Semi-supervised Learning, Reinforcement Learning and Introduction to Deep Learning

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
- Learning from Positive and Unlabeled Data

Semi-supervised Learning [1]

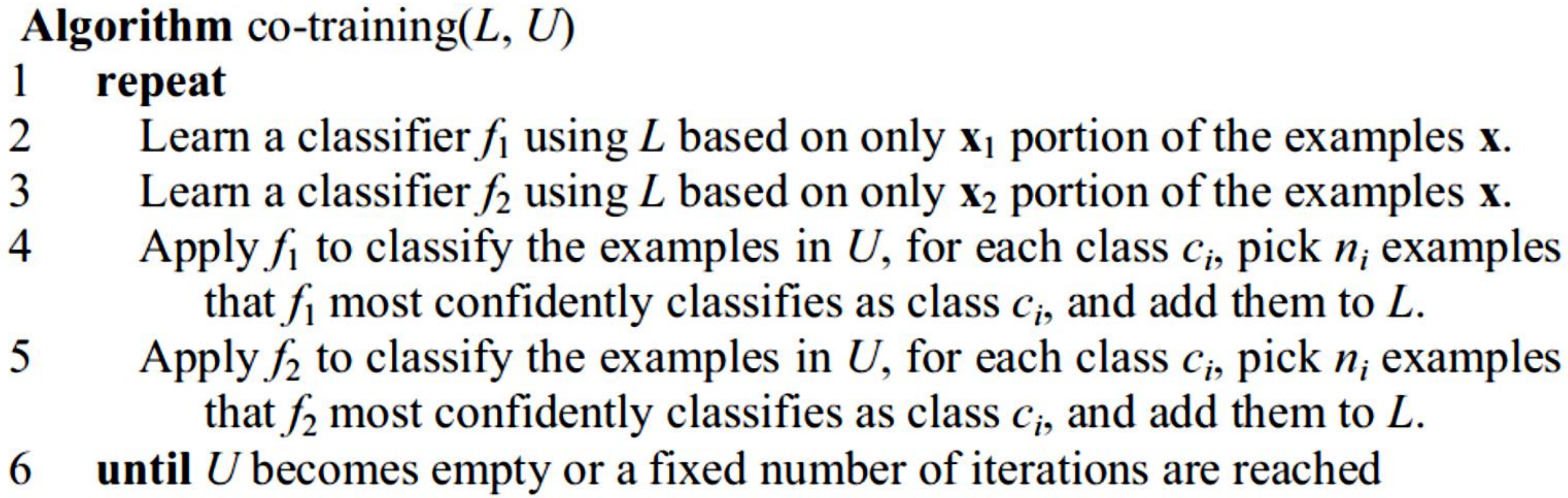
- Learning from Labeled and Unlabeled Data
  - Co-Training
  - Self-Training

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Co-Training
    - Co-training is one of the approaches to learning from labeled and unlabeled examples.
    - This approach assumes that the set of attributes (or features) in the data can be partitioned into two subsets and each of them is sufficient for learning the target classification function.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Co-Training



```
Algorithm co-training($L$, $U$)
repeat
    Learn a classifier $f_1$ using $L$ based on only $\mathbf{x}_1$ portion of the examples $\mathbf{x}$.
    Learn a classifier $f_2$ using $L$ based on only $\mathbf{x}_2$ portion of the examples $\mathbf{x}$.
    Apply $f_1$ to classify the examples in $U$, for each class $c_i$, pick $n_i$ examples
        that $f_1$ most confidently classifies as class $c_i$, and add them to $L$.
    Apply $f_2$ to classify the examples in $U$, for each class $c_i$, pick $n_i$ examples
        that $f_2$ most confidently classifies as class $c_i$, and add them to $L$.
until $U$ becomes empty or a fixed number of iterations are reached
```

- Note: In practice, we can set a different n_i for a different class c_i depending on class distributions. For example, if a data set has one third of class 1 examples and two thirds of class 2 examples, we can set n_1 = 1 and n_2 = 2.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Co-Training

```
Algorithm co-training($L$, $U$)
1   repeat
2       Learn a classifier $f_1$ using $L$ based on only $\mathbf{x}_1$ portion of the examples $\mathbf{x}$.
3       Learn a classifier $f_2$ using $L$ based on only $\mathbf{x}_2$ portion of the examples $\mathbf{x}$.
4       Apply $f_1$ to classify the examples in $U$, for each class $c_i$, pick $n_i$ examples
            that $f_1$ most confidently classifies as class $c_i$, and add them to $L$.
5       Apply $f_2$ to classify the examples in $U$, for each class $c_i$, pick $n_i$ examples
            that $f_2$ most confidently classifies as class $c_i$, and add them to $L$.
6   until $U$ becomes empty or a fixed number of iterations are reached
```

- When the co-training algorithm ends, it returns two classifiers. At classification time, for each test example the two classifiers are applied separately and their scores are combined to decide the class.
- For naïve Bayesian classifiers, we multiply the two probability scores, i.e., $\Pr(c_j|\mathbf{x}) = \Pr(c_j|\mathbf{x}_1)\Pr(c_j|\mathbf{x}_2)$

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Co-Training
    - The first assumption is that the example distribution is compatible with the target functions; that is, for most examples, the target classification functions over the feature sets predict the same label. In other words, if f denotes the combined classifier, f1 denotes the classifier learned from X1, f2 denotes the classifier learned from X2 and c is the actual class label of example x, then f(x) = f1(x1) = f2(x2) = c for most examples.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Co-Training
    - The first assumption is that the example distribution is compatible with the target functions; that is, for most examples, the target classification functions over the feature sets predict the same label. In other words, if f denotes the combined classifier, f1 denotes the classifier learned from X1, f2 denotes the classifier learned from X2 and c is the actual class label of example x, then f(x) = f1(x1) = f2(x2) = c for most examples.

