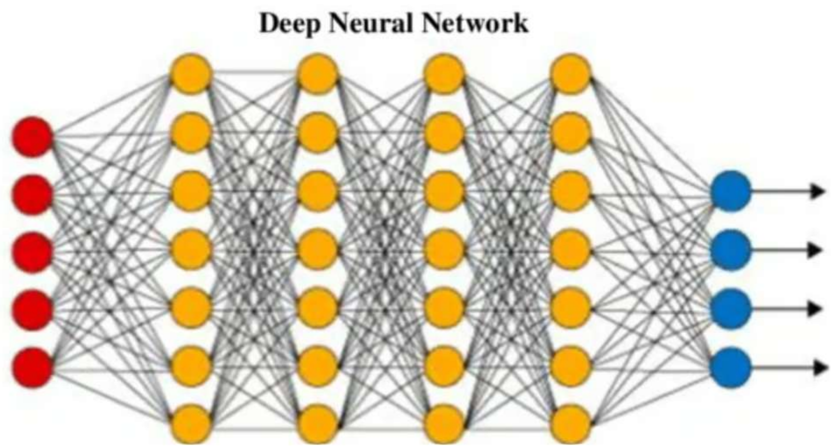
Artificial Neural Networks

- Made of layers: input, hidden, and output
- Each layer learns patterns from data
  - Set connection weights during ***Training***

Deep Neural Networks

- Can understand complex data
- Useful for images, text, and more





How Training Works?

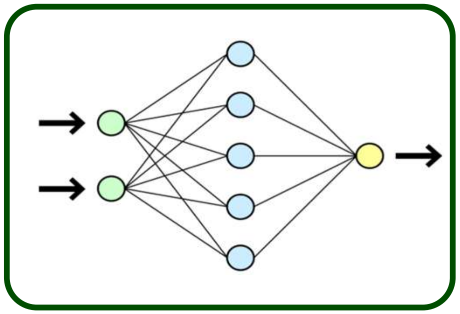
- Give the model examples
- Compare predictions to the right answer
- Adjust the model to improve

Input → Prediction → Error → Update

What's a Loss Function?

- A loss function measures how well a neural network predicts the target output
- It calculates the difference between the predicted output and the actual label
- A key component in training — guides the model to improve over time
- Lower loss = better model

Input Data (x)



Predicted Output (y_pred)

Loss=L (y_pred-y)

Target Value (y)

