Improving Data Quality in Applications of Dynamic Forms

[1]Kotakonda. Madhu Babu, [2]G. Subba Lakshmi, [3]Ch. Raja Jacob
[123]Dept. of CSE, Nova College of Engineering and Technology, Vegavaram, AP, India

Abstract
Quality of data is a major problem in high dimensional and modern databases. Especially in data entry forms present the initial and arguable an efficient opportunity for identifying and mitigating the errors, but there are many methods and researches which are conducted in order to improve the data quality during the entry of data into the forms. In this paper we propose an end to end system for designing of forms, entry and providing the quality assurance to the data during the submission of forms. This system learns a probabilistic model over the question of the form. At every step of the form entry this model is implemented to provide the better quality assurance compare with the earlier methodologies. Before the entry, it induces and identifies a form layout that captures the important data values of a form. Before entry, it induces a form layout that captures the most important data values of a form instance as quickly as possible and reduces the complexity of error-prone questions. During entry, it dynamically adapts the form to the values being entered by providing real-time interface feedback, reasking questions with dubious responses, and simplifying questions by reformulating them. After entry, it revisits question responses that it deems likely to have been entered incorrectly by reasking the question or a reformulation thereof. We evaluate these components of USHER using two real-world data sets. Our results demonstrate that USHER can improve data quality considerably at a reduced cost when compared to current practice

Keywords
Data quality, Data Entry, Form Design, Adaptive Form

I. Introduction

Organizations and individuals routinely make important decisions based on inaccurate data stored in supposedly authoritative databases. Data errors in some domains, such as medicine, may have particularly severe consequences. These errors can arise at a variety of points in the life cycle of data, from data entry, through storage, integration, and cleaning, all the way to analysis and decision making [1]. While each step presents an opportunity to address data quality, entry time offers the earliest opportunity to catch and correct errors. The database community has focused on data cleaning once data have been collected into a database, and has paid relatively little attention to data quality at collection time [1-2]. Current best practices for quality during data entry come from the field of survey methodology, which offers principles that include manual question orderings and input constraints, and double entry of paper forms [3]. Although this has long been the de facto quality assurance standard in data collection and transformation, we believe this area merits reconsideration. For both paper forms and direct electronic entry, we posit that a data-driven and more computationally sophisticated approach can significantly outperform these decades-old static methods in both accuracy and efficiency of data entry. The problem of data quality is magnified in low-resource data collection settings. Recently, the World Health Organization likened the lack of quality health information in developing regions to a "gathering storm," saying, "[to] make people count, we first need to be able to count people" [4]. Indeed, many health organizations, particularly those operating with limited resources in developing regions, struggle with collecting high-quality data. Why is data collection so challenging? First, many organizations lack expertise in paper and electronic form design: designers approach question and answer choice selection with a defensive, catchall mind-set, adding answer choices and questions that may not be necessary; furthermore, they engage in ad hoc mapping of required data fields to data-entry widgets by intuition [5-6], often ignoring or specifying ill-fitting constraints. Second, double entry is too costly. In some cases this means it is simply not performed, resulting in poor data quality. In other cases, particularly when double entry is mandated by third parties, it results in delays and other unintended negative consequences. We observed this scenario in an HIV/AIDS program in Tanzania, where time-consuming double entry was imposed upon a busy local clinic. The effort required to do the double entry meant that the transcription was postponed for months and handled in batch. Although the data eventually percolated up to national and international agencies, in the interim the local clinic was operating as usual via paper forms, unable to benefit from an electronic view of the data latent in their organization. Finally, many organizations in developing regions are beginning to use mobile devices like smartphones for data collection; for instance, community health workers are doing direct digital data entry in remote locations. Electronic data-entry devices offer different affordances than those of paper, displacing the role of traditional form design and double entry [5]. We often found that there were no data quality checks at all in naı¨vely implemented mobile interfaces, compounding the fact that mobile data-entry quality is 10 times worse than dictation to a human operator [7].