    - The second assumption is that the features in one set of an example are conditionally independent of the features in the other set, given the class of the example. In the case of movie sentiment classification, this assumes that the likes and dislikes received by the movie are not related to the other features of the movie, except through the class/sentiment of the movie. This is a somewhat unrealistic assumption in practice.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Self-Training
    - Self-training, which is similar to co-training, is another method for LU learning.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Self-Training
    - Self-training, which is similar to co-training, is another method for LU learning.
    - It is an incremental algorithm that does not use the split of features.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Self-Training
    - Self-training, which is similar to co-training, is another method for LU learning.
    - It is an incremental algorithm that does not use the split of features.
    - Initially, a classifier (e.g., naïve Bayesian classifier) is trained with the small labeled set considering all features.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Self-Training
    - Self-training, which is similar to co-training, is another method for LU learning.
    - It is an incremental algorithm that does not use the split of features.
    - Initially, a classifier (e.g., naïve Bayesian classifier) is trained with the small labeled set considering all features.
    - The classifier is then applied to classify the unlabeled set.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Self-Training
    - Self-training, which is similar to co-training, is another method for LU learning.
    - It is an incremental algorithm that does not use the split of features.
    - Initially, a classifier (e.g., naïve Bayesian classifier) is trained with the small labeled set considering all features.
    - The classifier is then applied to classify the unlabeled set.
    - Those most confidently classified (or unlabeled) documents of each class, together with their predicted class labels, are added to the labeled set.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Self-Training
    - Self-training, which is similar to co-training, is another method for LU learning.
    - It is an incremental algorithm that does not use the split of features.
    - Initially, a classifier (e.g., naïve Bayesian classifier) is trained with the small labeled set considering all features.
    - The classifier is then applied to classify the unlabeled set.
    - Those most confidently classified (or unlabeled) documents of each class, together with their predicted class labels, are added to the labeled set.
    - The classifier is then re-trained and the procedure is repeated.

Semi-supervised Learning [1]

- Learning from Labeled and Unlabeled Data
  - Self-Training
    - Self-training, which is similar to co-training, is another method for LU learning.
    - It is an incremental algorithm that does not use the split of features.
    - Initially, a classifier (e.g., naïve Bayesian classifier) is trained with the small labeled set considering all features.
    - The classifier is then applied to classify the unlabeled set.
    - Those most confidently classified (or unlabeled) documents of each class, together with their predicted class labels, are added to the labeled set.
    - The classifier is then re-trained and the procedure is repeated.
    - This process iterates until all the unlabelled documents are given class labels. The basic idea of this method is that the classifier uses its own predictions to teach itself.

Semi-supervised Learning [1]

- Learning from Positive and Unlabeled Data
  - Two-Step Approach
  - Direct Approach

Semi-supervised Learning [1]

- Learning from Positive and Unlabeled Data
  - Two-Step Approach
    - As its name suggests the two-step approach works in two steps:

    1. Identifying a set of reliable negative documents (denoted by RN) from the unlabeled set U.

    2. Building a classifier using P and RN. This step may apply an existing learning algorithm once or iteratively depending on the quality and the size of the RN set.

Semi-supervised Learning [1]

- Learning from Positive and Unlabeled Data
  - Two-Step Approach
    - NB Technique for Step 1:

      1. Assign each document in P the class label 1;
      2. Assign each document in U the class label -1;
      3. Build a NB classifier using P and U;
      4. Use the classifier to classify U. Those documents in U that are classified as negative form the reliable negative set RN.

Semi-supervised Learning [1]

- Learning from Positive and Unlabeled Data
  - Two-Step Approach
    - Second Step:
      - Run a learning algorithm (e.g., NB or SVM) using P and RN. The set of documents in U - RN is discarded.
      - This method works well if the reliable negative set RN is sufficiently large and contains mostly negative documents.

Reinforcement Learning

- What is Reinforcement Learning?
- Implementation Approaches of RL Algorithms
  - Model-based RL
  - Model-Free RL
    - Q-learning (Value-based)
    - Policy Optimization (Policy-based)

