GESTS International Transactions on Computer Science and Engineering, Volume 28, Number 1, Feb 2006.

Performance Measurement of an Agglomerative Clustering Algorithm for Data Enhancement

Sumanth Yenduri[1], S.S. Iyengar[2]

[1] Assistant Professor
Computer Science Dept., University of Southern Mississippi
Sumanth.Yenduri@usm.edu
[2] Professor and Chairman
Computer Science Dept.,
Louisiana State University
Iyengar@bit.csc.lsu.edu

Abstract. Commercial software project effort estimation remains to be a challenging problem even today because of the inherent discrepancies found in most of the data bases that are used to predict these estimates from. Various imputation strategies have helped to better the prediction accuracies of these estimates. We have implemented an agglomerative clustering technique to impute data in six real-time data sets and there by predict better effort estimates. The methodology is described briefly. In this paper, we analyze the quality metrics of the clustering achieved and there by validate the performance of the algorithm. Finally, we discuss and elaborate our findings.

1 Introduction

In spite of the enormous amount of research done, predicting accurate effort estimates for software projects still remains to be an exigent problem. Most statistical procedures in use require historical data bases of past projects. Data are required to analyze and construct "true" models using which effort estimates for future projects can be predicted. But collection of data throughout the life cycle of a software project requires non-trivial amounts of effort, cost and time [1, 2, 3]. It is more severe in data collected through on-site surveys [4]. Moreover, collection of data over a long period time introduces inconsistency and errors. We cannot avoid this situation [5, 6]. Therefore, most of these data bases have significant amounts of missing information. Current practices used in the software engineering community ignore all the missing information and predict estimates based on the remaining information by using data ignoring methods such as Listwise Deletion and Pairwise Deletion. Obviously, the estimates are biased. More over these methods increase the percentage of missingness. It has been proven in numerous studies that using these kinds of ignoring methods

amount to huge data losses. The findings of Kim and Curry have shown that 2% of missing values in each of the 10 variables has amounted to 18.3% of total data loss on average when using listwise deletion. 10% of missing values in 5 variables amounted to 41% of data loss (when using listwise deletion) [7]. Such kinds of huge data losses could seriously affect prediction accuracies. Accurate effort estimates help in cost reduction, risk management, resource allocation and timely completion of the software projects.

Our intent is to improve the use of commercial software project data sets using hybrid methodologies by filling in probable values for the missing data. By analyzing data sets having "fuller information" and building "truer models", we believe there is a high likelihood of enhancing the accuracies of the effort estimates. There are many applications that need such collections of data sets but we concentrate on enhancing software project data sets. Our study benefits both project managers in the industry and researchers in the academia. In this paper we describe the design of the hybrid methodology briefly and then proceed to elaborate on the metrics used to evaluate the clustering quality achieved in the proposed agglomerative clustering algorithm. We report and discuss our findings with respect to three different quality measures. We perform useful experimental analyses and evaluate the impact of the methodology. Henceforth, we validate the performance of our clustering algorithm. Finally, we discuss the appropriateness of the methodology. The reliability of the constructed data sets using these techniques was further tested by building prediction models using stepwise regression which is not the scope of this paper though [8]. So far, little research has been done in exploring the implications of applying data imputation methods to software project data sets. There are a few references in the literature related to such exploration [9, 10, 11, 12]. All of them have quoted a significant increase in the prediction accuracy of estimates when different kinds of imputation methods were used. But all of them stress there still remains a great deal of research to be done on this topic for more concrete answers [9, 10, 11, 12, 13, 14, 15].

2 Methodology Design

We implement a hybrid methodology to overcome the limitations common to most traditional imputation methods. The methodology was designed by taking into aspect the missing mechanism, data set size, missing percentage and the pattern in which the data are missing. It imputes data by creating multiple homogenous clusters. It works in two phases. It creates homogenous clusters and then imputes the missing values by selecting the appropriate donors from the created clusters.