To address this spectrum of data quality challenges, we have developed USHER, an end-to-end system that can improve data quality and efficiency at the point of entry by learning probabilistic models from existing data, which stochastically relate the questions of a data-entry form. These models form a principled foundation on which we develop information-theoretic algorithms for form design, dynamic form adaptation during entry, and answer verification:

1. Since form layout and question selection is often ad hoc, USHER optimizes question ordering according to a probabilistic objective function that aims to maximize the information content of form answers as early as possible—we call this the greedy informa-tion gain principle. Applied before entry, the model generates a static but entropy-optimal ordering, which focus on important questions first; during entry, it can be used to dynamically pick the next best question, based on answers so far—appropriate in scenarios where question ordering can be flexible between instances.
2. Applying its probabilistic model during data entry, USHER can evaluate the conditional distribution of answers to a form question, and make it easier for likely answers to be entered—we call this the appropriate entry friction principle. For difficult-to-answer questions, such as those with many extra-neous choices, USHER can opportunistically reformu-late them to be easier and more congruous with the available information. In this way, USHER effectively allows for a principled, controlled trade-off between data quality and form

filling effort and time.

3. Finally, the stochastic model is consulted to predict which responses may be erroneous, so as to reask those questions in order to verify their correctness—we call this the contextualized error likelihood princi-ple. We consider reasking questions both during the data-entry process (integrated reasking) and after data entry has been finished (post hoc reasking). In both cases, intelligent question reasking approxi-mates the benefits of double entry at a fraction of the cost.

In addition, we may extend USHER's appropriate entry friction approach to provide a framework for reasoning about feedback mechanisms for the data-entry user inter-face. During data entry, using the likelihood of unanswered fields given entered answers, and following the intuition that multivariate outliers are values warranting reexamina-tion by the data-entry worker, USHER can guide the user with much more specific and context-aware feedback. In Section 9, we offer initial thoughts on design patterns for USHER-inspired dynamic data-entry interfaces.

The contributions of this paper are fourfold:

1. We describe the design of USHER's core: probabilistic models for arbitrary data-entry forms
2. We describe USHER's application of these models to provide guidance along each step of the data-entry life cycle: reordering questions for greedy information gain, reformulating answers for appropriate entry friction, and reasking questions according to contextualized error likelihood.
3. We present experiments showing that USHER has the potential to improve data quality at reduced cost. We study two representative data sets: direct electronic entry of survey results about political opinion and transcription of paper-based patient intake forms from an HIV/AIDS clinic in Tanzania.
4. Extending our ideas on form dynamics, we propose new user-interface principles for providing contex-tualized, intuitive feedback based on the likelihood of data as they are entered. This provides a foundation for incorporating data cleaning mechan-isms directly in the entry process.

II. Data Entry Learning Model

The core of the USHER system is its probabilistic model of the data, represented as a Bayesian network over form questions. This network captures relationships between a form's question elements in a stochastic manner. In particular, given input values for some subset of the questions of a particular form instance, the model can infer probability distributions over values of that instance's remaining unanswered questions. In this section, we show how standard machine learning techniques can be used to induce this model from previous form entries.

We will use F ¼ fF$_1$; . . . ; F$_n$g to denote a set of random variables representing the values of n questions comprising a data-entry form. We assume that each question response takes on a finite set of discrete values; continuous values are discretized by dividing the data range into intervals and assigning each interval one value.2 To learn the probabilistic model, we assume access to prior entries for the same form.

USHER first builds a Bayesian network over the form questions, which will allow it to compute probability distributions over arbitrary subsets G □ F of form question random variables, given already entered question responses G^0 ¼ g^0 for that instance, i.e., P ðG j G ^{0} ¼ g^{0}Þ. Constructing this network requires two steps: first, the induction of the graph structure of the network, which encodes the condi-tional independencies between the question random vari-ables F; and second, the estimation of the resulting network's parameters.

The naı¨ve approach to structure selection would be to assume complete dependence of each question on every other question. However, this would blow up the number of free parameters in our model, leading to both poor generalization performance of our predictions and prohibi-tively slow model queries. Instead, we learn the structure using the prior form submissions in the database. USHER searches through the space of possible structures using simulated annealing, and chooses the best structure accord-ing to the Bayesian Dirichlet Equivalence criterion [23]. This criterion optimizes for a trade-off between model expres-siveness (using a richer dependency structure) and model parsimony (using a smaller number of parameters), thus identifying only the prominent, recurring probabilistic dependencies. Figs. 1 and 2 show automatically learned structures for two data domains.3

In certain domains, form designers may already have strong commonsense notions of questions that should or should not depend on each other (e.g., education level and income are related, whereas gender and race are indepen-dent). As a postprocessing step, the form designer can manually tune the resulting model to incorporate such intuitions. In fact, the entire structure could be manually constructed in domains where an expert has comprehensive prior knowledge of the questions' interdependencies. However, a casual form designer is unlikely to consider the complete space of question combinations when identi-fying correlations. In most settings, we believe an automatic approach to learning multivariate correlations would yield more effective inference.

