ISSN:0975-9646

Text database cleaning by filling the Missing values using Object Oriented Intelligent Multi-Agent System Data Cleaning Architecture

Dr. G. Arumugam[1],T. Joshva Devadas[2]

*[1]Department of Computer Science, Madurai Kamaraj University, Madurai 625021**
[2]Department of Computer Science The American College, Madurai 625002

Abstract— Agents are software programs that perform tasks on behalf of others and they are used to clean the text database with their characteristics. Agents are task oriented with the ability to learn by themselves and they react to the situation. Learning characteristics of an agent is done by verifying its previous experience from its knowledgebase. An agent concept is a complementary approach to the Object Oriented paradigm with respect to the design and implementation of the autonomous entities driven by beliefs, goals and plans. Text database cleaning process detects and cleans the wrong data or duplicates data or missing data by identifying the outliers. Cleaning of Text Databases focuses on incomplete data cleaning. Incomplete data cleaning is performed using the attribute missing rate. Agents incorporated in the architectural design of a Text database cleaning process combines both the features of Multi-Agent System (MAS) Framework and MAS with learning (MAS-L) Framework. MAS framework reduces the development time and the complexity of implementing the software agents. MAS-L framework incorporates the intelligence and learning properties of agents present in the system. MAS-L Framework makes use of the Decision Tree learning and an evaluation function to decide the next best decision that applies to the machine learning technique. This paper proposes the design for Multi-Agent based Data Cleaning Architecture that incorporates the structural design of agents into object model. The Design of an architectural model for Multi-Agent based Data Cleaning inherits the features of the Multi-Agent System (MAS) and uses the MAS-L framework to design the intelligence and learning characteristics.

***Keywords*— Text database, Incomplete data, Agents, MAS, MAS-L, Architecture, Data Cleaning**

1. INTRODUCTION

Data cleaning and knowledge discovery are the emerging research areas that are growing exponentially [5]. Popularity of the internet and traditional information sources, like relational databases are the factors that influence this growth of information in online. The availability of these resources not only increases the complexity of data leads to the question of considering about the quality of such data.

In collecting the data from the real-world application, we may receive data with important feature missing such as attribute missing values, duplicates, etc., [17]. Such data collected from various sources may not be useful to obtain complete information and it is referred as incomplete data. The data records with incomplete information need to undergo data cleaning process. Text database cleaning process detects and cleans the wrong data or missing data by identifying the outliers.

Incomplete data is considered as one of the most common means of error occurs to the database. Incomplete data are caused due to non availability of certain data or the difference in detailed schema from multiple databases. Appropriate procedures are used to handle incomplete information of database [18]. Since, there are many methods available for handling incomplete information, identifying a method that is more suitable for the given application is considered as the prime task of this cleaning process.

Data Cleaning Multi-Agent (DCMA) Architecture uses knowledge processing elements (Agents) to identify the incomplete information of the database and uses necessary procedure / methods to fill the missing data [32]. Further, agents present in the DCMA architecture make use of the MAS and MAS-L frameworks to establish communication and intelligence [6]. Combined features of those two object oriented frameworks and the proposed DCMA Architecture make the system to react using their intelligent behavior. This system is referred as Intelligent Multi-Agent System Data Cleaning (IMASDC) Architecture [15]. Text database with incomplete data records is cleaned using this IMASDC Architecture.

In this paper Section 2 describes about the object oriented frameworks MAS and MAS-L. Section 3 discusses the Text Database cleaning in terms of Incomplete Data cleaning. Section 4 describes the functionalities of Data Cleaning Multi-Agent Architecture and the text database cleaning by

identifying the missing values present in the text database is described using Multi-Agent Data Cleaning Architecture. Section 5 introduces Agent learning for Incomplete data in Text database cleaning and presents Decision Tree based learning algorithm for it. Also this Section illustrates the significance of Decision tree construction process with sample training example. Section 6 explains Text Database Cleaning by identifying the missing values and filling them using the IMASDC Architecture and the implementation process with a sample text database. Section 7 analyses normal Text Database Cleaning of Incomplete data method and the method based on IMASDC Architecture.

2. OBJECT ORIENTED FRAMEWORKS

2.1MAS Framework

Objective of MAS framework is to reduce the development time and the complexity of implementing software agents [31]. The design of this framework is based on *Gaia methodology*, which is used to build models in the analysis and the design phase [34]. This framework is composed of one abstract class Agent, two final classes namely ProcessMessageThread and AgentCommunicationLayer and four interfaces [10]. The interfaces are AgentMessage, AgentBlackBoard information, InteractionProtocol and AgentInterface [11].