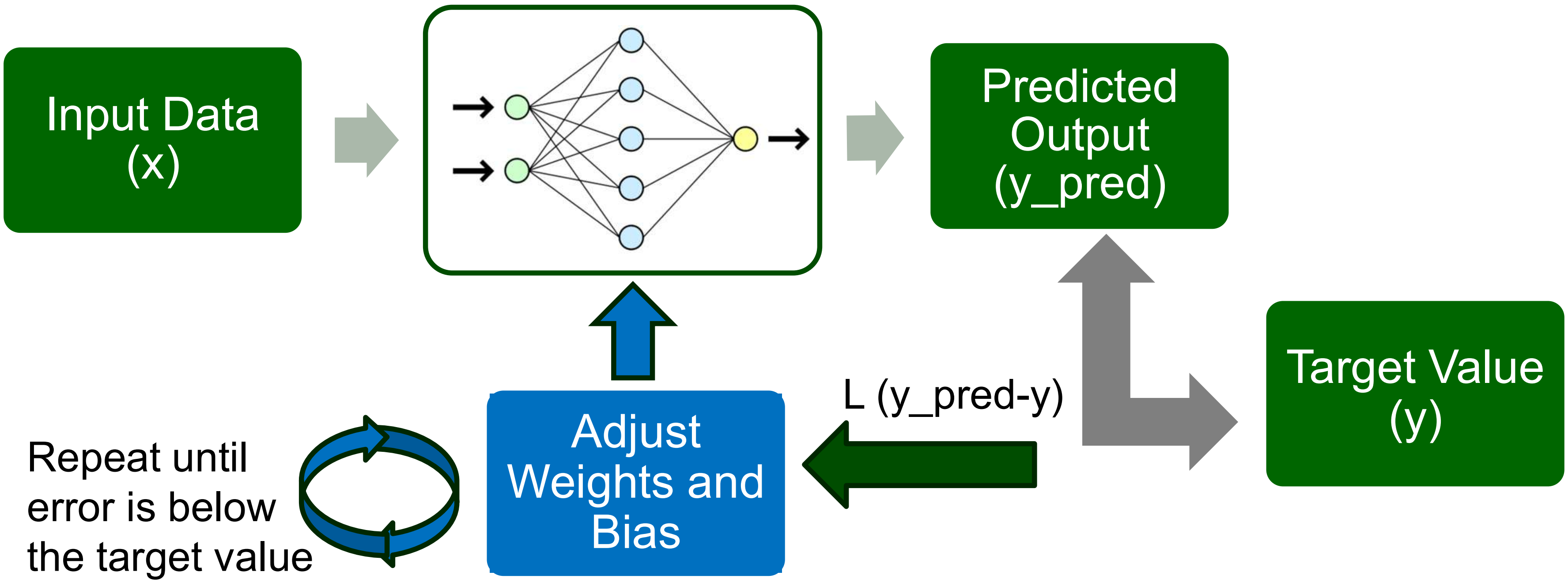
Common Types of Loss Functions

- **Mean Squared Error (MSE) –** used for regression problems $MSE: \frac{1}{n}\sum(y_{true} - y_{pred})^2$
- **Cross-Entropy Loss –** used for classification tasks $Cross - Entropy\ Loss: -\sum y\log(\widehat{y})$
- Choice depends on problem type and model output

y: true label (0 or 1 in binary)
$\hat{y}$: **predicted probability** that the input belongs to class
Range: 0< $\hat{y}$<1

Why Do We Need a Loss Function?

- Helps the network learn by quantifying error
- Acts as the objective to minimize during training
- Used in backpropagation to update weights

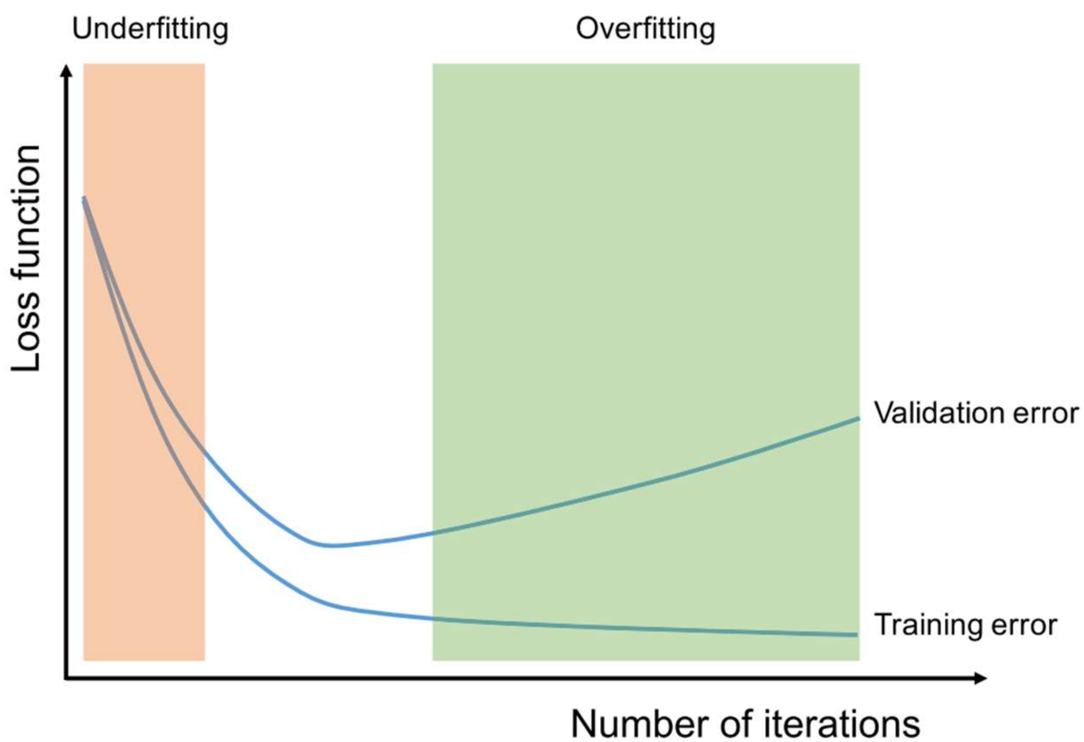


Loss vs Accuracy

- **Accuracy** tells how many predictions were correct
- **Loss** provides more nuanced information about how wrong predictions were
- Important during training: **even when accuracy plateaus, loss may still improve**