First the given data set is divided into 2 sets, complete (with no missing information) and incomplete. The clustering algorithm is implemented on the complete data set to form multiple homogenous clusters or "similar type" clusters. A hierarchical agglomerative clustering algorithmic approach is used to form clusters which contain one or more "similar" cases. They are bottom-up approaches in which each case is considered an individual cluster and at each step, the most similar pair of clusters is merged together. Cluster similarity is calculated using a distance measure. The algorithm starts with n clusters each representing the n cases in the complete data set for

which a symmetric similarity matrix is generated. The entries of the matrix represent the similarity (is a distance metric) between the cluster pairs. The matrix is searched for the most similar pair or in other words the matrix entry having the least value is found and the corresponding pair is merged to form another new cluster. The similarity distances between the newly formed cluster and the remaining clusters are updated in the matrix. Again the matrix is searched for the most similar pair and this goes on again and again until one huge cluster that contains all the cases is formed. The complexity of the algorithm is $O(n^2 logn)$. Average Linkage Agglomerative Clustering Algorithm was used in our approach. In this method, the distance between two clusters is defined as the average of distances between all pairs of cases, where each pair is made up of one case from each group. The average distance $d(i,j)$ computed at each level is given by the following equation:

$$d(i, j) \text{ is computed as } d(i, j) = D_{ij} / (n_r * n_s)$$

Where D_{ij} is the sum of all pairwise distances between cluster i and cluster j. n_r and n_s are the sizes of the clusters i and j respectively. At each stage of hierarchical clustering, the clusters i and j, for which d_{ij} is the minimum, are merged. The distance D measured is the Euclidean Distance. Next, it selects the donors from the clusters in order to impute missing data. The number of clusters to be formed though is decided by considering the first quality metric Miss-Assignment Count explained in the later sections.

Once the clusters are formed, missing values for each case are imputed by selecting donors from that particular cluster that they most probably would belong to. The cluster that would contribute the donor(s) is determined by calculating a proximity metric for each missing case, which determines the donating cluster. By creating homogenous clusters and selecting the most appropriate cluster for a particular incomplete case using a proximity metric makes sure that the incomplete case gets the most suitable donor(s) which is extremely important. This is done by first calculating the centroid vector for each cluster. The Euclidean distance between each missing case and the centroid of each cluster gives the proximity metric. The cluster representing the centroid vector with which the incomplete case gives the smallest proximity metric value is selected. Moreover, the selection process of the donors implemented in the methodology is very significant as it liberates the methodology from the hurdles caused by the inherent characteristics of a data set. After selecting the appropriate donating cluster, we implement our Combination Method in order to select the most similar donor(s). We designed the Combination Method so that it works for both qualitative and quantitative variables. From within the cluster, the donors are selected using the *k*-Nearest Neighborhood (Combination Method). The method works by finding "*k*" most similar/nearest complete cases to the incomplete case where the similarity is measured by a distance parameter (Cosine Distance (Quantitative Variables) and Hamming Distance (Qualitative Variables)). The value of "*k*" was set to 2. That is, 2 most similar/nearest cases were selected to impute the values in the incomplete case.

It takes into account the input from both kinds of variables by using two metrics for determining the donor(s), which is different from many existing methods. In fact, many existing methods work with only quantitative variables. We implemented our

methodology over six real-time software project data sets and evaluated its performance with a number of existing methods. We acquired six real-time software project data sets in the past one year period from six different companies nationally and internationally. We obtained three small sized software project data sets, two medium sized and one large sized data set. They all differ in characteristics such as missing mechanism, size, physical missing pattern, percentage of missing data etc which is visible in table 1.

