Abstract—In this paper, we review some pattern recognition learning methods and the models published in recent years. With the fast advancement of computer architecture, machine learning, and computer vision, computational complexity is possible to be dealt with and more and more new ways of thinking are brought into the research of pattern recognition. The Multi-Layer Perceptron algorithm for classification in pattern recognition is adopted to a particular situation. The objective of this paper is to provide better ability of learning and adoption. Machine learning and pattern recognition complement with each other, which means concepts of pattern recognition could be used for a design of proper learning algorithm, while learning algorithm could be used to enhance the result of pattern recognition.

I. INTRODUCTION

Pattern recognition is a process that takes in raw data and makes an action based on the category of the pattern. It optimally extracts patterns based on certain conditions and separates one class from another. Pattern recognition was often achieved using linear and quadratic discriminants, the k-nearest neighbour classifier or the Parzen density estimator, template matching and Neural Networks. These methods are basically statistic. Neural Network have gained prominence in the field of pattern classification. The overview of this paper is as follows. We first introduce some learning methods of:
1. Pattern recognition[1] and basic models in section
2. MLP in section
3. Implementation example
4. Conclusions

II. LEARNING METHODS & MODELS

Algorithms for pattern recognition based on statistical modelling of data. With statistical model in hand, one applies probability theory and decision theory to get an algorithm. This is opposed to using heuristics/”common sense” to design an algorithm.

2.1 Supervised learning
Supervised learning is the machine learning task of inferring a function from labelled training data. The training data consists of a set of training examples. It incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. It assumes that a set of training set has been provided, consisting of a set of instances that have been properly labelled by hand with the correct output. Usually, supervised learning is performed off-line. We say that a neural network learns off-line if the learning phase and the operation phase are distinct. Paradigms of supervised learning include error-correction learning, rein for cement learning and stochastic learning.

2.2 Unsupervised learning
The learner is unlabelled and is based upon only local information. It self-organizes data presented to the network and detects their emergent collective properties and controls degree of similarity between members. It virtualises training data that has not been hand-labelled, and attempts to find pattern recognition tasks in the data that can then be used to determine the correct output value for new data instances. A neural network learns on-line if it learns and operates at the same time. Unsupervised learning is performed on-line. Paradigms of unsupervised learning [3] are Hebbian learning and competitive learning.

2.3 Generative Model
Generative models are typically more flexible than discriminative models in expressing dependencies in complex learning tasks. Examples of generative models include:
- Gaussian mixture model and other types of mixture model
- Hidden Markov model
- Naive Bayes
- AODE
- Latent Dirichlet allocation
- Restricted Boltzmann Machine

2.4 Discriminative Model
These class of models are used in machine learning for modelling the dependence of an unobserved variable y on an observed variable x. For classification discriminative models generally yield superior performance. These models belong to supervised learning. Examples of discriminative models used in machine learning include:
- Logistic regression
- Linear discriminant analysis
- Support vector machines

III. MULTI LAYER PERCEPTRON

A multi-layer perceptron (MLP) is a feed-forward artificial neural network. A more sophisticated neuron (figure 1) is the McCulloch and Pitts model (MCP). They are the “threshold logic units”. The
The difference from the single layer model is that the inputs are 'weighted'; the effect that each input has at decision making is dependent on the weight of the particular input. The weight of an input is a number which when multiplied with the input gives the weighted input. These weighted inputs are then added together and if they exceed a pre-set threshold value, the neuron fires. In any other case the neuron does not fire.

In mathematical terms, the neuron fires if and only if:  
\[ X_1w_1 + X_2w_2 + X_3w_3 + \ldots > T \]

The addition of input weights and of the threshold makes this neuron a very flexible and powerful one. The MCP neuron has the ability to adapt to a particular situation by changing its weights and/or threshold. Various algorithms exist that cause the neuron to 'adapt'; the most used ones are the Delta rule and the back error propagation. The former is used in feed-forward networks and the latter in feedback networks. MLP utilizes a supervised learning [3] technique called back propagation for training the network. MLP is a modification of the standard linear perceptron, which can distinguish data that is not linearly separable.

**IV. AN EXAMPLE**

The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not), for particular input patterns. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.

**4.1 Firing Rule**

The firing rule is an important concept in neural network and accounts for their high flexibility. A firing rule determines how one calculates whether a neuron should fire for any input pattern. It relates to all the input patterns, not only the ones on which the node was trained. A simple firing rule can be implemented by using Hamming distance technique. The rule states:

'Take a collection of training patterns for a node, some of which cause it to fire (the 1-taught set of patterns) and others which prevent it from doing so (the 0-taught set). Then the patterns not in the collection cause the node to fire if, on comparison, they have more input elements in common with the 'nearest' pattern in the 1-taught set than with the 'nearest' pattern in the 0-taught set. If there is a tie, then the pattern remains in the undefined state'.

**4.2 Implementation**

An important application of neural networks is pattern recognition. Pattern recognition can be implemented by using a feed-forward (figure 1) neural network that has been trained accordingly. During training, the network is trained to associate outputs with input patterns. When the network is used, it identifies the input pattern and tries to output the associated output pattern. The power of neural networks comes to life when a pattern that has no output associated with it, is given as an input. In this case, the network gives the output that corresponds to a taught input pattern that is least different from the given pattern. For Example: The network of figure 1 trained to recognize the patterns T and H. The associated patterns are all black and all white respectively as shown below using a feed-forward (figure 1) neural network that has been trained accordingly. During training, the network is trained to associate outputs within put patterns. When the network is used, it identifies the input pattern and tries to output the associated output pattern. The power of neural network comes to life when a pattern that has no output associated with it, is give nasanin put. In this case, the network gives the output that corresponds to a taught input pattern that is least different from the given pattern.
If we represent blacksquares with 0 and white squares with 1 then the truth tables for the 3 neurons after generalization are:

**Top neuron:**

<table>
<thead>
<tr>
<th>X11</th>
<th>X12</th>
<th>X13</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Middle neuron:**

<table>
<thead>
<tr>
<th>X21</th>
<th>X22</th>
<th>X23</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Bottom neuron:**

<table>
<thead>
<tr>
<th>X31</th>
<th>X32</th>
<th>X33</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**REFERENCES:**


**CONCLUSION:**

The computing world has gained a lot from neural networks. Their ability to learn by example makes them very flexible and powerful. Furthermore, there is no need to devise an algorithm in order to perform a specific task; i.e., there is no need to understand the internal mechanisms of that task. They are also very well suited for real-time systems because of their faster response and computational times which are due to their parallel architecture. They are regularly used to model part of a living organ is ms and to investigate the in ternal mechanisms of the brain. The basic idea we get is: the more relevant patterns at your process, the better feature subsets you obtain in, the more or less your classifier will be applied, finally the better your decisions will be. Finally, I would like to state that even though neural networks have huge potential we will only get the best of them when they are integrated with computing, AI, fuzzy logic and related subjects. In summary, to get a better way for our final goal we should attempt to design a hybrid system combining with multiple models.

In this case, it is obvious that the output should be all black since the input pattern is almost the same as the 'T' pattern. Here also, it is obvious that the output should be all whites since the input pattern is almost the same as the 'H' pattern.

Here, the top row is two errors away from the Tand 3 from an H. So the top output is put black. The idle row is one error away from both T and H so the output is random. The bottom row is one error away from Tand 2 away from H. Therefore, the output is black. The total output of the network is still in favour of the Tshape.

---

**Pattern Recognition In Neural Networks**

26