Neural Networks and SVMs: Applications to Modeling and Control

Dr. Alok Kanti Deb

Department of Electrical Engg, IIT Kharagpur
alokkanti@ee.iitkgp.ernet.in
Contents

• Neural Networks
• Support Vector Machines
• Adaptive Critic Designs for Control
• Modeling the Bias in Automobile modeling
Basic Learning Element: Human Brain

- Weight = 3lbs
- Volume = 90 cu. inches.
- Total no. of cells = 90 billions. (1 billion=10⁹)
- Neuron – a special cell that conducts electrical signals (10 billions)
- Size of a neuron = 1/100-th of a ‘dot’.
- Brain made up of vast network of neurons, that are coupled with receptors and effector.
Some Organizational & Computational Principles of the Brain

• Massive parallelism.
• High degree of connection complexity.
• Trainability.
• Binary states and continuous variables.
• Numerous types of neurons and signals.
• Intricate signal interaction – interaction of impulses received at a single neuron is highly nonlinear.
• Physical decomposition- brain consists of several subnetworks each consisting of several thousand densely connected neurons.
• Functional decomposition – different subnetworks are responsible for different functions.
- Neuron – a special cell that conducts electrical signals (10 billions)
- Size of a neuron = 1/100-th of a ‘dot’.
Brain made up of vast network of neurons, that are coupled with receptors and effector.
Background

- Recent neuro-biological evidence points towards the spiking nature of neural networks.

- Spikes are suitable for implementations, in digital and analog domain.

- Computational consideration have made spiking neural networks unpopular.

- Problems lie in training since the values are in the form of a train of spikes.

- Actual neuro-biological evidence increasingly point towards the mechanism of Spike Timing Dependent Plasticity (STDP) for modification of synaptic connections.
Synapses either facilitates or inhibits during regular firing – a phenomenon called **plasticity**.

- **Excitatory Synapse**— excitation of pre-synaptic neuron contributes to the depolarization of the post-synaptic neuron.

- **Inhibitory Synapse**— excitation of the pre-synaptic neuron moves the potential of the post synaptic neuron away from the threshold or makes it harder to be depolarized.
Spiking Neuron Models

Hodgkin-Huxley model (1952)

Integrate and Fire model
  - Leaky Integrate and Fire model (Stein, 1967)
  - Nonlinear Integrate and Fire model
    (Abbott and van Vreeswijk, 1993)

Spike Response model (Gerstner, 1995)

Multi-compartment Integrate and Fire model
  (Abbott et. al., 1991, Bressloff and Taylor, 1994)
Network Architecture

Synaptic connections between $i$-th and $j$-th neuron

\[ \begin{align*}
&d_{ij}^1 & &\varepsilon_{ij}^1 \\
&d_{ij}^2 & &\varepsilon_{ij}^2 \\
&d_{ij}^n & &\varepsilon_{ij}^n
\end{align*} \]
Any neuron $j$ is identified by a set of firing times, \[ \mathcal{I}_j = \{ t^f_j ; 1 \leq f \leq n \} \]

The synaptic potentials are modeled as $\alpha$-functions
\[ \varepsilon(t) = \frac{t}{\tau} e^{\frac{1-t}{\tau}} \]

Contribution of $k$-th branch of $i$-th synaptic terminal to the $j$-th state is given by
\[ y_{ij}^{k}(t) = \varepsilon_{ij}^{k}(t - t^f_i - d_{ij}^{k}) \]

The potential of neuron $j$,
\[ a_j(t) = \sum_{i \in \mathcal{I}_j} \sum_{t_i' \in \mathcal{F}_j} \sum_k w_{ij}^{k} \varepsilon_{ij}^{k}(t - t^f_i - d_{ij}^{k}) \]
Spikes are generated as the neuron membrane potential $a_j$ crosses a given threshold $\theta$ from below,

$$a_j(t^f) = \theta \wedge \left. \frac{da_j(t)}{dt} \right|_{t=t^f} > 0$$

$\theta = 1$
Spike Timing Dependent Plasticity (STDP) model
(van Rossum et. al., 2000)

Long Term Potentiation (LTP)
If a synaptic event precedes a postsynaptic spike, the synapse is potentiated.

$$w = w + w_p; w_p = c_p e^{\tau_{STDP} \delta t}$$

Long Term Depression (LTD)
If a synaptic event follows a postsynaptic spike, the synapse is depressed.

$$w = w + w_d; w_d = -c_d w e^{\tau_{STDP} \delta t}$$

$$\tau_{STDP} = 20 \text{ msecs}; c_d = 0.003; c_p = 7 \times 10^{-12}$$
Rate as a Spike Count (Average over Time)

Temporal average of spikes describes the activity of a neuron.

\[ \gamma = \frac{n_{sp}}{T} \]

Temporal averaging works well when the stimulus is constant or slowly varying.