Table 1. The real-time data sets used in the experimental analysis

Data Set	Size	Project Type	Time (years)	Missing Mechanism	% of missing data (rounded)	Missing Pattern
D1	S	Medical	5	MAR	12	A
D2	S	Customer Service	4	MAR	32	M
D3	S	Web Focus	2	MCAR	4	U
D4	M	Bank	6	MAR	26	A
D5	M	Customer Service	9	MAR	46	A
D6	L	Network Management	10	NI	18	A

Size (S-small, M-Medium, L-Large)
Missing Pattern (U- Univariate, M – Monotonous, A – Arbitrary)
MAR (Missing At Random), MCAR (Missing Completely at Random), NI (Non-Ignorable) [3]

To study the impacts of these methods, the imputed data sets were evaluated using prediction models. A significant step in the construction of a prediction model is the selection of independent variables. We used the Forward Entry Stepwise Regression Model-Building Procedure [16]. To begin with, an initial model is identified. It always includes the regression intercept. Next "iterative stepping" is performed. That is changing the model repetitively by adding or removing a predictor/independent variable, which is based on the "stepping constraints (tests)". Finally the termination procedure is initiated when stepping cannot be done any more or if the maximum number of steps has been reached. Thus the prediction models are built for each of the six real-time data sets.

2.1 Concept of "United Clusters"

As hierarchical agglomerative algorithms start with all cases, they are particularly effective in identifying many small clusters. The quality of clusters deteriorates as more number of mergers is made and hence with such a concept of united clusters, we essentially decrease the number of mergers, thereby increasing cluster quality. We use the concept of "united clusters" specified as "mutual clusters" in [17] which con-

tain a group of cases that are sufficiently close to each other and far from all other cases and hence should never be separated. The united clusters are mutual clusters and in no way are different. We termed them united clusters, as it seemed more appropriate. A united cluster is atomic. We used the same in our methodology.

3 Measuring the quality of clusters formed

The measures of quality let us evaluate how well the clustering has been performed. We use three measures to quantify the quality of the clusters. The measures are oriented towards measuring the effectiveness of the approach [18]. The first measure Miss-assignment count examines whether any of the cases were assigned wrongly to a cluster. To verify this, centroid vectors are calculated for each of the cluster initially. Next, cases having smaller distance to centroid vectors of other clusters when compared to the centroid vector of the cluster they belong to are gathered. Such cases are considered to be wrongly assigned to a cluster.

The next measure is within-cluster distances, which provides a measure of "goodness" for the clusters. It identifies clusters that have minimum within-cluster distances. After the clustering algorithm is run, the dendrogram representing the sequence of partitions of the data set is cut at different levels to form varying numbers of clusters. For each number of clusters, the sum of all pairwise within-cluster (Euclidean Distance) distances is calculated. Good clusterings minimize the sum. The sum of distances is plotted versus the number of clusters formed. Our approach does well when there is more number of clusters. One final measure is to look at the inter-cluster dissimilarity in order to find how different are each of the clusters. For each of the clusters, a centroid vector is initially calculated and the pairwise cosine similarity is measured between all of them. The cosine formula is $\cos(d_i, d_j) = d_i^t * d_j$ where d_i and d_j are centroids vectors of two clusters of unit length. The measure is 1 if the centroids are identical and 0 if they are orthogonal. Inter-cluster dissimilarity should be high for good clustering quality.

3.1 Miss-Assignment Count

Table 2 shows wrongly assigned cases for each of the six data sets. The rows indicate each of the six data sets and the columns indicate the number of clusters taken into account. The number of miss-assigned points is only one for DS 1 and DS 2. The method performed well with no miss-assigned points for DS 3. DS 4 had one miss-assigned point when five clusters were formed and had two miss-assigned points when ten clusters were formed. DS 5 had four miss-assigned points when five clusters were formed, two miss-assigned points when ten clusters were formed, and one miss-assigned point when twelve clusters were formed. The number of miss-assignments is low for the first 5 data sets when compared to DS 6. The number of miss-assigned cases is highest for DS 6 where missing mechanism is under NI conditions. DS 6 had seven miss-assigned points when five clusters were formed, five miss-assigned points when ten clusters were formed, four miss-assigned points when

