The Research Monograph Series in Computing, Electrical & Communication Networks
Vol. 1 (02), January 2024, pp. 17–21

MATHEMATICAL BACKGROUND FROM AI TO DL

This section discusses about the reviews of transformation from neural networks to DL from Russell et al. [74] and specifically to Goodfellow et al. [35] and Schmidhuber [75]. Figure 3.1 shows the group of AI approaches used for BMI [43].

3.1 PREDICTIVE ANALYTICS

BM predictive analysis took input features $x \in X \subseteq R^m$ to predict the variable y ∈ Y which is unknown. The input features are the recent sales numbers that can be used to forecast the future production prediction. Depends on its discrete or real value, this is referred as classification or regression task. The objective is the mapping function $f : x \mapsto y$. The choice of this model is given to predictive model *f(.,w)* with additional parameter w such that $y \approx f(x;w)$. The choice of best parameter value w is chosen with optimization problem. There are variety of models of f are used with business analytics such as linear models [76,77], support vector machine, decision tree, neural networks [78,79] or deep neural network. These methods are accompanied with optimization strategies [80]. This models can analyze the error between predicted value and true observation which is denoted as loss function $L : Y \times Y \rightarrow R$. predictive models will minimize the loss $L(\hat{y}, y) = L(f(x;w), y)$ for set of samples *i*. optimization problem is declared as in Eqn (1) where W is the weight space.

$$w^* = \arg\min_{w \in W} \sum_i L(\hat{y}_i, y_i) = \arg\min_{w \in W} \sum_i L(f(\hat{x}_i, w), y_i) \quad (1)$$

The predictive model flexibility is gained with the dimensionality increment of *W*. To avoid Overfitting, the choice of *w* and function *f* representation is discussed as follows:

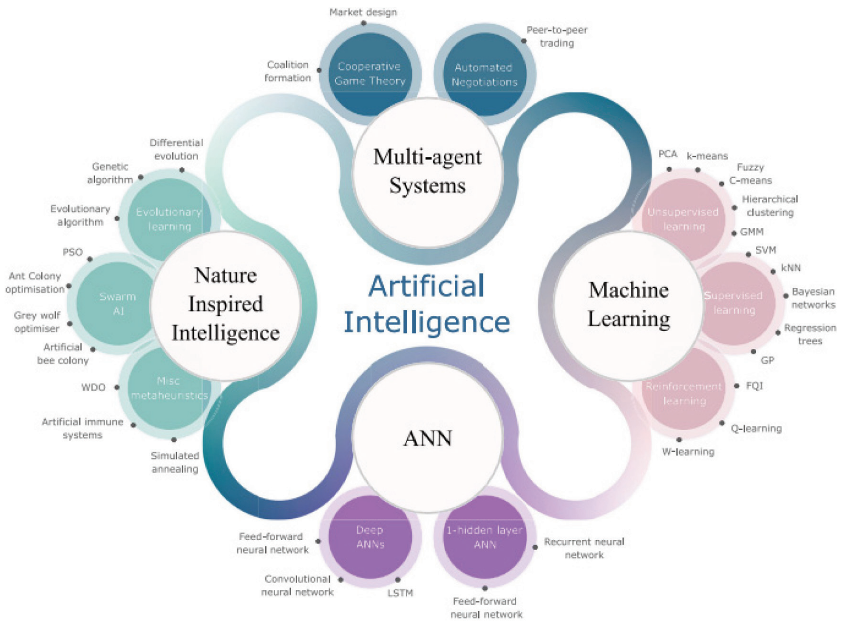

Figure 3.1. Group of AI approaches for BMI [43].

3.2. NEURAL NETWORKS

3.2.1. Single-layer neural networks

An ANN is defined as a collection of interconnected blocks called neurons. Artificial neuron can receive the inputs from other neurons, weighted sum of inputs are computed and map this sum through activation function to output. Figure 3.2 shows the single layer NN which takes input x and process it with weights and activation function. Finally it produces the output. If the connections are in forward direction then it is called as feed forward neural network. The network with no cycles and sends the feedback connection backward is called as recurrent neural network.