Rate as Spike Density (Average over several runs)

Spike density in Peri-Stimulus Time Histogram (PSTH)

\[ \rho = \frac{1}{\Delta T} \cdot \frac{1}{K} n(t, t + \Delta T) \]
• Information is not encoded only by the excitation and resting states of the neurons; the rate at which a neuron is excited also carries information.

• Hence a neuron is identified by the rate at which it generates the spikes.

• This is used for computational purposes in ANN.
Simple Neuron Model

\[
\text{net} = \sum_{i=1}^{n} w_i x_i - \theta = \sum_{i=0}^{n} w_i x_i; \ w_0 = -\theta; \ x_0 = 1
\]
Different Activation Functions

- **Threshold Logic Unit (TLU)**

  \[ y = \begin{cases} 
  1 & \text{if } \sum_{i=1}^{n} w_i x_i \geq \theta \\
  -1, & \text{otherwise} 
\end{cases} \]

- **Logsigmoid**

  \[ y = \frac{1}{1 + e^{-\lambda \left( \sum_{i=1}^{n} w_i x_i - \theta \right)}} \]
• **Tansigmoid**

\[ y = \frac{2}{1 + e^{-\lambda \left( \sum_{i=1}^{n} w_i x_i - \theta \right)}} - 1 \]

• **Saturated Linear**

\[ y = \begin{cases} 
1, & \text{if net} \geq 1 \\
-1, & \text{if net} \leq -1 \\
net, & \text{otherwise}
\end{cases} \]
### AND Problem

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### OR Problem

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
XOR Problem

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$A(0,0)$ $D(1,1)$

$B(0,1)$ $C(1,0)$

$z=0$ $z=1$

$\begin{align*}
-x_1 + x_2 - 1 &= 0 \\
-x_1 + x_2 - 1 &= 0 \\
-x_1 + x_2 + 1 &= 0
\end{align*}$

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$O_1$</th>
<th>$O_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

$A(0,0)$ $B(0,1)$ $D(1,1)$

$C(1,0)$

$z=0$ $z=1$
Kolmogorov’s Theorem

Any continuous function $f(x_1, x_2, \ldots, x_n)$ of $n$ variables $x_1, x_2, \ldots, x_n$ can be represented in the form,

$$f(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{2n+1} h_j \left( \sum_{i=1}^{n} g_{ij}(x_i) \right)$$

where, $h_j$ and the $g_{ij}$ are continuous functions of one variable, $g_{ij}$’s are fixed monotone increasing functions

⇒ Any function approximation problem can be learnt by a multilayer network.
• Given a complex classification problem, single layer may not be sufficient.

• Commonly neural networks are adjusted, or trained, so that a particular input leads to a specific target output.

• There, the network is adjusted, based on a comparison of the output and the target, until the network output matches the target.

• Typically many such input / target pairs are used, to train a network.

• No satisfactory way to choose the neural network architecture and number of neurons.

• Smaller networks may be unable to learn a problem.

• Larger networks tend to overfit the data
Techniques Available

◆ Pruning techniques
  
  Optimal Brain Damage
  Optimal Brain Surgeon.

◆ Constructive techniques
  
  Pyramidal Delayed Perceptron
  Oil Spot Algorithm, CARVE
  Mpyramidal and MTiling algorithms
  Neural network construction with cross-validation samples (N2C2S)

◆ Regularization
  
  N2P2F Algorithm
Pattern Recognition

Measurement Space
→ Feature Space
→ Decision Space

Main Tasks:
Feature Selection and
Supervised / Unsupervised / Reinforcement Learning
Learning from Data

• Modeling systems from first principles may be impeded by lack of enough knowledge about the system.

• Handwriting of different people inject variable amount of noise in an optical character recognition system.

• Empirical models that learn from data is attractive.
Machine Learning Problems

- **Classification Problem**

  Pattern Set

  \[ S = \{(X_i, y_i); X_i \in \mathbb{R}^n, y_i \in \{1,2,\cdots, M\}, i = 1,2,\cdots, N\} \]

  \[ y_i \in \{1,-1\} \quad \text{(binary classification problem)} \]

- **Regression Problem**

  Pattern Set

  \[ S = \{(X_i, y_i); X_i \in \mathbb{R}^n, y_i \in \mathbb{R}, i = 1,2,\cdots, N\} \]
• **Supervised Learning**: A class label is available.