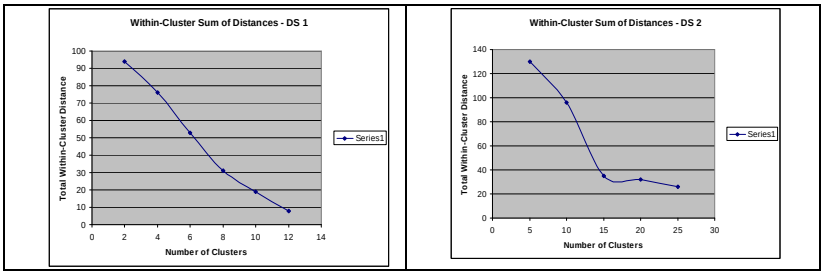
twelve clusters were formed, two miss-assigned points when twenty-five clusters were formed, two miss-assigned points when forty clusters were formed and one miss-assigned point when fifty-five clusters were formed. The clustering algorithm used is a bottom-up approach, which starts the process of clustering by considering each case a new "cluster", and thus forth proceeds. The very nature of this approach makes it a good method in identifying a large number of small clusters, which is highly significant. The overall number of miss-assigned points can be considered low. Even under NI conditions (for DS 6) the number of miss-assigned points was low when twenty-five or more clusters were formed. This metric was used to decide upon the number of clusters to be formed.

Table 2. Number of missassigned cases in each data set for varying number of clusters

	5	**10**	**12**	**25**	**40**	**55**	**80**	**93**
DS 1	0	1	0	-	-	-	-	-
DS 2	0	1	0	0	-	-	-	-
DS 3	0	0	0	-	-	-	-	-
DS 4	1	2	0	0	0	-	-	-
DS 5	4	2	1	0	0	0	-	-
DS 6	7	5	4	2	2	1	0	0

3.2 Within-Cluster Distances

It provides a measure of "goodness" for the clusters by identifying the minimum within-cluster (square root of the sum of squared) distances. For varying number of clusters, the sum of all pairwise within-cluster distances is calculated. Minimized sums represent good clusterings, which means that the cases within a cluster are closer to each other. This metric measures the similarity between the cases in a cluster, said otherwise within-cluster similarity. The proposed approach does well when there is reasonably higher number of clusters. The graphs plotted in fig 1 show us the same.



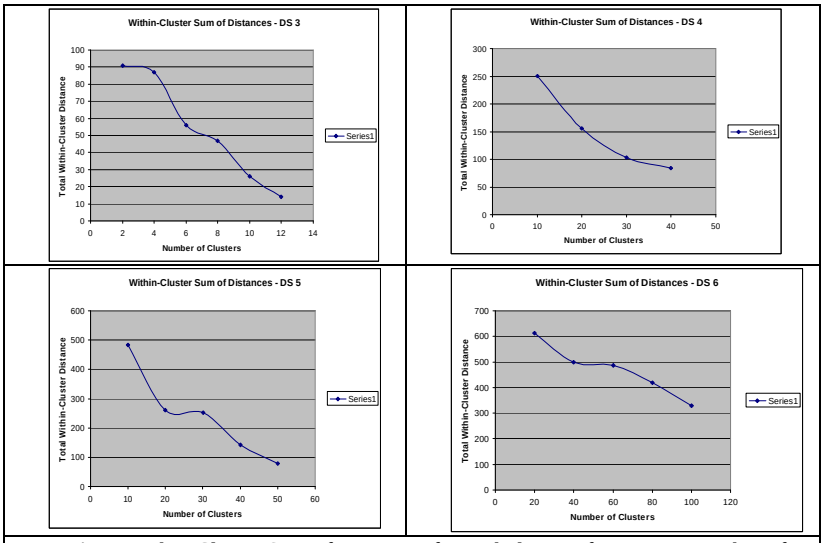


Fig 1. Within-Cluster Sum of Distances for each data set for varying number of clusters