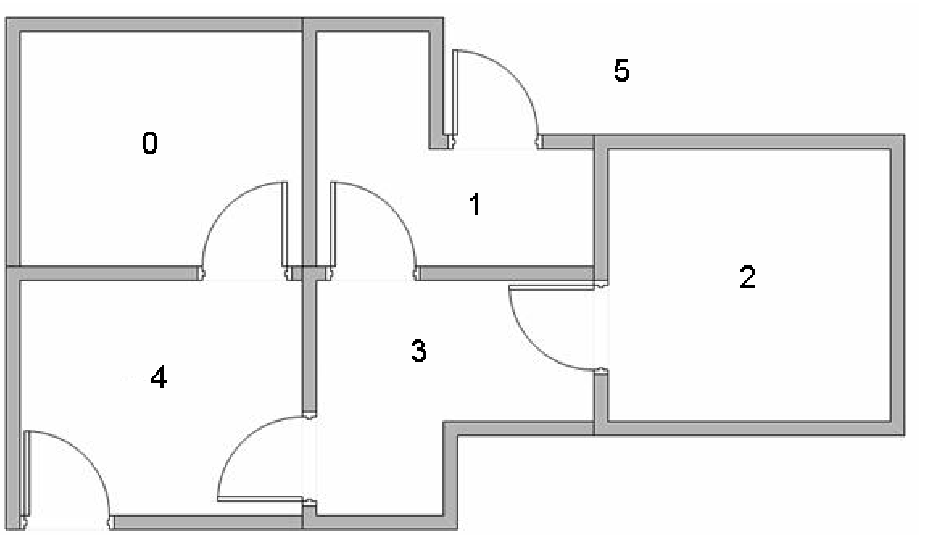
Reinforcement Learning

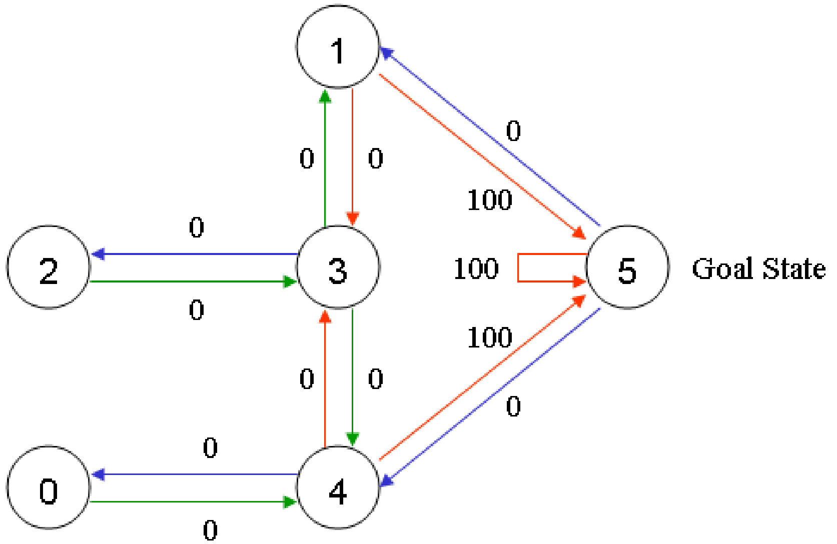
- Q-Learning Algorithm:
- The Q-Learning algorithm goes as follows:

```
1. Set the gamma parameter, and environment rewards in matrix R.
2. Initialize matrix Q to zero.
3. For each episode:
        Select a random initial state.
        Do While the goal state hasn't been reached.
            - Select one among all possible actions for the current state.
            - Using this possible action, consider going to the next state.
            - Get maximum Q value for this next state based on all
              possible actions.
            - Compute: Q(state, action) = R(state, action) + Gamma *
              Max[Q(next state, all actions)]
            - Set the next state as the current state.
        End Do
    End For
```

Reinforcement Learning [2]

- Q-Learning (Initial Setup)





$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

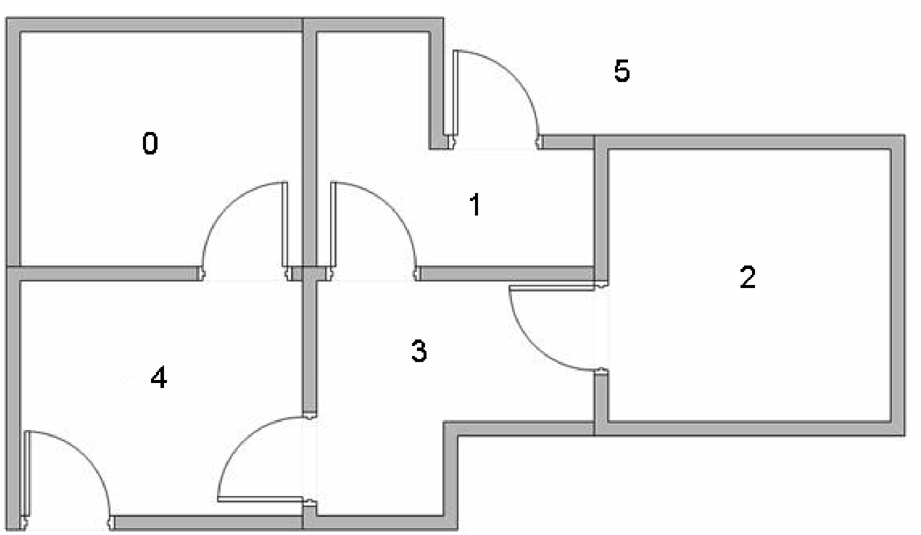
Action

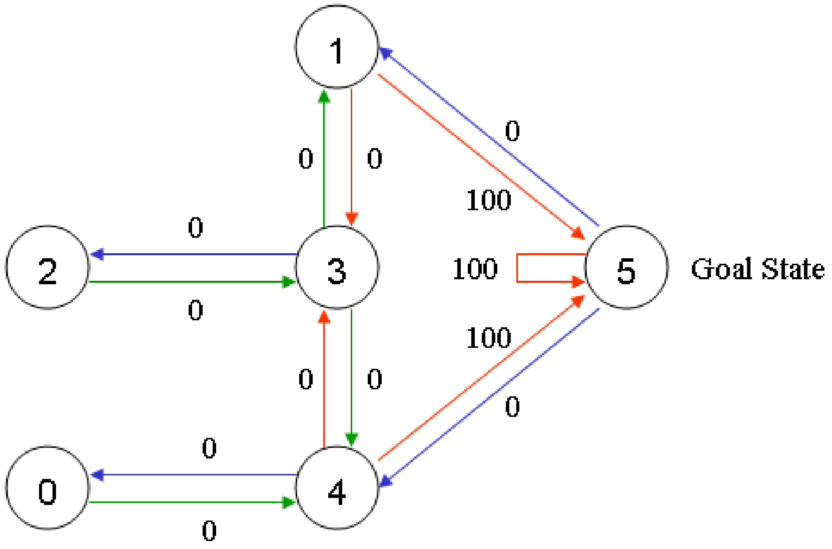
$$R = \begin{array}{c|cccccc} \text{State} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]

Reinforcement Learning [2]

➢ Q-Learning – Episode 1 (Gamma = 0.8, and the initial state as Room 1)





$$Q=\begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Action

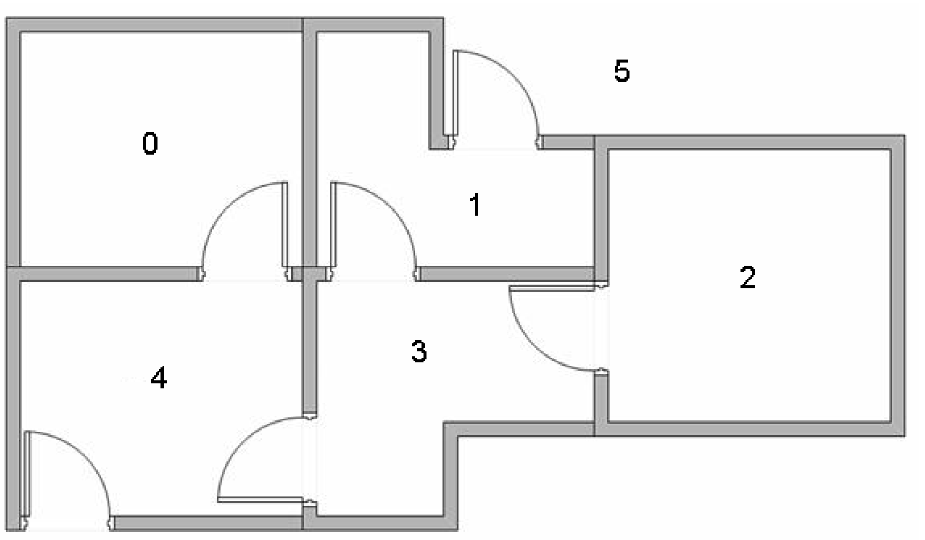
$$R=\begin{array}{c|cccccc} \text{State} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