III. Reordering Questions during Data Entry

In electronic form settings, we can take our ordering notion a step further and dynamically reorder questions in a form as an instance is being entered. This approach can be appropriate for scenarios when data-entry workers input one or several values at a time, such as on a mobile device. We can apply the same greedy information gain criterion as in Algorithm 1, but update the calculations with the previous responses in the same form instance. Assuming questions G F1; . . . ; F‘ghave already been filled in with values g ¼ ff1; . . . ; f‘g, the next question is selected by maximizing:

$$H(F_i \mid \mathbf{G} = \mathbf{g}) = -\sum_{f_i} P(F_i = f_i \mid \mathbf{G} = \mathbf{g}) \log P(F_i = f_i \mid \mathbf{G} = \mathbf{g}).$$

Notice that this objective is the same as (4), except that it uses the actual responses entered for previous questions, rather than taking a weighted average over all possible values. Constraints specified by the form designer, such as topical grouping, can also be respected in the dynamic framework by restricting the selection of next questions at every step.

In general, dynamic reordering can be particularly useful in scenarios where the input of one value determines the value of another. For example, in a form with questions for gender and pregnant, a response of male for the former dictates the value and potential information gain of the latter. However, dynamic reordering may be confusing to data-entry workers who routinely enter information into the same form, and have come to expect a specific question order. Determining the trade-off between these opposing concerns is a human factors issue that depends on both the application domain and the user interface employed.

IV. Question Reasking

The next application of USHER's probabilistic model is for the purpose of identifying errors made during entry. Because this determination is made during and immedi-ately after form submission, USHER can choose to reask questions during the same entry session. By focusing the reasking effort only on questions that were likely to be misentered, USHER is likely to catch mistakes at a small incremental cost to the data-entry worker. Our approach is a data-driven alternative to the expensive practice of double entry. Rather than reasking every question, we focus reasking effort only on question responses that are unlikely with respect to the other form responses.

Before exploring how USHER performs reasking, we explain how it determines whether a question response is erroneous. USHER estimates contextualized error likelihood for each question response, i.e., a probability of error that is conditioned on every other previously entered field response. The intuition behind error detection is straight-forward: questions whose responses are "unexpected" with respect to the rest of the known input responses are more likely to be incorrect. These error likelihoods are measured both during and after the entry of a single form instance.

V. Error Model

To formally model the notion of error, we extend our Bayesian network from Section 4 to a more sophisticated representation that ties together intended and actual question responses. We call the Bayesian network augmen-ted with these additional random variables the error model. Specifically, we posit a network where each question is augmented with additional nodes to capture a probabilistic view of entry error. For question i, we have the following set of random and observed variables:

. Fi: the correct value for the question, which is unknown to the system, and thus a hidden variable.

. Di: the question response provided by the data-entry worker, an observed variable.

. thetai: the observed variable representing the para-meters of the probability distribution of mistakes across possible answers, which is fixed per ques-tion.5 We call the distribution with parameters □i the error distribution. For the current version of our model, □i is set to yield a uniform distribution.

. Ri: a binary hidden variable specifying whether an error was made in this question. When Ri ¼ 0 (i.e., when no error is made), then Fi takes the same value as Di.

Additionally, we introduce a hidden variable □, shared across all questions, specifying how likely errors are to occur for a typical question of that form instance.

The error model. Observed variable Di represents the actual input provided by the data-entry worker for the ith question, while hidden variable Fi is the true value of that question. The rectangular plate around the center variables denotes that those variables are repeated for each of the ' form questions with responses that have already been input. The F variables are connected by edges z 2 Z, representing the relationships discovered in the structure learning process; this is the same structure used for the question ordering component. Variable □i represents the "error" distribution, which in our current model is uniform over all possible values. Variable Ri is a hidden binary indicator variable specifying whether the entered data were erroneous; its probability □i is drawn from a Beta prior with fixed hyperparameters □ and □. Shaded nodes denote observed variables, and clear nodes denote hidden variables.