---

• **Unsupervised Learning**: A class label is not available.
• **Reinforcement Learning**: System information is available as an evaluation signal from a ‘teacher’.
Feature Selection

• Choosing the features (components of $X$) is a very important step in the learning process.

• Domain knowledge plays an important role.
  - In learning to drive a car, whether the driver is blonde or can play musical instruments is of little relevance.

• Retaining as many features may be advantageous; leaving out an important feature can be disastrous.

Features $x_1, x_2, \ldots, x_n$

$b$ no. of features to be selected. $b < n$
Uses of Feature Selection

• Reduction in computational complexity.
• Redundant features can be removed.
• Better insight into the classification problem.

Steps of Feature Selection

• Objective function that attaches a value to every subset of the features is to be devised.
• Formulation of feature selection algorithms.
Decision Tree learning

If $(Outlook = Sunny) \& (Humidity = High)$ then $Play\ Tennis = NO$

If $(Outlook = Sunny) \& (Humidity = Normal)$ then $Play\ Tennis = YES$
Tennis Example

Temperature

Humidity

- green circle = play tennis
- red circle = do not play tennis
Perceptron Learning Rule

\((X_i, y_i) \ \forall i = 1,2,\ldots, N\)

\(X_i \in \mathbb{R}^n \ \forall i\)

\(y_i \in \{1,2,\ldots, M\}\)

\(y_i\) denotes the class of \(X_i\)

Assumptions:

Let \(M = 2\)

Let there exist a hyperplane which classifies all points correctly.
Remarks

• If the original patterns are linearly separable, a perceptron will be able to classify the patterns.

• If the patterns are linearly inseparable, the patterns have to be mapped to an image space where they become linearly separable.

• If $0 < N < 2(n+1)$ a perceptron will correctly classify the $N$ patterns with asymptotic probability 1, as $n \to \infty$.

Linearly inseparable patterns can be made linearly separable by

• increasing $n$
• decreasing $N$, i.e. mapping multiple patterns to the same class.
Increasing $n$: XOR Problem

$x_1 \ x_2 \ z$

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 1 & 1 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
\end{array}
\]

$[x_1 \ x_2] \Rightarrow [x_1 \ x_2 \ x_3]$

$x_3 = T(x_1 x_2 - 0.5)$ where $T(x) = \begin{cases} 
1, & x > 0 \\
0, & x \leq 0 
\end{cases}$

$x_1 \ x_2 \ x_3 \ z$

\[
\begin{array}{ccc}
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 \\
\end{array}
\]
Decreasing $N$: XOR Problem

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

$\phi_1(x_1, x_2) = T(x_2 - x_1x_2 - 0.5)$
$\phi_2(x_1, x_2) = T(x_1 - x_1x_2 - 0.5)$

$T(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases}$
Backpropagation Training

Input Layer

Hidden Layer

Output Layer
Given the training set, \( \{X(k), y(k)\}_{k=1}^{N} \)

- \( X(k) \) - Input pattern vector
- \( y(k) \) - Output pattern vector
- \( o(k) \) - Actual output due to the \( k \)-th pattern

Sum squared error over all output units for the \( k \)-th pattern

\[
E(k) = \frac{1}{2} \sum_{j=1}^{m} (o_j(k) - y_j(k))
\]

Total classification error over the \( N \) patterns

\[
E_T = \frac{1}{2} \sum_{k=1}^{N} E(k)
\]
Weight update rule,

\[ \mathbf{w}(i + 1) = \mathbf{w}(i) - \eta \frac{\partial E_T}{\partial \mathbf{w}(i)} \]

\( \eta \) - Learning rate.

Weight update with momentum,

\[ \mathbf{w}(i + 1) = \mathbf{w}(i) - \left[ \eta \frac{\partial E_T}{\partial \mathbf{w}(i)} + \beta \mathbf{w}(i - 1) \right] \]

\( \beta \) - Momentum parameter
Statistical Learning Theory

\[(X^i, y^i); \quad \forall i = 1, 2, \ldots, N\]

\[X^i \in \mathbb{R}^n \quad \forall i\]

\[y^i \in \{+1, -1\}\]

**Goal:** To estimate a function, \(f : \mathbb{R}^n \rightarrow \{+1, -1\}\)

such that \(f\) can correctly classify unseen examples \((X, y)\) that were generated from the same probability distribution, \(P(X, y)\).