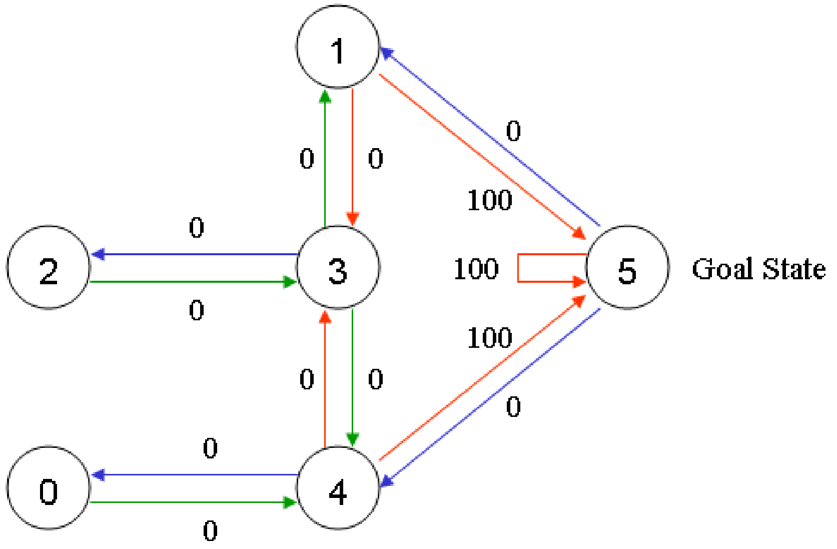
Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]

Q(1, 5) = R(1, 5) + 0.8 * Max[Q(5, 1), Q(5, 4), Q(5, 5)] = 100 + 0.8 * 0 = 100

Reinforcement Learning [2]

- Q-Learning – Episode 1 (Gamma = 0.8, and the initial state as Room 1)





$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Action

$$R = \begin{array}{c|cccccc} \text{State} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

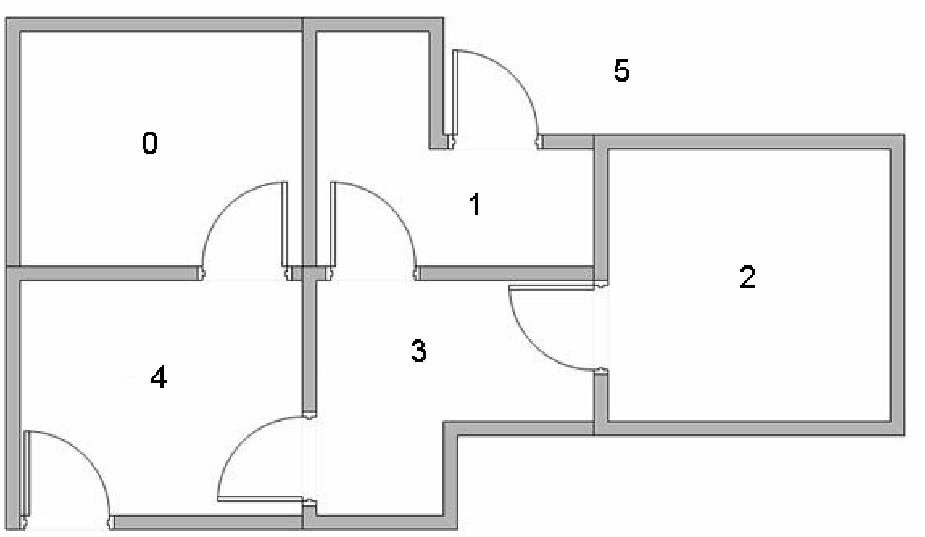
$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 100 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

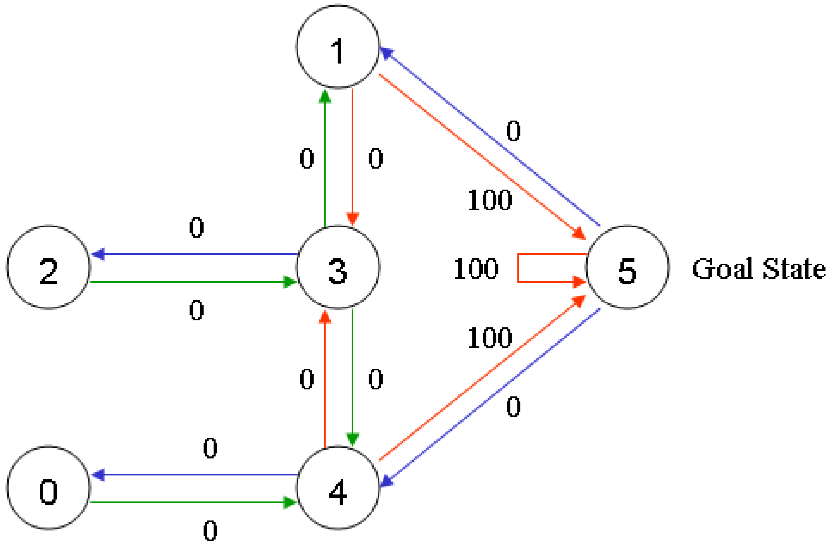
Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]

Q(1, 5) = R(1, 5) + 0.8 * Max[Q(5, 1), Q(5, 4), Q(5, 5)] = 100 + 0.8 * 0 = 100

Reinforcement Learning [2]

- Q-Learning - Episode 2 (Gamma = 0.8, and the initial state as Room 3)





