Comparing Cost Prediction Methods for Apartment Housing Projects: CBR versus ANN

Sang-Yong Kim*,1, Jae-Won Choi2, Gwang-Hee Kim3 and Kyung-In Kang4

1 M.S, Dept. of Architectural Engineering, Korea University, Korea
2 Junior Employee, Hyundai Engineering and Construction, Korea
3 Professor, Dept. of Architectural Engineering, Mokpo National University, Korea
4 Professor, Dept. of Architectural Engineering, Korea University, Korea

Abstract
Prediction of the cost estimation of apartment house is an important task in the management of construction projects. This study aims at illustrating the compared results of the application of two different approaches which are used case-based reasoning (CBR) and artificial neural networks (ANN) techniques. This study is conducted by using the same 540 cases which are obtained in Korea. 30 cases among the data are used for testing. Testing error rates of 3.69% in the CBR and 6.52% in the ANN were obtained. Results showed that CBR can produce slightly more accurate results and achieve higher computational efficiency than ANN. If the use of CBR and ANN is understood better, as a result, cost estimation can be predicted with reasonability, all parties involved in the construction process could save considerable money.

Keywords: apartment house; cost estimation; case-based reasoning; artificial neural networks

1. Introduction
Estimation of the cost of a construction project is an important task of construction project. The quality of construction management depends on the accurate estimation of the construction cost. But the information of project is limited, so it is very difficult to estimate the construction cost at the early stage in construction project.

Therefore several cost modeling methods such as regression analysis have been developed in order to estimate the initial construction cost of a project by using the historical data of construction project. However, the development of these models is a difficult task due to the inherent limitations of regression analysis. Most of the major disadvantage of regression-based techniques is their unsuitability to account for the large number of variables present in a construction project and the numerous interactions among them. These limitations have contributed to the low accuracy of traditional models and their limited use in construction (Garza and Rouhana, 1995). Another disadvantage is their requirement of a defined mathematical form for the cost function that best fits the available historical data (Creese and Li, 1995).

Case-Based Reasoning (CBR) and Artificial Neural Networks (ANN) may help to predict the cost estimation of construction that is normally affected by a large number of factors. For example, in this study reported later in the present paper, 12 factors were considered which were categorical variables and numeric variables. The assessment of the impact of the many combinations of these many factors is very difficult using conventional methods such as the systematic human review of documentation retrieved from court archives. The purpose of this study was to compare the accuracy of two estimating techniques (CBR and ANN) in estimating construction cost. The next section briefly describes these two estimating techniques. In next section, the collected cost data are applied to the two approaches and the effectiveness of the estimating techniques in terms of estimation accuracy is discussed. The final section includes our conclusion and suggestions for further research.

2. Methodology of Study
2.1 Case-Based Reasoning
CBR means using old experience to understand and solve new problems. In CBR, a reasoner remembers a previous situation similar to the current one and uses that to solve the new problems. CBR can mean adapting old solutions to meet new demands; using old cases to explain new situations; using old cases to critique new solutions; or reasoning from precedents to interpret a new situation (much like lawyers do) or create an equitable solution to a new problem (much like labor mediators do) (Kolodner 1993).

One way this is often visualized is in terms of the problem space and the solution space. In Fig. 1, you can
see that an individual case is made up of two components: problem description and a stored solution. These reside respectively in the problem space and the solution space. The description of a new problem to be solved is positioned in the problem description (the arrow labeled “R” in Fig. 1), and its stored solution is found. If necessary, adaptation occurs (the arrow labeled “A”) and a new solution is created. This conceptual model of CBR assumes that there is a direct one-to-one mapping between the problems and solution spaces. In other words, if a new problem is “down and to the left” of a known problem, then the new solution will also be “down and to the left” of the retrieved problem’s solution (Watson 1997).