Intuitively, □ plays the role of a prior error rate, and is modeled as a hidden variable so that its value can be learned directly from the data.

Note that the relationships between field values dis-covered during structure learning are still part of the graph, so that the error predictions are contextualized in the answers of other related questions.

Within an individual question, the relationships between the newly introduced variables are shown in Fig. 5. The diagram follows standard plate diagram notation [26]. In brief, the rectangle is a plate containing a group of variables specific to a single question i. This rectangle is replicated for each of ' form questions. The F variables in each question group are connected by edges z 2 Z, representing the relationships discovered in the structure learning process; this is the same structure used for the question ordering component. The remaining edges represent direct probabil-istic relationships between the variables that are described in greater detail below. Shaded nodes denote observed variables, and clear nodes denote hidden variables.

Node Ri 2 f0; 1g is a hidden indicator variable specify-ing whether an error will happen at this question. Our model posits that a data-entry worker implicitly flips a coin for Ri when entering a response for question i, with probability of one equal to □. Formally, this means Ri is drawn from a Bernoulli distribution with parameter □:

$$R_i \mid \lambda \sim \text{Bernoulli}(\lambda).$$

The value of Ri affects how Fi and Di are related,

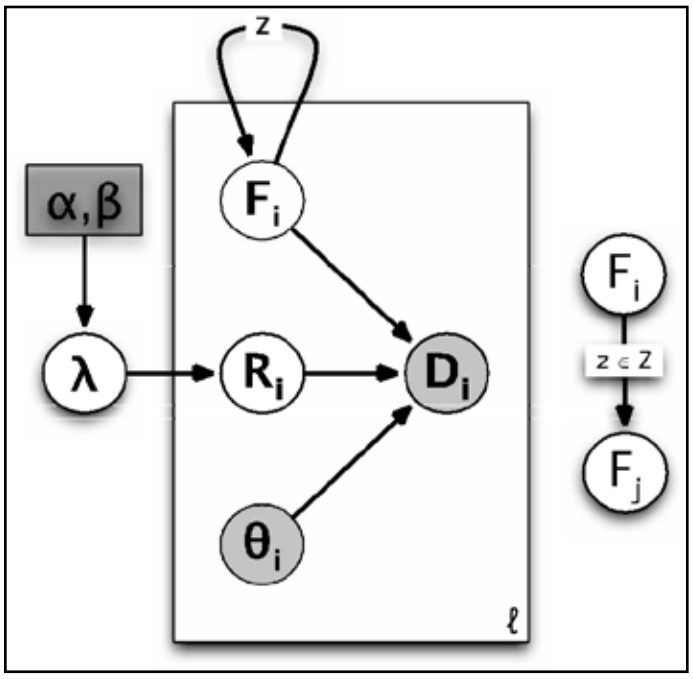

Fig. 1:

which is described in detail later in this section.

We also allow the model to learn the prior probability for the □ directly from the data. This value represents the probability of making a mistake on any arbitrary question. Note that □ is shared across all form questions. Learning a value for □ rather than fixing it allows the model to produce an overall probability of error for an entire form instance as well as for individual questions. The prior distribution for □ is a Beta distribution, which is a continuous distribution over the real numbers from zero to one:

$$\lambda \sim \mathrm{Beta}(\alpha, \beta).$$

The Beta distribution takes two hyperparameters alpha and Beta, which we set to fixed constants (1, 19). The use of a Beta prior distribution for a Bernoulli random variable is standard practice in Bayesian modeling due to mathema-tical convenience and the interpretability of the hyperpara-meters as effective counts [27]. We now turn to true question value Fi and observed input Di. First, P (Fi | . . .) is still defined as in Section 4, maintaining as before the multivariate relationships be-tween questions. Second, the user question response Di is modeled as being drawn from either the true answer Fi or the error distribution theta, depending on whether a mistake is made according to Ri:

$$D_i \mid F_i, \theta_i, R_i \sim \begin{cases} \mathrm{PointMass}(F_i), & \text{if } R_i = 0, \\ \mathrm{Discrete}(\theta_i), & \text{otherwise.} \end{cases} \quad \text{ð9Þ}$$