$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 100 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Action

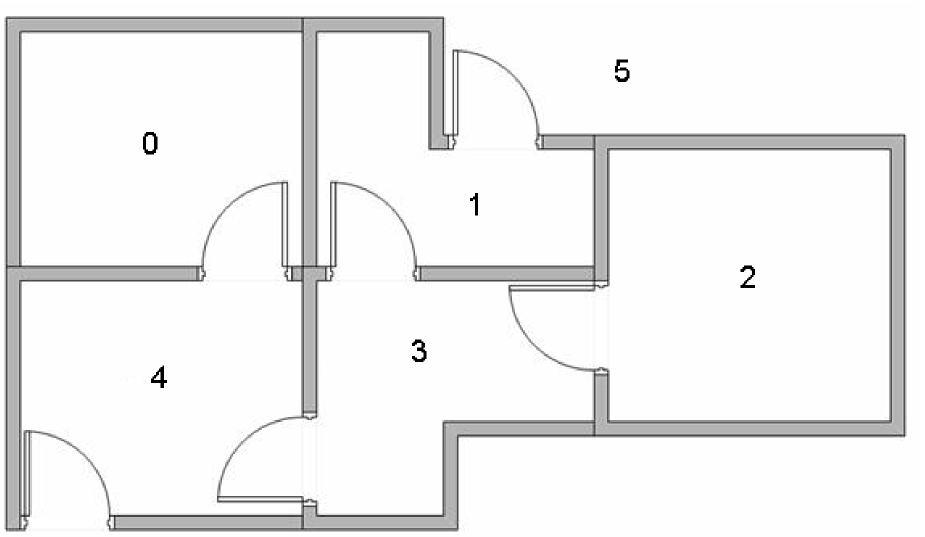
$$R = \begin{array}{c|cccccc} \text{State} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

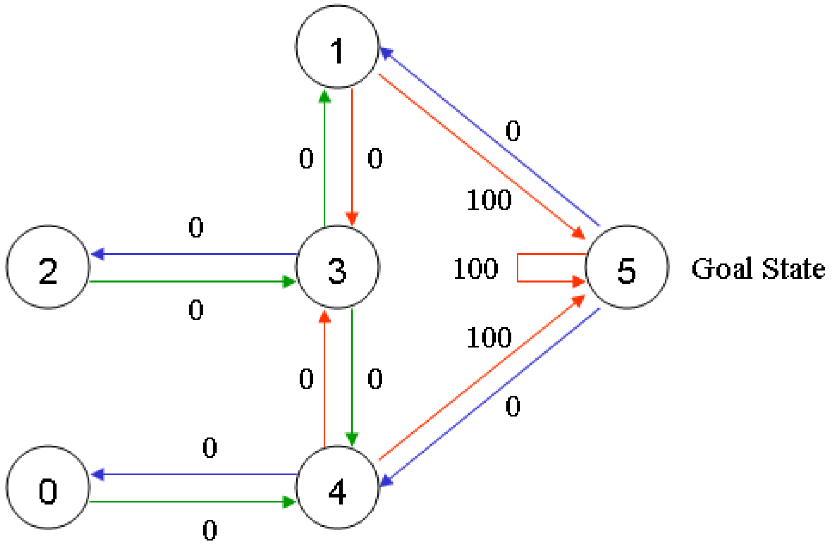
Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]

Q(3, 1) = R(3, 1) + 0.8 * Max[Q(1, 3), Q(1, 5)] = 0 + 0.8 * Max(0, 100) = 80

Reinforcement Learning [2]

- Q-Learning – Episode 2 (Gamma = 0.8, and the initial state as Room 3)





$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 100 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Action

$$R = \begin{array}{c|cccccc} \text{State} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

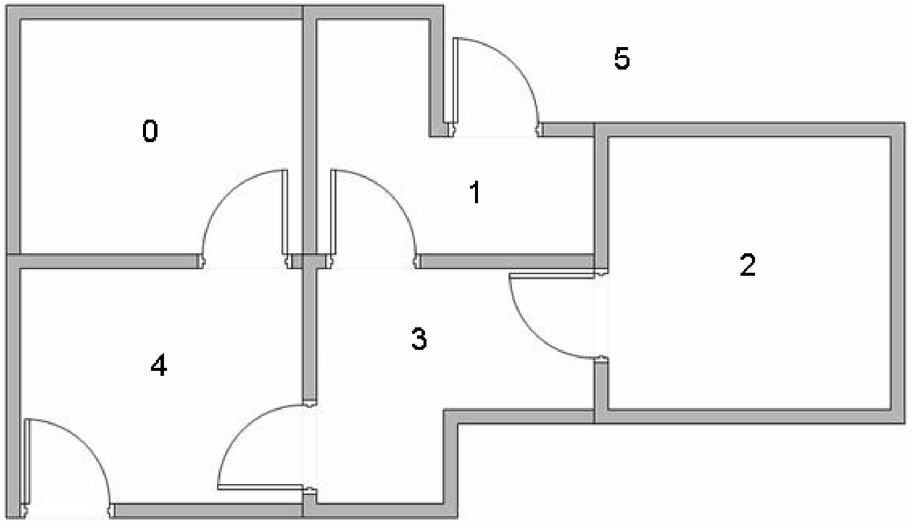
$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 100 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 80 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

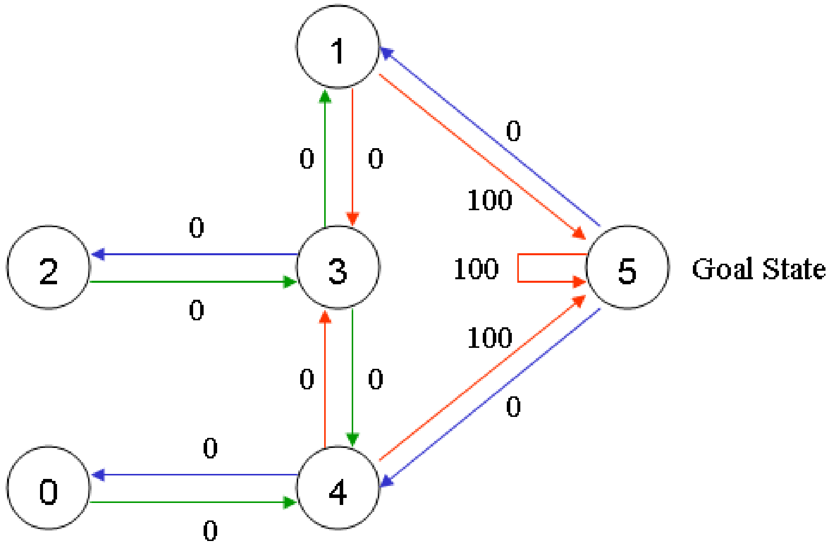
Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]

