An Experiment on the Design of RBF Networks for Software Cost Estimation

Ali Idri, Ph.D., ENSIAS, Rabat, Morocco
Alain Abran, Ph.D. ETS, Montreal, Canada
Samir Mbarki, FSK, Kenitra, Morocco

2th IEEE International Conference on Information and Communication Technologies: from Theory to Applications,
24-28 April 2006, Damascus, Syria
Outline

- Motivations and Objectives
- Clustering techniques for RBF networks
- Experiment Design and Data Description
- Overview of Empirical Results
- Conclusions and Future Work
Motivations and Objectives

- Software cost estimation is one of the most critical activities in managing software projects
Estimation by ANN is a promising technique to solve the software cost estimation problem:

- It provides learning from historical projects data
- It can model a complex set of relationships between the dependent variable (cost, effort) and the independent variables (cost drivers)

\[
\text{Effort} = \sum_{j=1}^{M} y_j \beta_j \quad \text{with} \quad y_j = e \left( \frac{\|x-c\|^2}{\sigma_j^2} \right)
\]
The use of RBFN to estimate Software development effort requires the determination of the midde-layer parameters:

- Clustering techniques
- Empirical domain knowledge

Objectives

- To discuss the preprocessing phase of the design of RBFN using two clustering techniques: C-means and APC-III algorithms
- To validate the software cost estimation models based on RBFN
Clustering is one method to find most similar groups from given data, which means that data belonging to one cluster are the most similar; and data belonging to different clusters are the most dissimilar.

Clustering techniques may be grouped into major categories: Hierarchical and Partitional.

Partitional clustering algorithms suppose that the data set can be well represented by finite prototypes.
Clustering has been often exercised as a preprocessing phase used in the design of the RBF neural networks.

The primary aim of clustering is to set up an initial distribution of the receptive fields (hidden neurons) across the space of the input variables.

We use two clustering techniques:

- APC-III
- C-means
- APC-III clustering algorithm (Hwang and Bang, 1997)
  - APC-III is a one-pass algorithm unlike C-means
  - It has a constant radius $R_0$
    \[ R_0 = \alpha \frac{1}{N} \sum_{i=1}^{N} \min_{i \neq j} (\|P_i - P_j\|) \]
  - The number of clusters depends on the $\alpha$ and the sequence presentation of the projects

- The C-means algorithm partitions a collection of $N$ vectors into $C$ clusters $C_i$, $i=1,..,c$ to minimize
  \[ J = \sum_{i=1}^{c} \sum_{x_k \in C_i} \|x_k - c_i\|^2 \]
  - C-means is a multipass algorithm
  - The number of clusters is fixed in the beginning
The experiment uses the COCOMO’81 dataset

This dataset contains 263 historical software projects

- Each project is described by 13 attributes
  - Software size measured in terms of KDSI
  - 12 attributes related to the software development environment such as software complexity, the method used in the development and the time and storage constraints imposed on the software

- Number of input neurons is the same as the number of cost drivers, say 13

- The number of hidden neurons is determined by the number of clusters (c) provided by the APC-III or the C-means algorithms
In addition, the configuration of the RBFN depends also on the widths $\sigma_i$ used by the hidden units.

In the literature, the widths were determined to cover the input space as uniformly as possible.

- **Consequence**: The RBFN will be able to generate an estimate for a new project even though it is not similar to any historical project.

- **Preference**: The RBFN does not provide any estimate than one it may easily lead to wrong managerial decisions.

- **Our strategy**:

\[
\sigma_i = \begin{cases} 
\max_{x_j \in C_i} d(x_j, c_i) & \text{if } \text{card}(C_i) > 1 \\
\max_{k/\text{card}(C_k) > 1} \sigma_k & \text{if } \text{card}(C_i) = 1
\end{cases}
\]

\[
\sigma_i = \begin{cases} 
\max_{x_j \in C_i} d(x_j, c_i) & \text{if } \text{card}(C_i) > 1 \\
\min_{k/\text{card}(C_k) > 1} \sigma_k & \text{if } \text{card}(C_i) = 1
\end{cases}
\]
Empirical Results

- The calculations were made using two software prototypes developed with C language under a Microsoft Windows PC environment.
  - The first software prototype implements the APC-III and the C-means clustering algorithms, providing both the clusters and their centers from the COCOMO’81 dataset.
  - The second software prototype implements a cost estimation model based on an RBFN architecture in which the middle-layer parameters are determined by means of the first software prototype.

- We have conducted several experiments with both the C-means and the APC-III to decide on the number of hidden units.
Choosing the ‘best’ classification to be used in the RBFN depends on the following two criteria:

- It improves the accuracy of the RBFN
- It provides coherent clusters, i.e. the software projects of a given cluster have satisfactory degrees of similarity

The accuracy of the estimates is evaluated by:

- **MMRE**

\[
MMRE = \frac{1}{N} \sum_{i=1}^{N} \left| \frac{Effort_{actual,i} - Effort_{estimated,i}}{Effort_{actual,i}} \right| \times 100
\]

- **Pred (0.25)**

\[
Pred(p) = \frac{k}{N}
\]
○ Accuracy of an RBFN with C-means

- The coherence of clusters is measured by the means of two criteria: the objective function $J$ or the Dunn’s validity index defined by the following formula:

$$D_1 = \min_{1 \leq i \leq c} \left( \min_{i+1 \leq j \leq c-1} \left( \frac{d(C_i, C_j)}{\max_{1 \leq k \leq c} d(C_k)} \right) \right)$$

![Graph showing the relationship between number of clusters and accuracy metrics](graph.png)
Accuracy of an RBFN with APC-III is better when $\alpha$ is lower than 1.04 (MMRE<30 and Pred (25)>70)

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>Number of clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>0.5</td>
</tr>
<tr>
<td>0.9</td>
<td>1.0</td>
</tr>
<tr>
<td>1.06</td>
<td>1.08</td>
</tr>
</tbody>
</table>

![Graph showing Pred and MMRE values]
Comparing the accuracy of RBFN using C-means vs APC-III
Conclusion and Future Work

- We have studied the use of two clustering techniques when designing an RBFN for software cost estimation.
- The two used clustering techniques are C-means and APC-III.
- The RBFN designed with C-means performs better, in terms of cost estimates accuracy, than the RBFN designed with the APC-III algorithm.

Further work:

- Applying an RBFN construction based C-means on other historical software projects datasets.
- Using different datasets for training and testing an RBFN in software cost estimation.
Thank you