At the highest level of generality, a general CBR cycle may be described by the following 4“Re”s:

1. Retrieve the most similar case or cases
2. Reuse the information and knowledge in that case to solve the problems
3. Revise the proposed solution if necessary
4. Retain the parts of this experience likely to be useful for future problem solving

A new problem is solved by retrieving one or more previously experienced cases, reusing the case in one way or another, revising the solution based on reusing a previous case, and retaining the new experience by incorporating it into the existing knowledge-base. The four processes each involve a number of more specific steps, which will be described in the task model. In Fig. 2, this cycle is illustrated.

CBR model typically contains five essential processes: case representation, case indexing and retrieval, adaptation, storage, and evaluation, repair and test. Each process is described as follows:

1. Case representation
   A case is a contextualized piece of knowledge representing an experience. It contains the past lesson that is the content of the case and the context in which the lesson can be used. A case can be an account of an event, a story, or some record typically comprising.
2. Case indexing and retrieval
   Most database systems use indexes to speed up the retrieval of data. An index is a computational data structure that can be held in memory and searched very quickly. CBR also uses indexes to speed up retrieval. A retrieve is a technique developed by machine learning researchers to extract rules or construct decision trees from the past data. In CBR systems, the case-base is analyzed by a retrieval algorithm to produce decision trees that classifies the cases. Three major types of indexing have been applied (often in combination). They include: Inductive retrieval methods are best when the retrieval goal is well-defined. Cases are indexed based on the most important features affecting the outcome as induced from the data itself. The resulting decision trees provide for considerably faster retrieval times than nearest-neighbor retrieval. Knowledge-based retrieval applies existing domain knowledge to locate relevant cases. This approach is similar to rule-based expert systems. Nearest-neighbor matching retrieves cases based on a weighted sum of features in the input cases. The cases with the “closest” overall match according to some similarity metric are returned from the match process. This approach is best if the retrieval needs are not focused tightly on solving a specific problem.
3. Adaptation
   Once a matching case is retrieved, a CBR system will attempt to reuse the solution suggested by the retrieved case. The CBR system must then adapt the solution stored in the retrieved case to the needs of the current case.
(4) Storage
The storage should reflect the conceptual view of what is represented in the case and take into account the indexes that characterize the case. The case-base should be organized into a manageable structure that supports efficient search and retrieval methods.

(5) Evaluation, repair and test
An evaluation takes the results from applying the solution in the real environment. The results from applying the solution may take some time to appear, depending on the type of application. Case repair involves detecting the errors of the current solution and retrieving or generating explanations for them. This task uses the failure explanations to modify the solution in such a way that failures do not occur.

2.2 Artificial Neural Networks
ANN are formed from hundreds or thousands of simulated neurons that are connected in much the same way as the brain’s neurons and are thus able to learn in a similar manner to people. The purpose of the neural networks is to apply various field in engineering using an artificial brain.

Fig. 3 shows simple two-layer network architecture. The NN has an input buffer to which data is presented to the network. And an output layer which holds the response of the network to a given input. Layers distinct from the input layer and the output layer are called hidden layers. Each layer calculates its own output by receiving the weighted sum of its inputs. The strength of connection is called a weight.

There are two main phases in the operation of a NN: learning and recall. Learning is the process of adapting the connection weights in response to a number of examples being presented at the input layer and, optionally at the output layer. The task is to arrive at a unique set of weights that are capable of correctly associating all example patterns used in learning, with their desired output patterns. The most common NN model is the multi-layer perceptron (Hegazy et al. 1994). The multi-layer perceptron is known as a supervised network because supervised learning refers to the case in which the network is presented with some input examples and their desired responses. The desired outputs are used in this case to teach the network the correct responses. The multi-layer perceptron learns using an algorithm called back-propagation (Rumelhart et al. 1986). Back-propagation incorporates a learning algorithm called “generalized delta rule” (Rumelhart et al. 1986), which is responsible for training the network, usually over thousands of iterations. The algorithm uses a Gradient Descent Method (GDM) to determine a unique set of network weights that enables the network to produce outputs that are very close to the desired outputs associated with a number of training examples. Local minima, on the other hand, refers to the case when the weights settle on a less than optimum status. If the network training is successful, the network can be recalled by applying a set of inputs to the network. Then, the network is expected to produce outputs that are satisfactorily close to the desired set of output used in training. And it incorporates a nonlinear “sigmoid” transfer function for its processing elements.