Q(3, 1) = R(3, 1) + 0.8 * Max[Q(1, 3), Q(1, 5)] = 0 + 0.8 * Max(0, 100) = 80

Reinforcement Learning [2]

- Q-Learning - Episode 2 Continue (Gamma = 0.8, and the current state is Room 1)





$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 100 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 80 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Action

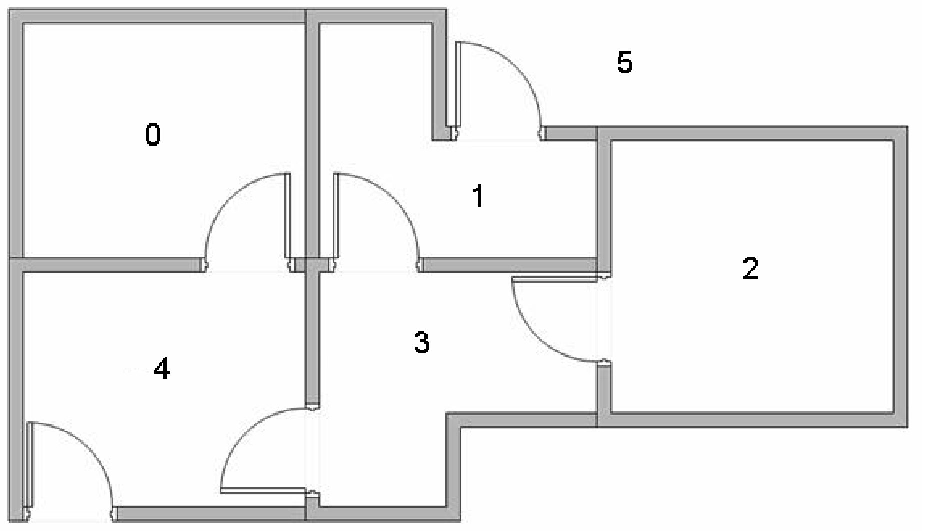
$$R = \begin{array}{c|cccccc} \text{State} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

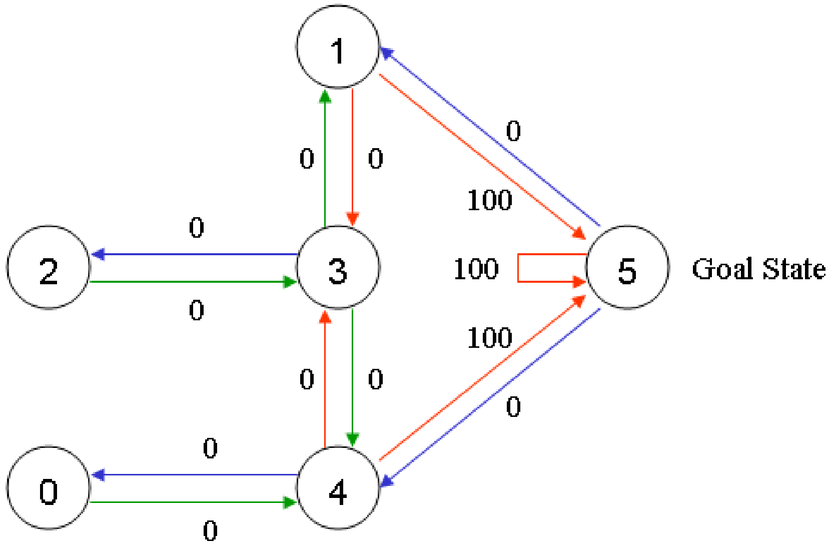
Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]

Q(1, 5) = R(1, 5) + 0.8 * Max[Q(5, 1), Q(5, 4), Q(5, 5)] = 100 + 0.8 * 0 = 100

Reinforcement Learning [2]

➢ Q-Learning - Episode 2 Continue (Gamma = 0.8, and the current state is Room 1)





$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 100 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 80 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Action

$$R = \begin{array}{c|cccccc} \text{State} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & -1 & -1 & -1 & 0 & -1 & 100 \\ 2 & -1 & -1 & -1 & 0 & -1 & -1 \\ 3 & -1 & 0 & 0 & -1 & 0 & -1 \\ 4 & 0 & -1 & -1 & 0 & -1 & 100 \\ 5 & -1 & 0 & -1 & -1 & 0 & 100 \end{array}$$

$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 100 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 80 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]

Q(1, 5) = R(1, 5) + 0.8 * Max[Q(5, 1), Q(5, 4), Q(5, 5)] = 100 + 0.8 * 0 = 100

Reinforcement Learning [2]

- Q-Learning
  - If our agent learns more through further episodes, it will finally reach convergence values in matrix Q like:

$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 400 & 0 \\ 1 & 0 & 0 & 0 & 320 & 0 & 500 \\ 2 & 0 & 0 & 0 & 320 & 0 & 0 \\ 3 & 0 & 400 & 256 & 0 & 400 & 0 \\ 4 & 320 & 0 & 0 & 320 & 0 & 500 \\ 5 & 0 & 400 & 0 & 0 & 400 & 500 \end{array}$$