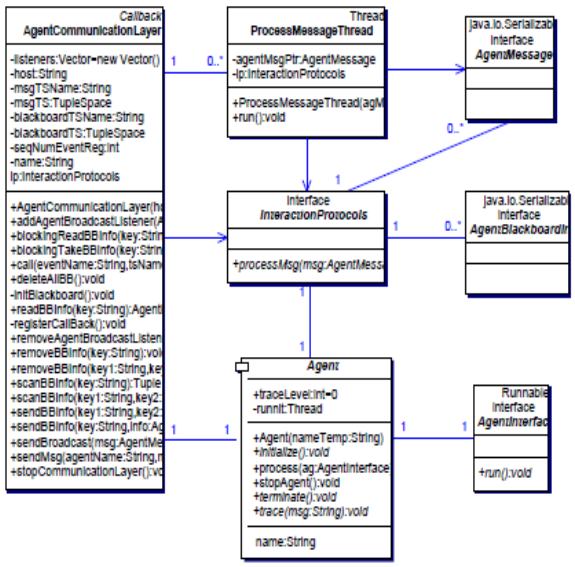


Figure 1 – MAS Framework

Agent class is responsible for providing start up code, ending code, and display of messages [9]. AgentInterface, a sub class inherited from Agent, implements the run method and is responsible for initiating agent's private actions. InteractionProtocol interface defines the interaction among the agents. ProcessMessageThread creates a new control for every incoming message and AgentMessage interface takes care of the message format. AgentBlackboard information specifies the black board message format. The class AgentCommunicationLayer implements the entire communication needed for agents to interact with the distributed system.

2.2 MAS-L Framework

The MAS-L Framework is combined with the MAS framework to implement the learning and intelligence characteristics of software agents [23]. This Framework incorporates two machine learning approaches namely Decision Tree based learning and an Evaluation function. The decision tree approach of software agent learning is based on the current state of environment. The nodes of the tree are intermediate decision points and the leaf nodes are the final states where the searching ends successfully. The search algorithm indicates the best move that leads to successful final state. The Evaluation function is used to determine the next possible move to perform action.

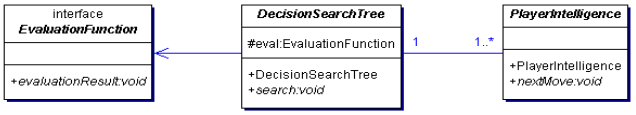


Fig 2. MAS-L Framework

The MAS-L Framework has two abstract classes namely, DecisionSearchTree (DST) and PlayerIntelligence and an interface Evaluation Function [24]. The first class to be extended is Evaluation function that evaluates the intermediate node of the decision tree. This class emphasises well defined data structure along with the properties of the Decision trees to choose a machine learning technique to implement this function. The next class to be extended is the DecisionSearchTree (DST) that implements the data structure of Decision Tree along with the searching. To reduce the searching time in the DST, pruning techniques are incorporated using the maximum depth search algorithm. This class shall instantiate the class that implements evaluation function to perform the evaluation of intermediate nodes. Finally the PlayerIntelligence class acts as an interface to the software agents. This class shall instantiate the class that extends DecisionSearchTree to decide the next best decision.

3. Problem Definition

Cleaning of Text Databases focuses on Incomplete data cleaning. Incomplete data cleaning is performed using the attribute missing rate [7]. The description of the Incomplete data cleaning approaches is explained as follows.

3.1 Incomplete Data Cleaning

While handling large amount databases we may obtain some database with important information missing. Missing values present in the database makes the database an inconsistent one. To overcome the inconsistent problem the database should be cleaned to be used in the real-world application [14]. Main objective of this problem is to identify the incompleteness and then detect the usability of the attribute to determine whether to fill the database or to remove the entire record / tuple.

3.1.1 Principle of Incomplete Data Cleaning

The principle of incomplete data cleaning is described by the following procedure. The cleaning process receives text database as input and produces the output as cleaned text database [16]. Given text database is verified to check the existence of incomplete data by executing appropriate query. Incompleteness is determined by verifying the missing values of the attribute from the resultant query [32]. Next usability of the record is identified by computing the Attribute Missing in Record (AMR)[36]. If the missing rate is low then filling of missing values is done by using manual method, constant substitute method, average substitute method, regular substitute method, or estimated attribute method. If the missing rate is high then there is no use in filling the database tuple and hence it is deleted / removed from the database.