3. Case Study: Apartment Building’s Cost Data
The data used in the case study were collected from 540 constructed in South Korea, from 1997 to 2001. The data of building can be obtained at the early project stage is extracted from the data. The data is converted by applying the building cost index (Table 1) on the base year of 1995.

| Year | Building cost index |
|------|---------------------|
| 1995 | 100.00              |
| 1996 | 106.72              |
| 1997 | 111.86              |
| 1998 | 121.15              |
| 1999 | 116.61              |
| 2000 | 116.84              |
| 2001 | 118.23              |
The building cases that can be obtained at the early project stage is extracted from the historical data and it is adopted as the variable. The details of the applied variables are as follows in the Table 2.

| Variable       | Description         | Type       | Value          |
|----------------|---------------------|------------|----------------|
|                |                     |            | Min. | Max.  |
| Location       | Categorical         | Kangnam,  | 363  | 5,777|
|                |                     | Kangbuk, Kyeonggi, |     |      |
| Area(m²)       | Numerical           | 7          | 27              |
| Story          | Numerical           | 2          | 12              |
| Roof types     | Categorical         | Flat, Pitched, Steel-frame |   |      |
| Total unit     | Numerical           | 14         | 387             |
| Unit per story | Numerical           | 2          | 20              |
| Average area of unit | Numerical         | 13         | 85              |
| Foundation types | Categorical     | Mat, Pile, Mat + Pile |   |      |
| Usage of basement | Categorical    | Pit floor, Base floor |  0 |      |
| Finishing grades | Numerical        | 5          | 100             |
| Duration (months) | Numerical      | 12         | 40              |
| Output         | Actual direct cost (US$) | Numerical | 3,641 | 55,404 |

### 3.1 Case-Based Reasoning (CBR)

To develop the CBR model, *version 1.4 Esteem* software was chosen for this study.

This study used e.g. (1) as a similarity function to calculate the similarity between the new problem and cases in the case base. In e.g. (1), \( S_I \) stands for the similarity index, \( n \) denotes the number of features in each case, \( W \) represents each feature’s importance weight, and \( SS \) is each feature’s similarity score.

\[
S_I = \frac{\sum_{i=1}^{n} (W_i \times SS_i)}{\sum_{i=1}^{n} (W_i)} \times 100 \quad \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdOTS

The similarity index \( S_I \) is a calculated numerical value which demonstrates the degree of similarity between a case in the case base and the problem case. \( S_I \) is normalized into a scale of 0~100 for easy comparison. Each case in the case base has one similarity index value for every new problem. Notably, a higher similarity index value indicates that the particular case closely resembles to the problem case.

The importance weight \( W \) of a feature can be either assigned by the user or automatically calculated by various existing algorithms (Kolodner, 1993).

This study applies the Gradient Descent Method (GDM) and Regression analysis to generate each feature’s importance weights (Yau, 1998). The both methods used herein can obtain a set of weights for all features. But the GDM begins by randomly selecting cases and might get stuck at local maxima in the weight generation process (Esteem 1995). Therefore two methods were used to develop model for generating each feature’s importance weights.

The software used for regression analysis is *SPSS for Window Release 10*. And using the method to input the variables stepwise was used. Table 3 summarizes the results of regression analysis and calculated weights value by using absolute vale that takes unstandardize coefficients (B) as a standard.

| Model                | Unstandardized coefficients | Standardized coefficients | t   | Sig. |
|----------------------|----------------------------|---------------------------|-----|------|
| Constant             | -54488                     | 10673.083                 | -5.105 | 0.000|
| Location             | 121.037                    | 2.502                     | 0.892 | 0.4834| 0.000|
| Area(m²)             | 1686.534                   | 85.947                    | 0.252 | 19.621| 0.000|
| Roof types           | 1734.967                   | 293.226                   | 0.061 | 5.917 | 0.000|
| Total unit           | -12331.2                   | 2224.769                  | -0.030 | -2.832| 0.000|
| Unit per story       | -10134.1                   | 2284.508                  | -0.044 | -4.436| 0.000|
| Foundation types     | 13417.91                   | 2984.483                  | 0.051 | 4.496 | 0.000|
| Usage of basement    | -991.489                   | 195.763                   | -0.095 | -5.065| 0.000|
| Duration (months)    | -290.762                   | 70.639                    | -0.089 | 2.711 | 0.000|