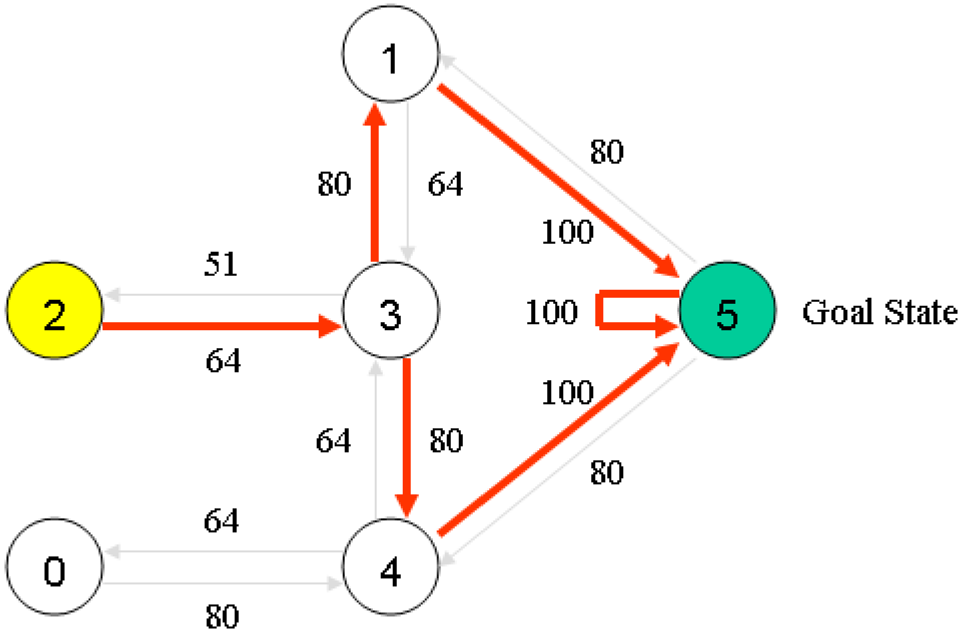
  - This matrix Q, can then be normalized (i.e.; converted to percentage) by dividing all non-zero entries by the highest number (500 in this case):

$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 80 & 0 \\ 1 & 0 & 0 & 0 & 64 & 0 & 100 \\ 2 & 0 & 0 & 0 & 64 & 0 & 0 \\ 3 & 0 & 80 & 51 & 0 & 80 & 0 \\ 4 & 64 & 0 & 0 & 64 & 0 & 100 \\ 5 & 0 & 80 & 0 & 0 & 80 & 100 \end{array}$$

Reinforcement Learning [2]

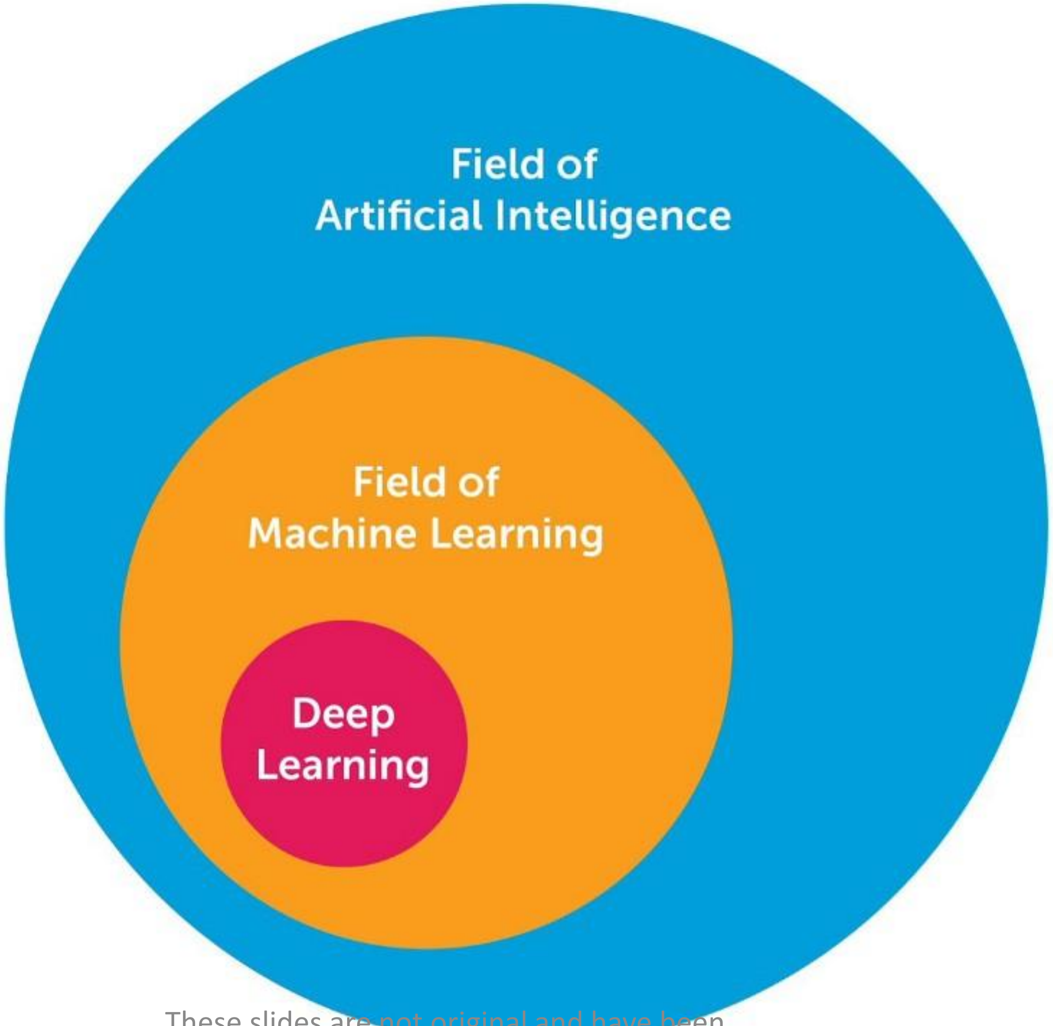
- Q-Learning

$$Q = \begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 0 & 0 & 0 & 80 & 0 \\ 1 & 0 & 0 & 0 & 64 & 0 & 100 \\ 2 & 0 & 0 & 0 & 64 & 0 & 0 \\ 3 & 0 & 80 & 51 & 0 & 80 & 0 \\ 4 & 64 & 0 & 0 & 64 & 0 & 100 \\ 5 & 0 & 80 & 0 & 0 & 80 & 100 \end{array}$$