3.3 Inter-Cluster Dissimilarity

It measures the dissimilarity between the clusters formed. The more dissimilar the clusters are, the better the quality of clustering accomplished. The dissimilarity between two clusters is calculated by finding the cosine similarity between their centroids. As the similarity between the clusters increases, the value of the metric approaches 1 and as the dissimilarity increases, the value approaches 0. The degree of orthogonality between the centroids is what the metric calculates. The decision on how many clusters would be ideal was made using the first metric, miss-assignment count. For data sets 1, 2, 3, and 4, 12 clusters were formed and for datasets 5 and 6, 25 clusters were formed. The values ranged from .01-.7, and only 28 readings recorded had values more than .5. This suggests that the clusters were different from one another and hence satisfactory clustering was achieved (Results have been shown only for data sets 1-4 because of space requirements (Table 3)).

Table 3. Inter-Cluster Dissimilarities measured using Cosine Metric for Data Sets 1 to 4

Cluster Pair	DS 1	DS 2	DS 3	DS 4
1-2	.11	.09	.02	.1
1-3	.2	.12	.09	.09
1-4	.32	.13	.13	.15
1-5	.06	.26	.06	.38
1-6	.12	.05	.08	.25

1-7	.24	.03	.11	.06
1-8	.31	.09	.22	.21
1-9	.27	.28	.25	.16
1-10	.14	.39	.05	.09
1-11	.16	.42	.06	.28
1-12	.13	.12	.18	.17
2-3	.26	.18	.06	.05
2-4	.4	.11	.19	.41
2-5	.32	.2	.17	.52
2-6	.16	.09	.16	.19
2-7	.29	.05	.06	.26
2-8	.17	.07	.01	.09
2-9	.09	.02	.09	.13
2-10	.1	.25	.04	.05
2-11	.23	.36	.25	.12
2-12	.33	.01	.24	.27
3-4	.6	.17	.19	.35
3-5	.19	.28	.13	.36
3-6	.15	.02	.18	.05
3-7	.35	.09	.04	.4
3-8	.25	.12	.06	.25
3-9	.04	.06	.09	.19
3-10	.09	.28	.18	.07
3-11	.11	.41	.11	.02
3-12	.21	.08	.2	.19
4-5	.03	.06	.01	.16
4-6	.16	.08	.06	.23
4-7	.42	.15	.16	.35
4-8	.12	.17	.12	.47
4-9	.06	.29	.2	.21
4-10	.14	.02	.1	.25
4-11	.08	.01	.08	.38
4-12	.36	.13	.12	.08
5-6	.3	.2	.15	.07
5-7	.02	.11	.07	.17
5-8	.22	.08	.06	.19
5-9	.01	.06	.13	.08
5-10	.26	.05	.24	.05
5-11	.09	.04	.01	.17
5-12	.05	.25	.14	.34
6-7	.23	.26	.21	.18
6-8	.06	.34	.17	.09
6-9	.14	.5	.03	.2
6-10	.01	.21	.08	.36

6-11	.34	.11	.15	.08
6-12	.21	.19	.04	.19
7-8	.04	.06	.06	.32
7-9	.11	.03	.11	.15
7-10	.1	.35	.25	.09
7-11	.29	.37	.18	.31
7-12	.05	.21	.14	.11
8-9	.01	.07	.09	.4
8-10	.22	.19	.21	.07
8-11	.35	.23	.32	.02
8-12	.06	.25	.19	.11
9-10	.28	.16	.15	.29
9-11	.08	.05	.13	.33
9-12	.07	.09	.22	.2
10-11	.02	.1	.1	.15
10-12	.14	.46	.09	.21
11-12	.07	.22	.06	.32

Overall, the performance of the methodology was satisfactory. It built models with an average Adjusted R-Squared value of .798 which is almost .8, had an average Mean Magnitude of Relative Error of 33%. An average value of .8 indicates that all the models built from the data sets imputed using the methodology are considerably good and having an average MMRE of 33% shows the estimates had less bias. Moreover, in every model built there were on an average 64% of cases having relative error less than or equal to 25%. As it identifies "like" cases and clusters them, before choosing a donor, there is a high reliability that missing cases are often imputed with the "most probable" values. Due to the very nature of the method to form homogenous clusters, the missing pattern or the missing mechanism cause no degradation in its performance. In our study though, we ended up with credible data sets and reasonable models were built when 46% of data were missing (DS 5). However, more number of data sets needs to be tested before conforming the performance of the methodology when missingness is present in high quantities (> 40%). The above arguments and statistics suggest that the methodology could be used for different kinds of data sets (such as small, medium, and large) and under different conditions (such as pattern of missing data, % of missing data and under different missing mechanisms).