The similarity score \( SS \) is determined on the basis of the features’ values. Two types of values are in a feature: numerical and categorical. For the categorical type, the similarity score \( SS \) equals “1” when the two values are identical; otherwise, \( SS \) equals “0”. For the numerical type, \( SS \) is determined by e.g., (2) and (3), where \( V_{problem} \) denotes the value of the problem’s feature, \( V_{case} \) represents the value of a case’s feature in the case base, and \( M\% \) is the matching range. The matching range could be different for each numerical type feature. For simplification reasons, the matching ranges of all numerical type features are arbitrarily defined as 10% by the *Esteem* software in this study.

\[
\text{IF } |V_{case} - V_{problem}| \leq V_{problem} \times M \% \quad \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots (2)
\]

Then \( SS = 1 \)

\[
\text{IF } |V_{case} - V_{problem}| > V_{problem} \times M \% \quad \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots \cdots (3)
\]

Then \( SS = 0 \)

After analyzing GDM and regression analysis, this study evaluated the methods with regard to the test results.
Result of the tests conducted on the models was the average of error rate of GDM and of regression analysis. As shown in Table 5 and 6, regression analysis is better than that of GDM.

Table 6. The Confidence of Results

|          | Mean error | Std. deviation | Std. error | 95% Confidence interval of the difference |
|----------|------------|----------------|------------|------------------------------------------|
| GDM      | 4.94       | 3.83           | 0.70       | -1.44 - 1.43                             |
| Regression | 3.68       | 3.80           | 0.69       | -1.42 - 1.42                             |

Table 4. Comparisons of Weights of GDM and Regression

|          | GDM | Regression |
|----------|-----|------------|
| Location | 0.1076 | 0.3296 |
| Area(m²) | 0.1206 | 0.0029 |
| Story    | 0.1091 |           |
| Roof types | 0.1044 | 0.0426 |
| Total unit | 0.1518 | 0.0071 |
| Unit per story | 0.1040 |          |
| Average area of unit | 0.1063 | 0.0243 |
| Foundation types | 0.0540 | 0.3029 |
| Usage of basement | 0.0308 | 0.2489 |
| Finishing grades | 0.0200 |          |
| Duration | 0.0908 | 0.0414 |