Empirical Risk,

\[R_{emp}[f] = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \left| f(X^i) - y^i \right|\]
Actual Risk

\[ R[f] = \int \frac{1}{2} |f(X) - y|dP(X, y) \]

Structural Risk Minimization (SRM) (Vapnik, 1979)

\[ R[f] \leq R_{emp}[f] + \sqrt{\frac{h \left( \log \frac{2N}{h} + 1 \right) - \log \left( \frac{\eta}{4} \right)}{N}}; \quad 0 < \eta < 1 \]
For better performance, the function space must be restricted such that $h$ is small enough (in relation to the available amount of data).

Problem is to find functions having optimum $h$. 
A set of functions that consists of nested subsets $\mathcal{I}_k$ are constructed such that,

$$\mathcal{I}_1 \subseteq \mathcal{I}_2 \subseteq \cdots \subseteq \mathcal{I}_k \subseteq \cdots$$

Each element of $\mathcal{I}_k$ has finite VC-dimension $h_k$.

The structure provides ordering of the elements according to the complexity

$$h_1 \leq h_2 \leq \cdots \leq h_k \leq \cdots$$
Steps to Learning using SRM

• Selecting an element of the structure (having optimum complexity) ⇒ Model Order Selection.

• Estimating the model from this element ⇒ Parameter Estimation.
Linear Support Vector Machine

Data: \[ \left\{ (X^i, y^i); X^i \in \mathbb{R}^n; \right. \]
\[ \left. y^i \in \{-1, +1\}; i = 1, 2, \ldots, N \right\} \]
All hyperplanes in $\mathbb{R}^n$ are parameterized by $w$ and a constant $b$

Objective: To find a hyperplane $f(X) = \text{sign}(\langle w, X \rangle + b)$ that correctly classify the data
The optimal hyperplane $H$ is such that

\[
\langle w.X^i \rangle + b \geq +1 \quad \text{when } y^i = +1
\]

\[
\langle w.X^i \rangle + b \leq -1 \quad \text{when } y^i = -1
\]

Define

\[
H : \langle w.X \rangle + b = 0
\]

\[
H_1 : \langle w.X \rangle + b = +1
\]

\[
H_2 : \langle w.X \rangle + b = -1
\]
$H : \langle w.X \rangle + b = 0$

$H_1 : \langle w.X \rangle + b = +1$

$H_2 : \langle w.X \rangle + b = -1$

$d^+ / d^- = \text{shortest distance to the closest positive / negative point}$

Margin of separation = $d^+ + d^-$
Non Linear SVM: The kernel trick

The mapping function $\phi$ maps the data to a higher dimensional linearly separable space.
Least Squares Support Vector Machine (LSSVM)


Given a training set, \( S = \{(x^i, y^i); x^i \in \mathbb{R}^n, y^i \in \mathbb{R}; i = 1, 2, ..., N\} \)

\[ \phi: \mathbb{R}^n \rightarrow \mathbb{R}^{n_h} \]

To find, \( w \in \mathbb{R}^{n_h}; w_0 \in \mathbb{R} \) from the following optimization problem.

Minimize \( \frac{1}{2} \langle w, w \rangle + \gamma \frac{1}{2} \sum_{i=1}^{N} \xi_i^2 \)

subject to, \( y^i = \langle w, \phi(x^i) \rangle + w_0 + \xi_i, \ i = 1, 2, ..., N \)

The Lagrangian

\[ L_{LS}(w, w_0, \xi, \alpha) = \frac{1}{2} \langle w, w \rangle + \gamma \frac{1}{2} \sum_{i=1}^{N} \xi_i^2 - \sum_{i=1}^{N} \alpha_i \left( \langle w, \phi(x^i) \rangle + w_0 + \xi_i - y^i \right) \]
\[ \nabla_w L_{LS} = 0 \Rightarrow w = \sum_{i=1}^{N} \alpha_i \phi(x^i) \dots (i) \]

\[ \frac{\partial L_{LS}}{\partial w_0} = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i = 0 \dots (ii) \]

\[ \frac{\partial L_{LS}}{\partial \xi_i} = 0 \Rightarrow \alpha_i = \gamma \xi_i, \ i = 1, 2, \ldots, N; \dots (iii) \]

\[ \frac{\partial L_{LS}}{\partial \alpha_i} = 0 \Rightarrow \left\langle w.\phi(x^i) \right\rangle + w_0 + \xi_i - y^i = 0, \ i = 1, 2, \ldots, N; \dots (iv) \]