4 Conclusions

We discussed our hybrid methodology to overcome the limitations in most imputation methods and evaluated its validity based on three different cluster metrics. We detailed on the three metrics and thus showed how they measured the performance of the methodology. Further more, our experimental results showed that we succeeded in decreasing bias [16]. Based on the results we are sure we have made a point about the validity of our methodology's performance. We do not recall such an application

of clustering algorithms for the enhancements of software project data sets to the best of our knowledge.

References

[1] B.W. Boehm, Software Engineering Economics. Prentice Hall, 1981.

[2] T. DeMarco, Controlling Software Projects: Management, Measurement, and Estimates, NY: Prentice-Hall, 1982.

[3] Little, R. J. A. and D. B. Rubin, Statistical Analysis with Missing Data. NY, John Wiley & Sons, 2002.

[4] Roth, P. (1994), "Missing data: A conceptual review for applied psychologists", *Personnel Psych*, *47*, 537-560.

[5] Schafer, J. L.,&Graham, J.W., "Missing data: Our view of the state of the art", *Psych. Methods*, 2002.

[6] Rubin, D. B. (1976) "Inference and missing data", *Biometrika*, *63*, 581-592.

[7] J. Kim and J. Curry, The Treatment of Missing Data in Multivariate Analysis, Social Methods & Research, vol. 6, pp. 215-240, 1977.

[8] Sumanth Yenduri, S.S. Iyengar, Louise A. Perkins, "Improving Prediction Accuracies using Data Imputation Methods", 2005 Proceedings of Intl., Conference on S/W Engg., Research & Practice, June 27-30, Nevada, 2005, USA.

[9] K.E. Emam and A. Birk, "Validating the ISO/IEC 15504 Measure of Software Requirements Analysis Process Capability", *IEEE Trans. Software Eng.*, vol. 26, June 2000.

[10] K. Strike, K.E. Emam, and N. Madhavji, Software Cost Est. with Incomplete Data, IEEE Trans. S/W Eng., 2001.

[11] Myrtveit, I., E. Stensrud and U.H. Olsson, Analyzing data sets with missing data: An empirical evaluation of imputation methods and likelihood-based methods, IEEE Transactions on S/W Engineering 27(11), pp999-1013, 2001.

[12] Song, Q and Shepperd, M, "A Short Note on Using Multiple Imputation Techniques for Very Small Data Sets", April 2003, Technical Report, Bournemouth Univ., UK.

[13] B. Cox and R. Folsom, "An Empirical Investigation of Alternate Item Nonresponse Adjustments", *Proc. Section on Survey Research Methods*, pp. 219-223, 1978.

[14] J. Kaiser, "The Effectiveness of Hot-Deck Proc. in Small Samples", *Ann. Meeting of the Am. Stat. Assoc.*, 1983.

[15] B. Ford, "Missing Data Procedures: A Comparative Study", *Proc. Social Stat. Sec.*, pp. 324-329, 1976.

[16] Sumanth Yenduri, "An Agglomerative Clustering Methodology For Data Imputation", IEEE proceedings of Third Intl., Conference on Information Technology: New Generations, ITNG 2006, April 10-12, 2006, USA.

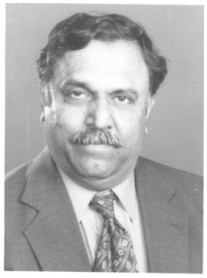
[17] Chipman, H., and Tibshirani, R. "Hybrid Hierarchical Clustering with Applications to Microarray Data", Biostatistics, 2003.