Table 5. Comparisons of Results of GDM and Regression

| No | Direct cost (US$) | GDM | Regression |
|----|-------------------|-----|------------|
|    | Expect cost (US$) | Error rate (%) | Expect Cost (US$) | Error rate (%) |
| 1  | 14,754,333        | 15,039,250 | 1.93 | 14,743,333 | 0.07 |
| 2  | 22,869,917        | 21,346,583 | 2.39 | 21,645,250 | 1.03 |
| 3  | 34,942,667        | 31,887,667 | 8.74 | 34,621,666 | 0.92 |
| 4  | 22,642,583        | 21,465,417 | 5.20 | 22,517,416 | 0.55 |
| 5  | 19,692,167        | 19,457,083 | 1.19 | 20,273,166 | 2.95 |
| 6  | 23,290,917        | 23,469,917 | 0.77 | 24,457,166 | 5.01 |
| 7  | 23,384,667        | 23,666,000 | 1.20 | 23,566,416 | 0.78 |
| 8  | 24,966,500        | 26,819,083 | 7.42 | 24,640,000 | 1.31 |
| 9  | 13,214,583        | 13,429,750 | 8.44 | 14,746,250 | 11.60 |
| 10 | 22,372,500        | 20,683,583 | 7.55 | 23,951,500 | 7.06 |
| 11 | 36,416,167        | 35,809,667 | 1.67 | 35,808,166 | 1.67 |
| 12 | 38,562,500        | 36,125,333 | 6.32 | 39,082,416 | 1.35 |
| 13 | 16,832,667        | 19,302,083 | 14.67 | 19,116,416 | 13.57 |
| 14 | 25,328,833        | 25,393,833 | 0.26 | 25,328,750 | 0.00 |
| 15 | 22,354,083        | 20,878,167 | 6.60 | 21,891,250 | 2.07 |
| 16 | 13,702,750        | 14,123,750 | 3.07 | 14,593,833 | 6.50 |
| 17 | 23,720,083        | 23,887,167 | 0.70 | 23,775,250 | 0.23 |
| 18 | 40,760,917        | 38,939,667 | 4.47 | 37,222,000 | 8.68 |
| 19 | 22,341,583        | 25,201,750 | 12.80 | 23,273,916 | 4.17 |
| 20 | 22,431,500        | 23,353,417 | 4.03 | 23,273,916 | 3.76 |
| 21 | 21,142,833        | 19,845,167 | 6.14 | 20,639,750 | 2.38 |
| 22 | 21,751,833        | 22,848,333 | 5.04 | 21,815,583 | 0.29 |
| 23 | 20,565,083        | 20,724,000 | 7.70 | 20,330,333 | 0.17 |
| 24 | 27,038,500        | 24,406,417 | 9.73 | 24,505,833 | 9.37 |
| 25 | 33,343,083        | 31,145,083 | 6.59 | 32,572,666 | 2.31 |
| 26 | 16,281,667        | 16,311,250 | 0.18 | 15,587,000 | 4.27 |
| 27 | 13,734,570        | 12,537,583 | 8.85 | 13,067,250 | 5.00 |
| 28 | 7,052,667         | 6,528,167 | 7.44 | 6,294,083 | 10.76 |
| 29 | 25,118,500        | 25,250,250 | 0.52 | 25,080,750 | 0.15 |
| 30 | 34,430,250        | 33,247,500 | 3.43 | 33,494,750 | 2.72 |
| mean | 4.94 | 3.83 | 0.70 | 1.43 |

3.2 Artificial Neural networks (ANN)

To develop the ANN model, NeuroShell 2 software was chosen for its ease of use, speed of training, and host of NN architectures, including back-propagation with flexible user selection of training parameters. The user has the ability to specify the learning rate, momentum, activation functions, and initial weight range. NeuroShell 2 also has multiple criteria for stopping the training, in addition to different methods for handling missing data and viewing the weight values during training.

Back-propagation networks tend to be overfitting in training. To avoid overfitting, it is necessary to have a cross-validation dataset which may be improved the prediction of training data or the cross-validation but it is uncertain how accurately the NN can predict with new data. Therefore, the performance of a NN model must be verified using a test dataset. Fig. 5 shows the point sought by a model.

This study divided the collected cost data same as the data for CBR into three parts (Table 7) and input variables too. Because the purpose of this study is to compare the accuracy of two estimating techniques in estimating construction cost. Among the input variables, the region, the roof type, the type of foundation and basement are the categorical variables so they are converted using binary values to eliminate implicit ordering to avoid artificial intervention in the historical data. The actual direct cost is set as the output variable.

Table 7. The Classification of Data

|          | Data number | Rate (%) |
|----------|-------------|----------|
| Training | 483         | 89.4     |
| Cross-Validation | 27         | 5.0      |
| Test     | 30          | 5.6      |
| Total    | 540         | 100.0    |

Designing a NN configuration suitable for a particular problem is highly problem-dependent. The configuration of NN has a huge impact on its performance. The network builder has many degrees of freedom in defining a
configuration for the network. All this flexibility represents a problem: No defined methodology as yet exists for defining the configuration of network. This forces the model developer to try different networks configuration to find an optimal setup. In this study, the trial-and-error process was adapted for determining network parameters which were the number of hidden layers and PEs in hidden layers, the learning rate coefficient and the momentum coefficient. For determining parameters:

Hegazy et al. (1999) proposed that one hidden layer is sufficient to generate an arbitrary mapping between inputs and outputs and the number of neurons in the hidden layer is $0.75m$, $m$ or $2m+1$, where $m$ is the number of input neurons. In this study, three kinds of numbers proposed for the hidden neurons were applied to the NN model.