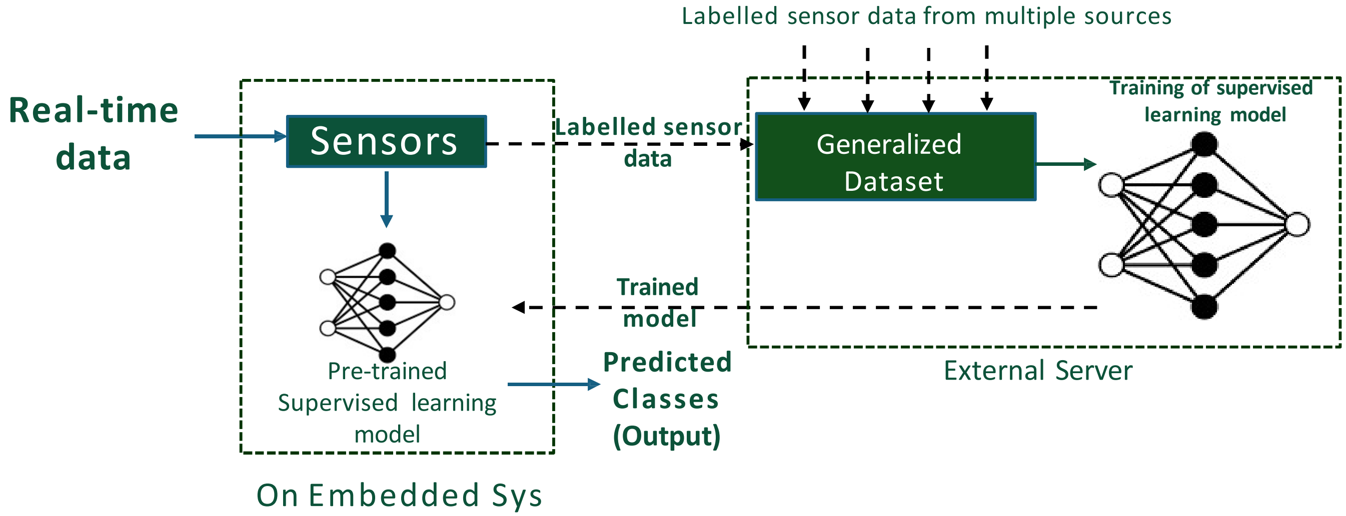
Training: Gradient Descent

- **Gradient descent** is an iterative optimization algorithm that updates learnable parameters (e.g., weights) to minimize a loss function by moving in the direction opposite to the gradient.
- The gradient is the partial derivative of the loss with respect to each parameter, and updates are scaled by a **learning rate (α)**, which is a crucial hyperparameter.
- **Stochastic Gradient Descent (SGD)** uses mini-batches of the training data to compute gradients and update parameters, balancing computational efficiency and convergence speed.