[18] Ying Zhao and George Karypis, "Evaluation of hierarchical clustering algorithms for document datasets", Proceedings of the eleventh international conference on Information and knowledge management (CIKM), Pg 515-524, Mclean, VA, Nov 4-9, 2002.

Biography

▲ Name: Sumanth Yenduri
Address: 730 East Beach Blvd, Computer Science Dept., USM, Long Beach, MS 39503
Education & Work experience: PhD in Computer Science, Louisiana State University & Assistant Professor, University of Southern Mississippi
Tel: +1-228-5395023
E-mail: Sumanth.Yenduri@usm.edu

Other information: Sumanth Yenduri was born in Rajahmundry, India. He received his Bachelor of Technology in Computer Science and Systems Engineering from Andhra University in 1999. Later, he joined the Department of Computer Science at Louisiana State University and received his Master's degree in December, 2002. Further he received his PhD in Computer Science in August 2005. He is currently working as an Assistant Professor in Computer Science at the University of Southern Mississippi, USA. His research interests include software process development, data imputation methods, software metrics, software tools, effort/cost/time estimation models, software engineering data analysis and programming languages.



▲ Name: S.S. Iyengar
Address: 298 Coates Hall, Tower Drive, Computer Science Dept., Baton Rouge, LA 70802
Education & Work experience: PhD in Mech. Engg, Mississippi State University & Professor & Chairman, Louisiana State University
Tel: +1-225-5781252
E-mail: iyengar@bit.csc.lsu.edu

Other information: Dr. S. S. Iyengar is the Chairman and Roy

Paul Daniels Chaired Professor of Computer Science at Louisiana State University and is also Satish Dhawan Chaired Professor at Indian Institute of Science. He has been involved with research in high-performance algorithms, data structures, sensor fusion, data mining, and intelligent systems since receiving his Ph.D. degree (in 1974 at Mississippi State University) and his M.S. from the Indian Institute of Science (1970). He has directed over 30 Ph.D. candidates, many of whom are faculty at major universities worldwide or scientists or engineers at national labs/industry around the world. His publications include 13 books (authored or coauthored, edited; Prentice-Hall, CRC Press, IEEE Computer Society Press, John Wiley & Sons, etc.) and over 300 research papers in refereed journals and conference in areas of high-performance parallel and distributed algorithms and data structures for image processing and pattern recognition, and distributed data mining algorithms for biological databases. His books have been used by researchers at Purdue, University of Southern California, University of New Mexico, etc. at various times. He was a visiting professor at the Jet Propulsion Laboratory-Cal. Tech, Oak Ridge National Laboratory, the Indian Institute of Science, and at the University of Paris and other places. He has been on the prestigious National Institute of Health-NLM Review Committee for 4 years.

Dr. Iyengar was the winner of the IEEE Computer Society Technical Achievement Award for Outstanding Contributions to Data Structures and Algorithms in Image Processing and Sensor Fusion Problems. This is the most prestigious research award from IEEE Computer Society. Dr. Iyengar was awarded the LSU Distinguished Faculty Award for Excellence in Research, the Hub Cotton Award for Faculty Excellence, and the LSU Tiger Athletic Foundation Teaching Award in 1996. He has been a consultant to several industrial and government organizations (JPL, NASA etc.). In 1999, Professor Iyengar won the most prestigious research award titled Distinguished Research Award and a university medal for his research contributions in optimal algorithms for sensor fusion/image processing. He is also a Fellow of ACM, a Fellow of the IEEE, a Fellow of AAAS, IEEE Distinguished Visitor, etc. He received the Prestigious Distinguished Alumnus Award from Indian Institute of Science, Bangalore in 2003. Also, Elected Member of European Academy of Sciences (2002). He is a member of the New York Academy of Sciences. He has been the Program Chairman for many national/international conferences. He has given over 60 plenary talks and keynote lectures at numerous national and international conferences.