Considerable time must be spent in determining the coefficients of the learning rate and the momentum, which require a few trial-and-error processes. Hegazy et al. (1994) also proposed that the coefficients of learning rate and the momentum can be set to 0.9 and 0.7, respectively. In this study, both these coefficients were set between 0.6 and 0.9 (in steps of 0.1) to examine their effect and establish the best NN model.

Consequently, 48 NN models were evaluated in this study as the number of neurons in the hidden layer and the coefficients of the learning rate and the momentum were varied. The best structure of the NN was determined to be 3 neurons in the hidden layer, and 0.6 and 0.9 are the learning rate and the momentum coefficient of the back-propagation algorithm. As shown in Table 8, in the estimating results of the established NN models.

### 4. Discussion of results

After analyzing CBR and ANN, this study was conducted by using the same 510 cases. An additional 30 cases were used for the testing. The best prediction result of 96.31% was obtained from the CBR model, whereas the best prediction rate obtained from the ANN model was 93.48%. CBR model result is better than that of ANN.

\[
\text{Error rate} = \frac{\text{Actual Cost} - \text{Predicted Cost}}{\text{Actual Cost}} \times 100
\]

| No | Direct cost (US$) | ANN | CBRL | Error rate (%) | Error rate (%) |
|----|-----------------|-----|-----|----------------|---------------|
| 1  | 14,754,333      | 15,806,500 | 7.13 | 14,743,333 | 0.07 |
| 2  | 22,869,917      | 22,705,917 | 0.72 | 21,640,250 | 1.03 |
| 3  | 34,942,667      | 32,052,250 | 8.27 | 34,621,666 | 0.92 |
| 4  | 22,642,583      | 19,792,000 | 12.59 | 22,517,416 | 0.55 |
| 5  | 19,692,167      | 18,358,917 | 6.77 | 20,273,266 | 2.95 |
| 6  | 23,290,917      | 21,455,917 | 7.88 | 24,457,166 | 5.01 |
| 7  | 23,384,667      | 26,051,333 | 11.40 | 23,564,166 | 0.78 |
| 8  | 24,966,500      | 29,071,333 | 16.44 | 24,640,000 | 1.31 |
| 9  | 13,214,583      | 13,088,500 | 0.95 | 14,746,250 | 11.60 |
| 10 | 22,372,500      | 22,967,667 | 2.66 | 23,951,500 | 7.06 |
| 11 | 36,416,167      | 35,315,583 | 3.02 | 35,808,166 | 1.67 |
| 12 | 38,562,500      | 34,531,083 | 10.45 | 39,082,416 | 1.35 |
| 13 | 16,832,667      | 17,233,000 | 2.38 | 19,116,416 | 13.57 |
| 14 | 25,328,833      | 27,190,833 | 7.35 | 25,287,500 | 0.00 |
| 15 | 22,354,083      | 21,500,917 | 3.82 | 21,892,500 | 2.07 |
| 16 | 13,702,750      | 13,638,667 | 0.47 | 14,593,833 | 6.50 |
| 17 | 23,720,083      | 24,183,833 | 1.96 | 23,775,250 | 0.23 |
| 18 | 40,760,917      | 40,546,750 | 0.53 | 37,222,000 | 8.68 |
| 19 | 22,341,500      | 23,189,917 | 3.77 | 23,273,916 | 4.17 |
| 20 | 22,431,500      | 23,189,917 | 3.36 | 23,273,916 | 3.76 |
| 21 | 21,142,833      | 23,028,000 | 8.92 | 20,639,750 | 2.38 |
| 22 | 21,751,833      | 23,284,500 | 7.05 | 21,815,583 | 0.29 |
| 23 | 20,565,083      | 21,788,833 | 5.95 | 20,530,333 | 0.17 |
| 24 | 27,038,500      | 27,856,167 | 3.02 | 24,505,833 | 9.37 |
| 25 | 33,343,083      | 30,164,250 | 9.53 | 32,572,666 | 2.31 |
| 26 | 16,281,667      | 18,440,083 | 13.26 | 15,587,000 | 4.27 |
| 27 | 13,754,750      | 11,513,167 | 16.30 | 13,067,250 | 5.00 |
| 28 | 7,052,667       | 7,308,667  | 3.63 | 6,294,083  | 10.76 |
| 29 | 25,118,500      | 26,653,417 | 6.11 | 25,080,750 | 0.15 |
| 30 | 34,430,250      | 38,275,583 | 11.17 | 33,494,750 | 2.72 |