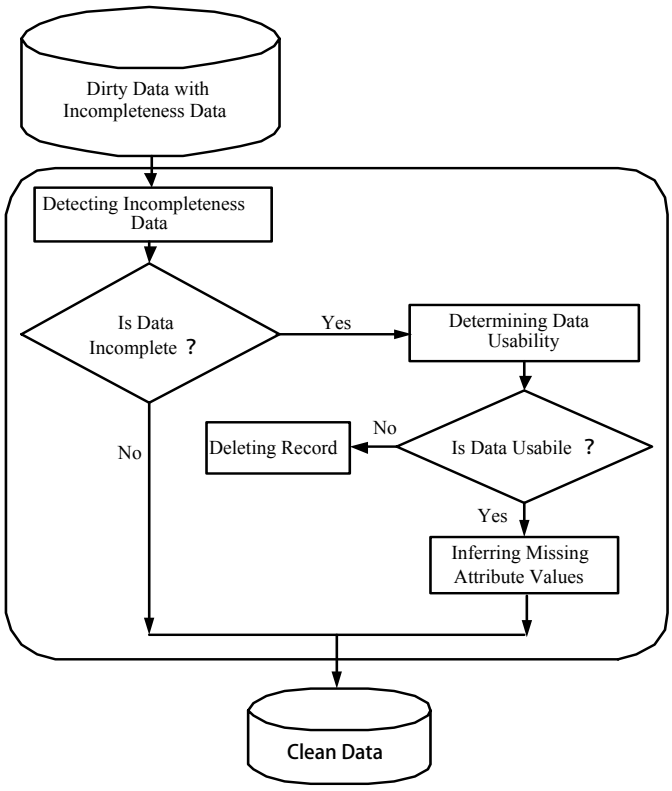


Fig.3 The Principle of Incompleteness Data Cleaning

3.1.2 Phases of Incomplete Data Cleaning

The first phase of the incomplete data cleaning is to detect the incomplete data from the given text database file. Existence of incomplete data present in the given text database are detected using programming constructs or by using query language. The subsequent phase in the incomplete data cleaning is to detect the data usability. The usability determines whether the record should be saved or removed and it is done by computing the degree of incompleteness. In the process of computing the degree of incompleteness the percentage of lost values or missing values is computed first in all the records and then other factors are considered. The existence of the key information in the residual values of the attributes is verified and is used to decide whether to accept or reject. If the attributes of one record has default values (0 for integer and NULL for character) then those default values are considered as lost values. The evaluation of data incompleteness is as follows:

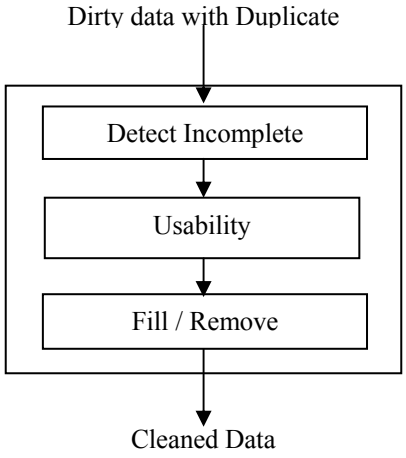


Fig 4. Phases of Incomplete Data Cleaning

The incomplete data cleaning process is to determine the usability of the record. Let $R = \{a_1, a_2, \ldots\ldots a_n\}$ denote the n attributes of record R and is denoted by $a_1, a_2, \ldots, a_n$ and let m denote the number of *missing attribute values* in record R (including the fields whose values are default values). Attribute missing in Record (AMR) denotes the percentage of missing attribute values in record R, ℓ denotes the threshold of the percentage of missing attribute in record R. The Attribute missing in Record is defined as $AMR = \frac{m}{n}$ and the threshold $\ell \in [0,1]$. If $AMR > \ell$ then the record should be retained otherwise it should be discarded. The value ℓ used in the incomplete data cleaning process is

determined by analyzing the source data by an expert and is stored in the system for further reference. Also, the default values are well defined and are used to calculate the value of m.

In the incomplete data cleaning process, apart from considering the degree of incompleteness as a major factor the existence of key attribute is also considered. During the analysis experts determine the key attribute of the source data. The records should also be saved even when $AMR > \ell$, given the key attributes exist in the incomplete data.