Substituting \( w \) and \( \xi_i \) from (i) and (iii) in (iv), (ii) and (iv)

can be compactly written as,

\[
\begin{bmatrix}
0 & \tilde{1}^T \\
\tilde{1}^T & K + \gamma^{-1}I
\end{bmatrix}
\begin{bmatrix}
w_0 \\
\alpha
\end{bmatrix}
= \begin{bmatrix}
0 \\
y
\end{bmatrix}
\dots (v)
\]

where, \( y = [y_1 \ y_2 \ \ldots \ y_N]^T \), \( \tilde{1} = [1 \ 1 \ \ldots \ 1]_{1 \times N}^T \), \( \alpha = [\alpha_1 \ \alpha_2 \ \ldots \ \alpha_N]^T \), and

\[ K_{ij} = K(x^i, x^j) = \left\langle \phi(x^i), \phi(x^j) \right\rangle, \ i, j = 1, 2, \ldots, N. \]
The Lagrangian variables $\alpha_i$ are proportional to the errors at the data points.

With some choice of the kernel matrix $K$, ($\nu$) can be solved for $w_0$ and $\alpha$.

**Advantage:** LS-SVM involves solving a set of linear equations.

The LS-SVM model of function approximation is given by,

$$y(x) = \sum_{i=1}^{N} \alpha_i K(x^i, x) + w_0$$
Example: To learn the function \( y = 1 + x^2 \sin(x) \)

\[
\phi(x) = \begin{bmatrix} x & x^2 & x^3 & x^4 & x^5 \end{bmatrix}^T
\]

\( \gamma = 1071; c = 0.01 \)
Example

\[ y = x_1 \sin(x_2) \]

\[ \phi(x) = \phi(x_1, x_2) = [x_1 \ x_2 \ x_1^2 \ \cdots \ x_2^2 \ x_1^3 \ \cdots \ x_2^3 \ x_1^4 \ \cdots \ x_2^4]^T \]

\[ \gamma = 10000; \ c = 0.01 \]
Example. An Artificial Classification Problem

Points in Class 1 : (1,8), (4,5), (4,4), (1,1), (6,5), (6,4), (10,8), (10,1)

Points in Class 0 : (2,6), (2,3), (8,6), (8,3)

SVM based Tree type neural network
Neuron 1: $x_1 - 3 = 0$

Neuron 2: $3 - 2x_1 = 0$

Neuron 3: $x_1 - 7 = 0$
Neuron 3: $x_1 - 7 = 0$

Neuron 4: $x_1 - 9 = 0$
Classification contours generated by the entire network.
Example: Results of 10-fold cross-validation on Real data sets

<table>
<thead>
<tr>
<th>Name of data set</th>
<th>SVM based Tree Network</th>
<th>Training accuracy (%)</th>
<th>Testing accuracy (%)</th>
<th>N2C2S [Setiono, 2001]</th>
<th>Training accuracy (%)</th>
<th>Testing accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer-W</td>
<td>99.59±0.27</td>
<td>95.32±0.52</td>
<td>97.47±0.06</td>
<td>96.58±0.24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heart-statlog</td>
<td>99.074±0.64</td>
<td>77.67±1.27</td>
<td>94.08±1.35</td>
<td>77.56±1.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Jayadeva, A. K Deb and Suresh Chandra, “Binary Classification by SVM based Tree Type Neural Networks”, *Procs. of Int. J. Conf. on Neural Networks (IJCNN)*, 2002.
Introduction to Control

**Open Loop Control**

- Command input
- Manipulated variable
- Disturbance input
- Process
- Controlled output

**Closed Loop Control**

- Command input
- Manipulated variable
- Disturbance input
- Process
- Controlled output
- Feedback signal
Control Example: Car Driving

- Direction of Highway
- Speed limits
- (Brain)
- Steering wheel position
- Wind + traffic Disturbance
- Actuator (Hands)
- (Foot) Acceleration/Brake position
- Actuator
- Speed
- Heading
**Control System: Block Diagram**

**Elements:**
A : Reference input elements  
G_C: Control logic elements  
G_A: Actuator elements  
  (Final control element)  
G_P: Controlled system elements  
H : Feedback elements

**Signals:**
b: Feedback signal  
y: Controlled variable  
w: Disturbance input  
e: Actuator error signal  
u: Control signal  
m: manipulated variable  
r: Reference input  
y_r: Command input
Adaptive Critic Designs for Control