If Ri ¼ 0, no error occurs and the data-entry worker inputs the correct value for Di, and thus Fi ¼ Di. Probabilistically, this means Di's probability is concentrated around Fi (i.e., a point mass at Fi). However, if Ri ¼ 1, then the data-entry worker makes a mistake, and instead chooses a response for the question from the error distribution. Again, this error distribution is a discrete distribution over possible question responses parameterized by the fixed parameters □i, which we set to be the uniform distribution in our current model.6

VI. Error Model Inference

The ultimate variable of interest in the error model is Ri: we wish to induce the probability of making an error for each previously answered question, given the actual question responses that are currently available:

$$P(R_i \mid \mathbf{D} = \mathbf{d}),$$

where D ¼ fF$_1$; . . . ; F'g are the fields that currently have responses, the values of which are d ¼ ff$_1$; . . . ; f'g, respec-tively. This probability represents a contextualized error likelihood due to its dependence on other field values through the Bayesian network.

Again, we can use standard Bayesian inference proce-dures to compute this probability. These procedures are black box algorithms whose technical descriptions are beyond the scope of this paper. We refer the reader to standard graphical model texts for an in-depth review of different techniques [25, 28]. In our implementation, we use the Infer.NET toolkit [22] with the Expectation Propagation algorithm [29] for this estimation.

In the sections above, we described how USHER uses statistical information traditionally associated with offline data cleaning to improve interactive data entry via question ordering and reasking. This raises questions about the human-computer interactions inherent in electronic form filling, which are typically device- and application-dependent. In one application, we are interested in how data quality interactions play out on mobile devices in developing countries, as in the Tanzanian patient forms we examined above. Similar questions arise in traditional online forms like web surveys. In this section, we outline some design opportunities that arise from the probabilistic power of the models and algorithms in USHER. We leave the investigation of specific interface designs and their evaluation in various contexts to future work. While an interactive USHER-based interface is presenting questions (either one-by-one or in groups), it can infer a probability for each possible answer to the next question; those probabilities are contextualized (conditioned) by previous responses. The resulting quantitative probabilities can be exposed to users in different manners and at different times. We present some of these design options in the following:

A. Time of Exposure

Pre- and post entryBefore entry, Usher's probabilistic model can be used to improve data-entry speed by adjusting the friction of entering different answers: likely results can be made easy or attractive to enter, while unlikely results can be made to require more work. One example of this is the previously described reformulation technique. Additional examples of data-driven variance in lists, and direct decoration (e.g., coloring or font size) of each choice in accordance with its probability. A downside of beforehand exposure of answer probabilities is the potential to bias answers. Alternatively, probabilities may be exposed in the interface only after the user selects an answer. This becomes a form of assessment—for example, by flagging unlikely choices as potential outliers. This can be seen as a soft, probabilistic version of the constraint violation visualizations commonly found in web forms (e.g., the red star that often shows up next to forbidden or missing entries). Post hoc assessment arguably has less of a biasing effect than friction. This is both because users choose initial answers without knowledge of the model's predictions, and because users may be less likely to modify previous answers than change their minds before entry.

B. Explicitness of exposure

Feedback mechanisms in adaptive interfaces vary in terms of how explicitly they intervene in the user's task. Adaptations can be considered elective versus mandatory. For instance, a drop-down menu with items sorted based on likelihood is mandatory with a high level of friction; whereas, a "split" drop-down menu, as mentioned above, is elective—the user can choose to ignore the popular choices. Another important consideration is the cognitive complexity of the feedback. For instance, when encoding expected values into a set of radio buttons, we can directly show the numeric probability of each choice, forcing a user to interpret these discrete probabilities. Alternatively, we can scale the opacity of answer labels—giving the user an indication of relative salience, without the need friction include type-ahead mechanisms in textfields, "popular choice" items repeated at the top of drop-down

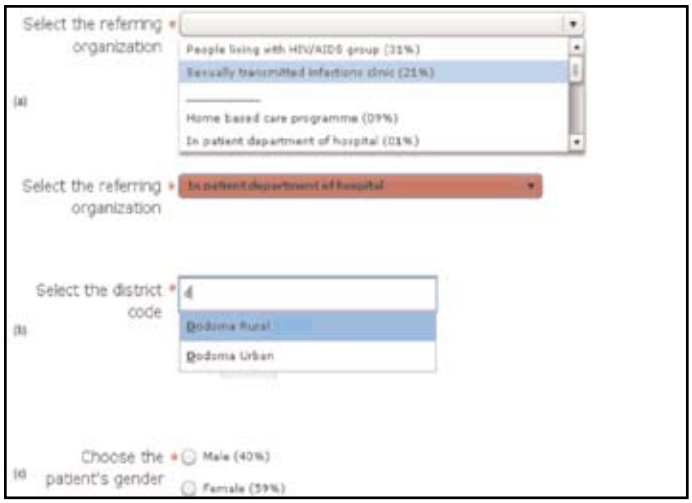

Fig. 2:

for interpretation. Even more subtly, we can dynamically adjust the size of answer labels' clickable regions—similar to the adjustments made by the iPhone's soft keyboard in response to the likelihood of various letters.