Data filling or removing in a record is the final phase of the incomplete data cleaning process and it needs to make use of a specific technique to fill the missing values. This process is described in the next section.

3.1.3 Techniques for Processing Missing Values

There are various methods associated with the missing value processing. Incomplete data cleaning technique is applied after detecting the usability of the record and it chooses manual method or constant substitute method or average substitute method or regular substitute method or/and estimated attribute method. When the amount of data taken for detecting incompleteness is not large *Manual method* is more suitable. *Constant Substitute method* fills all the missing values with the same numeric or string constant as "Unknown" or "Miss value" or "Zero" or "101". This method is very simple to substitute the missing values but may result in wrong analysis because all the missing values are replaced with the same constant. Missing values filled with the average value of an attribute is referred as *Average substitute method.* If the database is large this method takes more processing time to calculate the average of attributes and to fill the missing values. Data analysis applied to the resultant of this method will give the close result but not the exact one. *Regular Substitute Method* is considered as a common method used to fill the attribute missing values. The value that occurs more often in the attribute is used to substitute the missing values. This incomplete cleaning process gives better result than the other methods. *Estimated Attribute method* is a scientific method and is one of the most complex methods used to fill the missing values. This method uses relevant arithmetic as regress and a decision tree to predict the possible values of the lost attributes and then replace the defaults with predicted values. The methods discussed above are the usual approaches and selection of the method relies on the specific data source [32].

4 DATA CLEANING MULTI-AGENT ARCHITECTURE

Agents are made available with some initial knowledge in order to take initial decisions [2]. Data cleaning architecture describes the interaction and method of communication between the multi-agents present in the system [8][13]. The Data Cleaning Multi-Agent Architecture requires Interface Agent, Data Collection Agent, Data Cleaning Agent, Knowledge Management Agent and Message Handling Agent to perform different tasks [30]. These agents are described in the subsequent paragraphs.

Interface Agent (IA) present in the top most layer interacts with Data Collection Agent, Data Cleaning Agent, Message Handling Agent and Knowledge Management Agent. This agent submits the user provided details to the Data Collection Agent. Also, IA receives user information from the Data Cleaning Agent, interaction dialogues from the Message Handling Agent and mined results from the Data Cleaning Agent. Finally, it returns the result from the interacted agents to the user.

Data collection agent (DCOA) present in the second layer gets user details from Interface Agent (IA) and sends it to the Knowledge Management Agent (KMA) to verify the user information. The result of the verification process from the KMA is passed to the Interface Agent. Also, DCOA has the responsibility to handover all the collected information, namely the user name, file name, file type, user preference or the filter key to the Data Cleaning Agent to perform the data cleaning task.

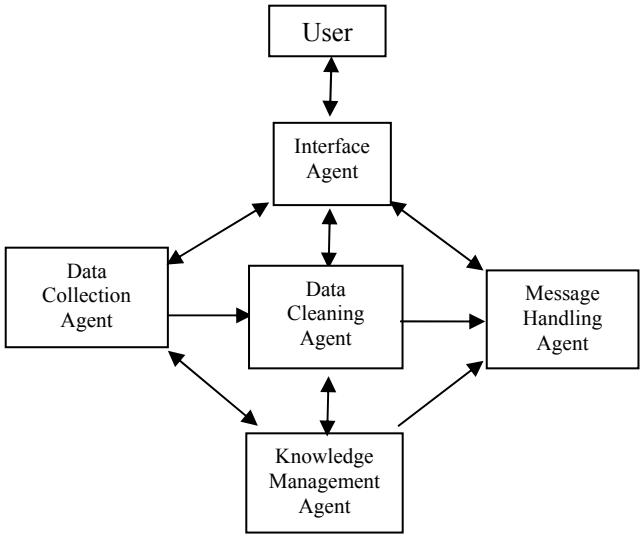


Fig 5. Multi-Agent Data Cleaning Architecture

Data cleaning agent (DCLA) in the second layer receives the user profile and data cleaning details from the Data collection agent (DCOA). The knowledge required for the data cleaning process is received from the KMA. If the KMA instructs to carryout the cleaning process then the data cleaning is performed by executing appropriate methods with the keyword. If the KMA finds that the cleaning has already been done then the result is handed over to the IA by the DCLA. DCLA uses the keyword and the file name given by the user for cleaning either by clustering or by filtering

technique. Irrelevant or the *inconsistent data* are identified by analyzing the data items which are not close to the context and are identified as a group of clusters. These clusters are either omitted or deleted from the database. The remaining data in the database after this process is sent to the IA.