Plant

Critic

Evaluation signal

Action Element
Various Adaptive Critic Design (ACD) Models

- $x(t) \in \mathbb{R}^n$: Denotes the system states
- $u(t) \in \mathbb{R}^m$: Control actions
- $J(t)$: Performance Index

$$R(t) = x(t);$$
$$R(t) = \begin{bmatrix} x(t) \\ u(t) \end{bmatrix};$$

- HDP: Heuristic Dynamic Programming
- DHP: Dual Heuristic Programming
- GDHP: Globalized Dual Heuristic Programming

**HDP-style CRITIC**

**DHP-style CRITIC**

**GDHP-style CRITIC**
Control by Action Dependant Heuristic Dynamic Programming

Critic Element: SVM based Tree type NN (Jayadeva, A. K. Deb, S. Chandra, *IJCNN* 2002)

Desired Action element Mapping  \[ A: \{ x(t) \} \rightarrow \{ 0(t) \} \]
Given a non-linear dynamical system,

\[ x(t + 1) = P[x(t), u(t), t] \]

\( x \in \mathbb{R}^n \) Denotes the system states

\( u \in \mathbb{R}^m \) Control actions

The performance index at any instant is given by,

\[ J[x(t), t] = \sum_{k=t}^{\infty} \gamma^{k-t} U[x(k), u(k), k] \]

\( 0 < \gamma \leq 1 \) Discount factor
Objective:

To have a set of controls, \( u(k), k = t, t + 1, \ldots \) so that \( J \) is minimized.

In ADHDP, the task of the Critic network is to learn the function \( J(t) \).
• Some Learning Element may be trained to act as the Critic so that it can output cost function for the immediate future.

• The input-output relationship of the Critic Element may be given as,

\[ J(t) = J(x(t), u(t), t, W_c) \]
Training Scheme of ADHDP

- Critic and Action elements are trained in tandem.
- First with fault data the Critic element is trained.
- With the trained critic, Action training starts so as to minimize the output of the critic at fault condition.

Desired Mapping for the Action element

\[ A : \{x(t)\} \rightarrow \{0(t)\} \]
Experimental Results

**Plant:** a SIMO system, the model of a single link Inverted Pendulum

**Utility Function**

\[ U(t) = \begin{cases} 
0, & |x_2| \leq 12^\circ \text{ and } |x_1| \leq 0.5 \text{m} \\
1, & \text{otherwise} 
\end{cases} \]
Dynamical Equations of the Cart-Pendulum system

\[ \dot{x}_1 = x_3 \]
\[ \dot{x}_3 = \frac{a(u - T_c - \mu x_2^2 \sin x_2) + l \cos x_2 \left( \mu g \sin x_2 - f_p x_4 \right)}{J + \mu l \sin^2 x_2} \]
\[ \dot{x}_2 = x_4 \]
\[ \dot{x}_4 = \frac{l \cos x_2 \left( u - T_c - \mu x_2^2 \sin x_2 \right) + \mu g \sin x_2 - f_p x_4}{J + \mu l \sin^2 x_2} \]

where, \( a = l^2 + \frac{J}{m_c + m_p} \); \( \mu = (m_c + m_p)l \)

g = acceleration due to gravity
\( m_c = \) mass of the cart
\( m_p = \) mass of the pendulum
\( J = \) moment of inertia of the pendulum-cart system
\( T_c = f_c x_3 = \) friction motion of the cart
\( D_p = f_p x_4 = \) friction in the angular motion of the pendulum

Parameters

\[ Track \ length = \pm 0.5 m \]
\[ g = 9.81 \ m/ s^2 \]
\[ l = 0.017 \ m \]
\[ m_c = 1.12 \ kg \]
\[ m_p = 0.11 \ kg \]
\[ J = 0.0136 \ kg \cdot m^2 \]
\[ f_p = \text{negligible} \]
\[ f_c = 0.05 \ Ns/ m \]
Simulation results

Failure condition when the pendulum falls past the positive limit
Failure condition when the pendulum falls past the negative limit
Balancing the pendulum by the trained ADHDP based controller. Initial angle of the pendulum at 10.8° from the vertical.
Phase plane plot showing Pendulum Angular Velocity vs Pendulum Angle
Phase plane plot showing Cart Velocity vs Pendulum Angle
Angular deviation of the Pendulum starting from initial pendulum angles on two sides of the vertical at $\pm 10.8^\circ$. 
Controller Robustness