C. Contextualization of Interface

USHER uses conditional probabilities to assess the likelihood of subsequent answers. However, this is not necessarily intuitive to a user. For example, consider a question asking for favorite beverage, where the most likely answers shown are milk and apple juice. This might be surprising in the abstract, but would be less so in a case where a previous question had identified the age of the person in question to be under 5 years old. The way that the interface communicates the context of the current probabilities is an interesting design consideration. For example, "type-ahead" text interfaces have this flavor, showing the likely suffix of a word contextualized by the previously entered prefix. More generally, USHER makes it possible to show a history of already entered answers that correlate highly with the value at hand.

References

[1] A. Ali, C. Meek,"Predictive Models of Form Filling", Technical Report MSR-TR-2009-1, Microsoft Research, Jan. 2009.

[2] L.A. Hermens, J.C. Schlimmer,"A Machine-Learning Apprenticefor the Completion of Repetitive Forms", IEEE Expert:Intelligent Systems and Their Applications, Vol. 9, No. 1, pp. 28-33, Feb. 1994.

[3] D. Lee, C. Tsatsoulis,"Intelligent Data Entry Assistant forXML Using Ensemble Learning", Proc. ACM 10th Int'l Conf.Intelligent User Interfaces (IUI), 2005.

[4] J.C. Schlimmer, P.C. Wells,"Quantitative Results Comparing Three Intelligent Interfaces for Information Capture", J. Artificial Intelligence Research, Vol. 5, pp. 329-349, 1996.

[5] S.S.J.R. Warren, A. Davidovic, P. Bolton,"Mediface: Anticipative Data Entry Interface for General Practitioners", Proc. Australasian Conf. Computer Human Interaction (OzCHI), 1998.

[6] J. Warren, P. Bolton,"Intelligent Split Menus for Data Entry: A Simulation Study in General Practice Medicine", Proc. Am. Medical Informatics Assoc. (AMIA) Ann. Symp., 1999.

[7] Y. Yu, J.A. Stamberger, A. Manoharan, A. Paepcke, "Ecopod: A Mobile Tool for Community Based Biodiversity Collection Building", Proc. Sixth ACM/IEEE CS Joint Conf. Digital Libraries (JCDL), 2006.

[8] S. Day, P. Fayers, D. Harvey,"Double Data Entry: What Value, What Price?", Controlled Clinical Trials, Vol. 19, No. 1, pp. 15-24, 1998.

[9] D.W. King, R. Lashley,"A Quantifiable Alternative to Double Data Entry", Controlled Clinical Trials, Vol. 21, No. 2, pp. 94-102, 2000.

[10] K. Kleinman,"Adaptive Double Data Entry: A Probabilistic Tool for Choosing Which Forms to Reenter", Controlled Clinical Trials, Vol. 22, No. 1, pp. 2-12, 2001.

Mr.Kotakonda.Madhu Babu is a student of NOVA college of engineering and technology, Vegavaram, presently he is pursuing M.Tech (C.S.E) From NOVA college and his arees of interest are Data Mining and Network Security.

Mrs. G.Subba Lakshmi an efficient teacher, received M.Tech(C.S.E) from JNTU Kakinada is working as an Assitant Professor in Department of C.S.E of NOVA College of Engineering and Technology Vegavaram. Having 4+ years of teaching experience her area of interest includes Data Mining, Network Security.

Mr. Ch.Raja jacob, well known author and excellent teacher received M.Tech (C.S.E) and he is working as H.O.D for the department of M.Tech., Computer Science & Engineering NOVA College of Engineering and Technology vegavarm. He has vast teaching experience in various engineering colleges. He has couple of publications both National & International Conferences / Journals. His intersted research on Data warehousing and data mining, information security. Data communications and networks.