A comparison of the result with the development of the CBR and ANN models shows following:

In the ANN model, the addition of new cases is quite difficult because the model needs to be retrained whenever new cases are added. This is a very long process because each of the parameters and algorithms used should be tested again, and the results are expected to change because of the additional cases. On the other hand, CBR is more flexible in updating the new cases. Also, CBR is good with categorical data and much better with complex, structured numerical data.

CBR is quite successful in handling missing cases. Although the cases in the case-base and the target case may contain missing cases, this does not prevent the model from producing answers. In these situations, the model may give lower similarity scores, telling the
analyst that the missing cases are part of the solution, or sometimes the system cannot produce an answer. In the ANN model, the training set should be complete and sound; if the target case contains missing cases the model may still produce a result but does not indicate that there is a great chance of error.

Table 10. The Confidence of Results

|        | Mean error | Std. deviation | Std. error | 95% Confidence interval of the difference |
|--------|------------|----------------|------------|-----------------------------------------|
| ANN    | 6.66       | 4.45           | 0.81       | Lower: -1.67, Upper: 1.65               |
| CBR    | 3.68       | 3.80           | 0.69       | Lower: -1.42, Upper: 1.42               |

5. Conclusion

This study presents a CBR and ANN cost estimation that facilitates decision making at the early project stage. For the study, 540 apartment buildings constructed in Seoul, Korea between 1997 and 2001 were used in training and verifying the models.

The contributions to the estimation of construction cost in the construction management made by this dissertation include the following:

First, the combined of estimating data and AI techniques use in new and innovative ways such as CBR and ANN.

Second, this advances the field of AI applications in the construction management, which is known to lack behind other industries in this area. This will aid practitioners in the construction management to perform estimating within the short time available to study a project.

Third, in the present study these two methods were tested, and CBR was found to be more successful in predicting the estimation of construction cost. Thus, it is proved more effective CBR than ANN in estimating the construction cost of apartment building in the early project stage. In these respects, the CBR model can be more useful for estimating construction cost.

Finally, a similar CBR approach can be applied to other domains in the experience oriented construction industry if previous construction cases are available. In the CBR model, however, the decision of weights heavily influences results of such a CBR approach.

Despite that this research introduces the useful methods as a tool for cost estimators to use in preparing conceptual cost estimates for the Korean construction management, it is clear that this study can be potentially enhanced by future works. These may include:

First, CBR model heavily influences results of such a decision of weights. Therefore, further study is needed to develop weights the various tools.

Second, many kinds of AI models to forecast the cost items of the early cost estimate models such as hybrid with genetic algorithms and other AI methods have to discuss for improving irregular patterns in construction management. A future investigation should more closely examine this study.

Third, it is available that CBR and ANN have to include more project cases and variables in the model’s database for improving accuracy of cost estimate.

Finally, detailed cost database models so that the user will be able to prepare all types of estimates in accordance with the different phases of the projects.