Effect of disturbance due to an impulse input at the 50th second.
Effect due to change in Plant parameters when the Pole length is reduced to half
Effect due to change in plant parameters when the mass of the Cart is doubled.
Other Applications to Control

- Power System Application
- Ship Tracking
- Aircraft Landing
Modeling the Bias in Automobile Engine Modeling
Overall Scope/Aim

**Aim: Develop methodology for on-board diagnosis based on MVEMs**

**AMESim**: Advanced Modelling Environment for Simulation of engineering systems by LMS Imagine Labs, Belgium

**MVEM**: Mean Value Engine Models

**Estimator**: EKF/UKF Estimators
AMESim's WCCM and MVEM – model structures

- 4 cylinder
- Fuel and EGR in closed loop control (PI)
- Simple modeling of driveline and load torque

WCCM: Within Cycle Crank Angle Based Models.

MVEM: Mean Value Engine Models.
General MVEM equations

Pressure Evolution
\[ \dot{P} = \frac{\sum \dot{m}_i RT + mRT + mRT}{\nu} \]

Temperature Evolution
\[ \dot{T} = \frac{\sum \dot{m}_i C_p T_i + \dot{Q}_i - (\dot{m}_C T + \dot{m}_C T)}{mC_v} \]

Mass flow across an orifice

Sub-sonic Flow
\[ \frac{P_{\text{down}}}{P_{\text{up}}} \geq \left( \frac{2}{\gamma + 1} \right)^{\frac{\gamma}{\gamma - 1}} \rightarrow \dot{m}_{\text{flow}} = C_d A_{\text{up}} \sqrt{\frac{\gamma'}{RT_{\text{up}}}} \left( \frac{P_{\text{down}}}{P_{\text{up}}} \right)^{\frac{2}{\gamma}} \left( \frac{P_{\text{down}}}{P_{\text{up}}} \right)^{\frac{(\gamma + 1)}{\gamma}} \]

Sonic Flow
\[ \frac{P_{\text{down}}}{P_{\text{up}}} \leq \left( \frac{2}{\gamma + 1} \right)^{\frac{\gamma}{\gamma - 1}} \rightarrow \dot{m}_{\text{flow}} = C_d A_{\text{up}} \sqrt{\frac{\gamma'}{RT_{\text{up}}}} \left( \frac{\gamma}{\gamma + 1} \right)^{\frac{(\gamma + 1)}{\gamma}} \]

Speed density Formula
\[ \dot{m}_{\text{in}} = \frac{\eta_{\text{vol}} * P_{\text{man}} * N * V}{R_{\text{man}} * T_{\text{man}} * \dot{m}_{\text{ap}} * 120} \]

Energy Flow Rates
\[ \dot{E}_{\text{ind}} = \dot{E}_{\text{ind}} \eta_{\text{ind}} \]
\[ \dot{E}_{\text{exh}} = \dot{E}_{\text{exh}} \eta_{\text{exh}} \]
\[ \dot{E}_{\text{them}} = \dot{E}_{\text{them}} \eta_{\text{them}} \]

MVEM states, inputs, outputs

### States

<table>
<thead>
<tr>
<th>Symbol</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{man}$</td>
<td>Intake Manifold Pressure</td>
</tr>
<tr>
<td>$T_{man}$</td>
<td>Intake Manifold Temperature</td>
</tr>
<tr>
<td>$P_{exh}$</td>
<td>Exhaust Manifold Pressure</td>
</tr>
<tr>
<td>$T_{exh}$</td>
<td>Exhaust Manifold Temperature</td>
</tr>
<tr>
<td>$m_{man}$</td>
<td>Total Mass of gas in intake manifold</td>
</tr>
<tr>
<td>$m_{man,a}$</td>
<td>Mass of air in intake manifold</td>
</tr>
<tr>
<td>$m_{man,bg}$</td>
<td>Mass of burnt gas in intake manifold</td>
</tr>
<tr>
<td>$m_{man,fub}$</td>
<td>Mass of unburnt fuel in intake manifold</td>
</tr>
<tr>
<td>$m_{exh}$</td>
<td>Total Mass of gas in exhaust manifold</td>
</tr>
<tr>
<td>$m_{exh,a}$</td>
<td>Mass of air in exhaust manifold</td>
</tr>
<tr>
<td>$m_{exh,bg}$</td>
<td>Mass of burnt gas in exhaust manifold</td>
</tr>
<tr>
<td>$m_{exh,fub}$</td>
<td>Mass of unburnt fuel in exhaust manifold</td>
</tr>
</tbody>
</table>