Source: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9

Supervised Learning Approaches

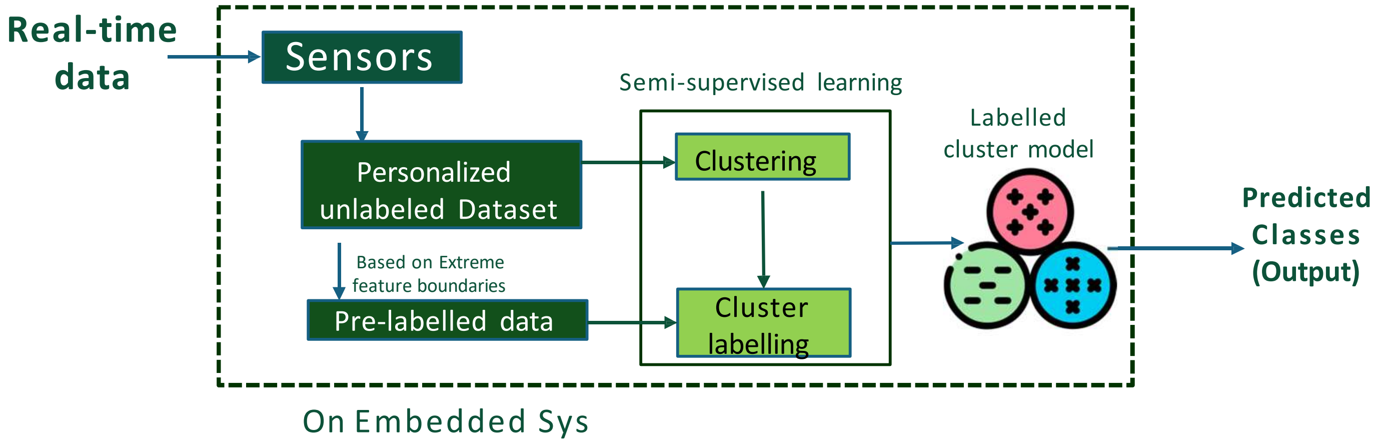


<u>*Supervised, pre-trained machine learning models*</u> are:

- Loaded into the embedded devices which infer decisions from real-time data.
  - *Lack of availability of enough pre-labelled data to pre-train ML models.*
- For user-specific applications are trained on the basis of a <u>*generalized dataset.*</u>
  - *ML models trained on generalized datasets struggle with applications requiring person-specific nuances.*
- Requires data collection from various subjects, external processing for training, and model transfer to the user's device for real-time inference.
  - *User's data privacy is compromised.*
  - *Added expenses for communication resources.*
- Involves a computationally expensive training process
  - *Embedded devices are generally scarce of computing resources(power, memory, etc).*

Semi-Supervised Learning for Embedded Sys

On-device semi-supervised learning using sparsely labelled datapoints:

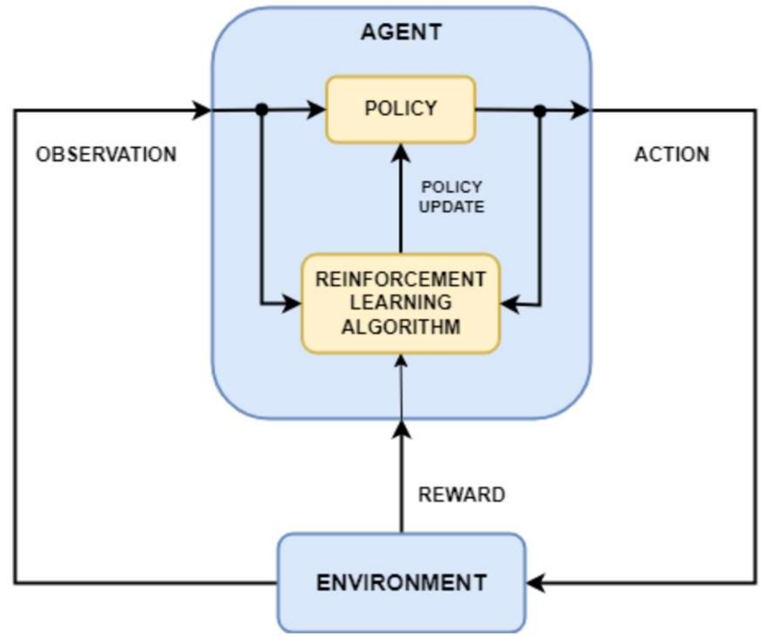


An **on-device semi-supervised learning** model using sparsely labeled data is proposed to address:

- Suited for scenarios ***lacking enough pre-labeled data*** to pre-train supervised models.
- The learning model is ***person-specific (personalized)*** by training on data directly from the user.
- Learning occurs ***on the same device*** that collects the user data, ***eliminating communication costs.***
  - Ensuring **data privacy**.
- Designed to consume ***minimal computational resources***, making it ideal for **lightweight devices** like wearables.