References

1) Arditi, D., Oksay, F. E., and Tokdemir, O. B. (1998). “Predicting the outcome of construction litigation using neural networks.” *CompAided Civ. And Infrastruct. Engrg.* 13(2), 75-81.
2) Arditi, D., and Tokdemir B. (1999). “Comparison of case-based reasoning and artificial neural networks.” *J. Comp.n Civ. Engrg.*, ASCE. 13(3), 162-169.
3) Attalla, M., and Hegazy, T. (2003). “Predicting cost deviation in reconstruction project : Artificial neural networks versus regression.” *J. Constr. Engrg. and Mgmt.*, ASCE, 129(4), 405-411.
4) Bode, J. (2000). “Neural networks for cost estimation: simulations and pilot application.” Int. J. Production Research, 38(6), 1231-1254.
5) Boussabaine, A. H. (1996). “The use of artificial neural networks in construction management: a review.” *Constr. Mgmt. and Economics*, 14, 427-436.
6) Creece, R., and Li, L. (1995). “Cost estimation of timber bridge using neural networks.” *Cost Engrg.*, AACE Int., 37(5), 17-22.
7) Esteem software (1995). "ESTEEM Software Documentation." *Version 1.4 Cambridge Civ. Engrg.*
8) Garza, J., and Rouhana, K. (1995). “Neural network versus parameter-based application.” *Cost Engrg.*, AACE Int., 37(2): 14-18.
9) Hegazy, T., and Ayed, A. (1998). “Neural network model for parametric cost estimation of highway projects.” *J. Constr. Engrg. and Mgmt.*, ASCE, 124(3), 210-218.
10) Hegazy, T., and Moselhi, O. (1994). “Analogy-based solution to markup estimation problem.” *J. Comp.n Civ. Engrg.*, ASCE, 8(1), 72-87.
11) Hegazy, T., Moselhi, O., and Fazio, P. (1994):“Developing practical neural network applications using back-propagation.” *Microcomputers in civ. Engrg.*, Blackwell Publisher, 9(2): 145-159.
12) Kim, G. H., An, S. H., and Kang, K. I. (2004), “Comparison of construction cost estimating models based on regression analysis, neural networks, and case-based reasoning.” *Building and Environment*, 39(10): 1235-1242.
13) Kolodner, J. (1993). “Case-Based Reasoning.” Morgan Kaufmann Publisher Inc.
14) McCaffer, R. (1975). “Some analysis of the use of regression as an estimating tool.” *Quantity surveyor*, 32, 81-86.
15) Mckim, R. (1993). “Neural network application to cost engineering.” *Cost Engrg.*, AACE Int., 35(7), 31-35.
16) Moselhi, O., and Hegazy, T. (1991). “Neural networks as tools in construction.” *J. Constr. Engrg. and Mgmt.*, ASCE, 117(4), 606-625.
17) Skitmore M. (1987). “The effect of project information on the accuracy of building price forecasts.” In: Brandon PS, editor *Building cost modeling and computers. London: E&FN Spon*, 327-336.
18) Smith, A. E., and Mason, A. K. (1997). “Cost estimation modeling: regression versus neural network.” *Engrg. Economist*, 42(2), 137-161.
19) Smith, S. D. (1999). “Earthmoving productivity estimation using linear regression techniques.” *J. Constr. Engrg. and Mgmt.*, ASCE, 125(3), 133-141.
20) Trost, S. M., and Oberlender, G. D. (2003). “Predicting accuracy of early cost estimates using factor analysis and multivariate regression.” *J. Constr. Engrg. And Mgmt.*, ASCE, 129(2), 198-204.
21) Tzafestas, S. G., Dalianis, P. J., and Anthopoulos, G. (1996). “On
the overtraining phenomenon of backpropagation neural networks.”

Math. and Comp. in Simulation, 40: 507-521.
22) Wang, Q., Stockton, D. J., and Baguley, P. (2000). “Process cost modelling using neural networks.” Int. J. Prod. Res., 38(16), 3811-3821.
23) Watson, I. (1997). “Applying Case-Based Reasoning.” Morgan Kaufmann Publisher Inc.
24) White, H. (1989). “Neural-network learning and statistics.” AI Expert (Dec.), 48-52.
25) Yau, N. J., and Yang, J. B. (1998). “Applying case-based reasoning technique to retaining wall selection.” Automation in Construction 7, 271-283.
26) Yeh, I. C. (1998). “Quantity estimating of building with logarithm-neuron networks.” J. Constr. Eng. and Mgmt., ASCE, 124(5), 374-380.