The simplest form of this network represented as single layer perceptron where the network fNN is computed with combination of activation function represented in Eqn (2) where W- weight matrix and b is bias.Common choices of the activation function is represented in Table 3.

$$fNN(x;W,b)=\sigma(Wx+b) \tag{2}$$

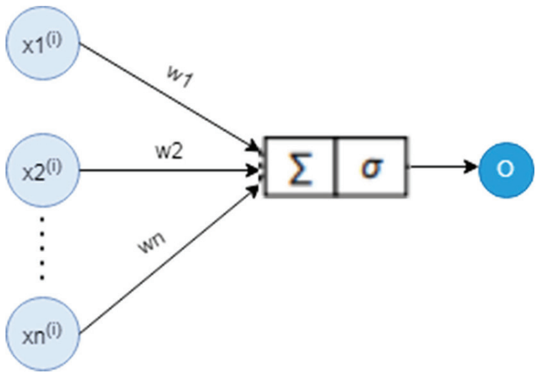

Figure 3.2. Architecture of single layer neural network.

Table 3.1. Activation functions.

Sigmoid function	$\sigma(x) = \frac{1}{(1+e^{(-x)})} \in (0,1)$
Hyperbolic tangent	$\sigma(x) = \frac{(e^2 x - 1)}{(e^2 x + 1)} \in [-1,1]$
Rectified linear unit (ReLU)	$\sigma(x) = \max(0,x) \in [0,\infty]$

Single layer neural network have the limitations such as monotonic activation function cannot learn the XOR function as $fNN([0,0],w) = 0$, $fNN([0,1],w) = 1$, $fNN([1,0],w) = 1$ *and* $fNN([1,1],w) = 0$. To overcome this, multi layer neural network with many layers that can transform their input into high dimensional representation into output.

3.2.2 Deep Neural Network (DNN)

The NN with arbitrary number of 5 layers are represented with the term "deep". The number of parameters of this NN are increased and used non-linear functions. Mathematical representation of this NN is the stacking of several single layer NN with k layers as in Eqn (3).

$$f_{DNN}(x) = fNN_1\left(fNN_2(\ldots fNN_k(x))\right) = fNN_1 \ldots fNN_k(x) \tag{3}$$

Input layer is the first layer and the last layer is the output layer. Layers between input and output layers are called as hidden layers from two to several hidden layers. Optimization of this DNN require two methods such as gradient based optimization and regularization method. The activation function of DNN is ReLU which will leads most of the hidden layers are not activated and make the output as zero. Recurrent neural network (RNN) are used sigmoid activation functions which makes the output between 0 and 1.

3.3. MODEL ESTIMATION

3.3.1. Weight optimization

In order to reduce the loss value of DL approaches, DL optimization is performed via Eqn (4)

$$w^* = \arg\min_{w \in W} L(f_{DNN}(x; w), y) \tag{4}$$

Single perceptron weight values are identified using convex optimization. DL optimization are complete due to the number of free parameters. NN optimization is NP hard problem that can produces the correct result for training samples. DL optimization is performed through gradient based numerical method [81]. Single layer NN optimized by computing the parameter partial derivatives in terms of L to decrease the loss value stated in Eqn (5) where η the gradient descent optimization learning rate is.

$$w \leftarrow w - \eta \sum_i \frac{\delta}{\delta w} L(f(x_i, w), y_i) \tag{5}$$

This can be done for all the layers from output to input and this approach is called as backpropagation. This can reuse the computation calculated from the previous layers. Since this is run though GPU with linear algebra, this is an effective method of optimization in DL.

3.1.2 Regularization

This regularization methods are used to avoid Overfitting of the network with weights regularization of the following forms:

(i) Weight decay: it is added as a regularization term with loss function. Set of all parameters are represented as W and the loss function is changed as in Eqn (6)

$$L_{wd} = \sum_i \frac{\delta}{\delta w} L\left(f(x_i, w), y_i) + \frac{\lambda}{2} \| W \|_2 \right) \tag{6}$$

The weight of gradient descent is updated as in Eqn (7)

$$w \leftarrow w - \eta \sum_i \frac{\delta}{\delta w} L(f(x_i, w), y_i) + \lambda \| W \|_2 \tag{7}$$

(ii) Dropout: during each iteration of training process, small portion of neurons are discarded by setting the corresponding row of *W* as zero.

(iii) Batch Normalization: each layer output is normalized before forwarding to the next layer [82] which shrink the value closer to zero with high learning rate.

3.4 ADVANCED ARCHITECTURES

apart from the architectures discussed, some more alternative methods are listed with specific data structures in Table 3.2.

Table 3.2. overview of common DL network architectures

Architecture	Data Structures	Examples	Parameters
Multi layer perceptron	Fixed length feature vectors	Traditional method simple replacement	Number of layers and neurons, activation function
Convolutional neural network	High dimensional data	Image and speech recognition	Number of convolution layers
Long short term memory	Sequential data	Time series forecasting, text classification	Number of layers and neurons
Gated recurrent unit	Sequential data	Text mining, time series forecasting	Number of layers and neurons