### Inputs

$$\mathbf{u} = \begin{bmatrix} \theta \\ N \\ \dot{m}_{f_{-av}} \\ EGR_{ctrl} \\ EGR_{ref} \end{bmatrix}$$

- Throttle angle
- Average Speed
- Average fuel flow rate
- EGR control signal
- EGR reference signal

### Outputs

$$\mathbf{y} = \begin{bmatrix} P_{man} \\ T_{man} \\ P_{exh} - P_{man} \\ T_{exh} \\ \dot{m}_{th} \end{bmatrix}$$

- MAP: Manifold Air Pressure
- MAT: Manifold Air Temperature
- DiffEGR: Differential EGR Pressure
- MATEx: Exhaust Manifold Temperature
- MAF: Mass Air Flow

Fuel and EGR Control is achieved using PI controllers
Efficiency factor Modelling

- \( \eta_{\text{vol}} \) vs \( \alpha \) and Speed
- \( \eta_{\text{therm}} \) vs \( \alpha \) and Speed
- \( \eta_{\text{ind}} \) vs \( \alpha \) and Speed
- \( \eta_{\text{exh}} \) vs \( \alpha \) and Speed

- EGR ratio = 0
- EGR ratio = 0.1
- EGR ratio = 0.2

- No. of operating points
  \( = \text{No. of throttle values} \times \text{No of speed values} \times \text{No. of EGR settings} \)
  \( = 8 \times 8 \times 3 \)
  \( = 192 \)

- DOE
Performance of MVEM vs Averaged WCCM

<table>
<thead>
<tr>
<th>State</th>
<th>Swing (WCCM)</th>
<th>Mean (WCCM)</th>
<th>% Peak error</th>
<th>% Average error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intake Manifold Pressure $P_{\text{man}}$</td>
<td>$0.81576 \times 10^5$</td>
<td>$1.03696 \times 10^5$</td>
<td>3.62%</td>
<td>0.01%</td>
</tr>
<tr>
<td>Intake Manifold Temperature $T_{\text{man}}$</td>
<td>76.7 K</td>
<td>329.7 K</td>
<td>7.62%</td>
<td>1.84%</td>
</tr>
<tr>
<td>Exhaust Manifold Pressure $P_{\text{exh}}$</td>
<td>$0.03611 \times 10^5$</td>
<td>$1.16879 \times 10^5$</td>
<td>1.12%</td>
<td>0.34%</td>
</tr>
<tr>
<td>Exhaust Manifold Temperature $T_{\text{exh}}$</td>
<td>357.4 K</td>
<td>799.3 K</td>
<td>11.34%</td>
<td>1.63%</td>
</tr>
</tbody>
</table>

% error = \( \frac{\text{error(abs)}}{\text{Normalization factor}} \) *100

Normalization factor: Mean Value (WCCM)

Average % error is comparable to that seen in literature

Why does it not match perfectly?

Within experimentally observed values
The Bias Problem, Bias modelling using LSSVR

- Estimates of states from EKF state Observer at healthy and MAF fault conditions
- Bias due to imperfections in MVEM
- Bias modelling with LSSVR found to far superior than with polynomial regressors
- Better quality state estimates obtained with knowledge of the bias

### Improved robustness against parameter perturbation with adaptable bias tables

- Store data from new operating points (input and state measurements) in normal course of operation (not a test bed experiment)
- Algorithm to detect if operating point is shifting
- If operating point is steady for more than a period of time, store Bias \( (\text{mean}(Ym) - \text{mean}(X)) \) points.
- Algorithm to check each new Bias point against old points and discard points that are “close” to the new points. Add all new points to existing data set
- Run LSSVR again and remodel Bias

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Old Value</th>
<th>New Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C_{d1} ) (Intake Valve 1)</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>( C_{d2} ) (Intake Valve 2)</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>( C_{d3} ) (Intake Valve 3)</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>( C_{d4} ) (Intake Valve 4)</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>( C_{d1} ) (Exhaust Valve 1)</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>( C_{d2} ) (Exhaust Valve 2)</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>( C_{d3} ) (Exhaust Valve 3)</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>( C_{d4} ) (Exhaust Valve 4)</td>
<td>1</td>
<td>0.6</td>
</tr>
</tbody>
</table>
Major contributions

- Derivation of Extended MVEM from WCCM, improved accuracy using bias tables
- Adaptable Bias tables for improved model robustness to parameter perturbation
Thank You