What is Reinforcement Learning?

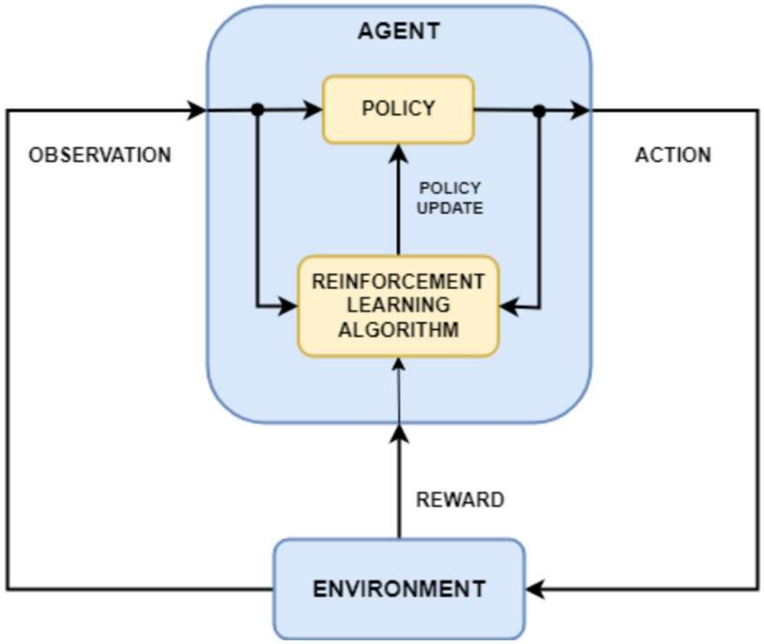
- The goal of reinforcement learning is to train an agent to complete a task within an unknown environment.
- The agent receives observations and a reward from the environment and sends actions to the environment.
- The reward is a measure of how successful an action is with respect to completing the task goal.
- **Policy** is a mapping that selects actions based on the observations from the environment.
  - *The policy is typically a function approximator with tunable parameters, such as a deep neural network.*
- **Learning algorithm** continuously updates the policy parameters based on the actions, observations, and reward.
  - *The goal of the learning algorithm is to find an optimal policy that maximizes the cumulative reward received during the task.*



Reinforcement learning involves an agent learning the optimal behavior through repeated trial-and-error interactions with the environment without human involvement.

Example: Automated Vehicle Parking

- **Agent:** Vehicle computer aims to park correctly using trial-and-error
- **Observations:** Data from sensors (camera, GPS, lidar, etc.)
- **Actions:** Steering, braking, and acceleration commands.
- **Environment:** Everything outside the agent (vehicle dynamics, road, sensors)
- **Reward:** 1 for successful parking, 0 otherwise.
- **Learning:** The agent updates its action strategy (policy) after each trial to maximize rewards and learn optimal parking behavior



Reinforcement vs. Supervised/Unsupervised Learning

Aspect	Reinforcement Learning	Supervised Learning	Unsupervised Learning
Learning from	Rewards	Labels	Data patterns
Feedback type	Delayed	Instantaneous	None
Exploration vs Exploitation	Both	Exploitation	Exploration

Challenges in Reinforcement Learning

- Sample inefficiency
  - Reinforcement learning often requires a large number of interactions with the environment to learn effective policies.
  - This makes it computationally expensive and slow, especially in real-world applications like robotics or driving.
- Reward shaping
  - Carefully designing intermediate rewards can help guide learning, but poorly chosen rewards may lead to unintended behavior.
  - It requires domain knowledge and can introduce bias or obscure the original task goal.
- Credit assignment problem
  - It's often unclear which actions in a sequence led to a final outcome or reward.
  - This makes it difficult for the agent to learn which behaviors are beneficial or harmful.
- Stability and convergence
  - RL algorithms can be unstable due to non-stationary environments or policy updates.
  - Convergence to an optimal or even satisfactory policy is not guaranteed and can be sensitive to hyperparameters.

MSU College of Engineering Request