Introduction

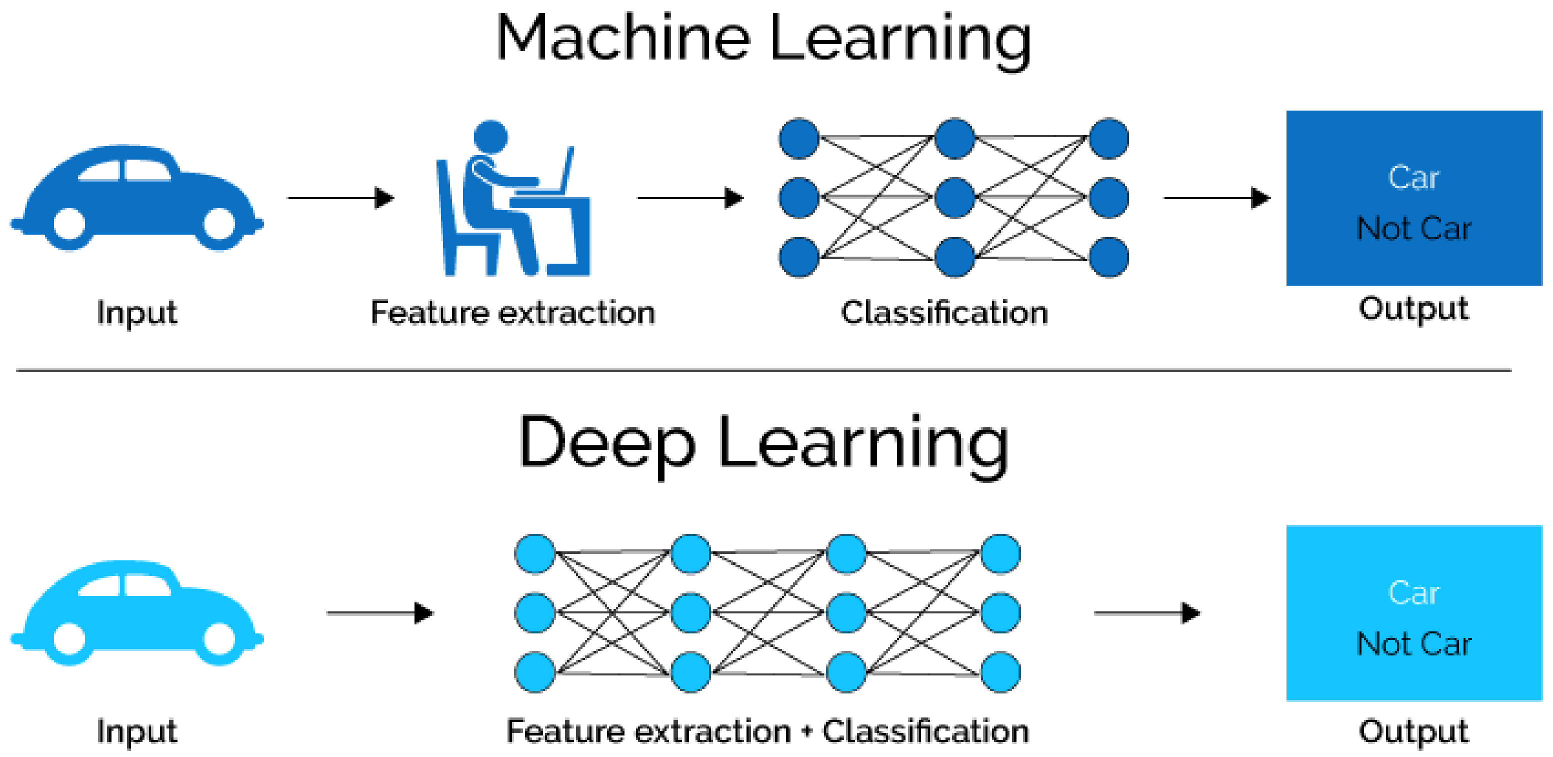
- AI, ML and DL



Source: [3]

Introduction

- Machine Learning vs Deep Learning



Source: [4]

Major Architectures of Deep Networks

- Four Major Architectures:
  - Unsupervised Pretrained Networks (UPNs)
  - Convolutional Neural Networks (CNNs)
  - Recurrent Neural Networks
  - Recursive Neural Networks

Major Architectures of Deep Networks

- Four Major Architectures:
  - Unsupervised Pretrained Networks (UPNs)
    - Autoencoders
    - Deep Belief Networks (DBNs)
    - Generative Adversarial Networks (GANs)

    - Use Cases:
      - Feature Extraction
      - Synthesizing

Source: [5]

Major Architectures of Deep Networks

- Four Major Architectures:
  - Convolutional Neural Networks (CNNs)
    - Lenet-5
    - AlexNet
    - VGGNet
    - GoogleNet (Inception)
    - ResNet
    - ResNext
    - DenseNet
    - RCNN (Region Based CNN)
    - YOLO (You Only Look Once)
    - SqueezeNet
    - SegNet

Major Architectures of Deep Networks

- Four Major Architectures:
  - Convolutional Neural Networks (CNNs)
  - Use Cases:
    - Computer Vision
    - Natural Language Processing

Major Architectures of Deep Networks

- Four Major Architectures:
  - Recurrent Neural Networks
    - Hopfield Network
    - Long Short-Term Memory (LSTM)
    - Gated Recurrent Unit (GRU)
  - Use Cases:
    - Sentiment Classification
    - Image Captioning
    - Language Translation
    - Video Captioning

Major Architectures of Deep Networks

- Four Major Architectures:
  - Recursive Neural Networks
    - Recursive Autoencoder
    - Recursive Neural Tensor Network
  - Use Cases:
    - Image scene decomposition
    - NLP
    - Audio-to-text transcription

Disclaimer

- These slides are not original and have been prepared from various sources for teaching purpose.