Knowledge Management Agent (KMA) does the process of identification and elimination of user information. KMA possesses some initial knowledge and functions to analyze the knowledgebase. This Agent is capable of learning from the previous experience present in its knowledgebase. The KMA updates with the addition of new knowledge or by the removal of conflicting patterns that may present in the knowledgebase. At this stage either the knowledge is added to the knowledgebase or removed from it. The functionality of the KMA is to receive user details from Data Collection Agent and newly learned knowledge from Data Cleaning Agent. Also, it requests the Message Handling Agent (MHA) to prepare erroneous message or interaction dialogues to handle the situation. With the user profile and preferences it verifies its knowledgebase to decide for initiating the Data Cleaning Agent to perform the cleaning task. If the cleaning process has already been done with the specified preference in a particular file, the KMA decides not to do the cleaning process. Otherwise the cleaning process is initiated in the DCLA by the KMA. The MAS-L framework is designated for KMA-Learning. KMA initiates the Intelligence class of MAS-L Framework by instantiating the Decision Tree that invokes the evaluation function to decide the next best move.

Message Handling Agent (MHA), handles messages among the agents present in the system. KMA initiates the MHA to prepare a message when the requested file is not available in the specified path or directory and/or when the specified preference does not match with the data present in the system. During the cleaning task the Data Cleaning Agent may encounter an erroneous situation or may require the user to provide some additional information. To handle this situation Data Cleaning Agent requests the MHA to prepare either an erroneous message or user interaction dialogue. MHA sends appropriate message to the Interface Agent.

4.1 Incomplete Data cleaning using Multi-Agent Architecture

Text database cleaning focuses incomplete information present in the database. Incomplete Data cleaning procedure is described in detail in section 3. Filling of missing values present in the text database is referred as Incomplete Text database cleaning and is done with the help of Multi-Agent Data Cleaning (MADC) Architecture [25]. Agents present in MADC architecture are facilitated with the object oriented frameworks MAS and MAS-L to have reusable components and to establish intelligence. Agents involved in the cleaning process of the incomplete data cleaning task make use of appropriate methods to perform the cleaning task [1]. During the data cleaning task agents are capable of learning from the environment and react to the situation [15]. Knowledge Management Agent defined in the MADC architecture is designated as learning agent [35]. Knowledge Management Agent stores its previous experience and uses it when necessary [20]. Hence, need of agents become mandatory in the text database cleaning task and they are used to improve the performance of the system [3].

5. Agent learning in Text Data Cleaning of Incomplete Data

5.1 Agent learning process

Machine learning is the area of artificial intelligence that examines how to write programs that can learn. Machine learning is often used for prediction or classification [30]. Prediction is done based on the feed back and an agent learns through examples. When a similar prediction occurs in future the feed back is used to make the same prediction or completely a new prediction. Machine learning applied in the data mining task uses a model to represent the data. The model is a graphical structure such as neural networks or decision trees.

A decision tree is a tree where the root and each internal node are labelled with a question. The arcs emanating from each node represents each possible answer to the associated question. Each leaf node represents a prediction of a solution to the problem under consideration [19]. A Decision Tree is a prediction modelling technique used in classification, clustering, and prediction tasks. Decision Tree uses *divide and conquer* technique to split the problem space into subsets. Searching in a decision tree starts at the root node and ends successfully at the leaf node. Leaf nodes represent the successful guess of the object being predicted.

Decision tree based machine learning approach is introduced in the IMASDC architecture describes the examples in terms of attribute-value pairs that range over finitely many fixed possibilities. Also the concept to be learnt may have discrete and disjunctive description may be required to arrive at a decision. Learning decision trees classifies the training example with a finite set of attributes along with its associated values [19]. Attributes and values are used to learn the structure of the decision trees which can be used to decide the category in an unknown situation.

Learning takes place as a result of the interaction between the agent and the environment, and from observation by the agent of its own decision making processes [12]. Learning is used not only to react but also to improve the agent's ability to act in the future [22]. Agents incorporated in the Text data cleaning process works under unsupervised learning principle and they use machine learning based decision trees as a tool to learn from the environment with its past experience [33].

5.2 Decision Tree based learning algorithm for Text database cleaning of Incomplete data

The Decision Tree based learning for the Text Database cleaning is focused on incomplete data cleaning. Incomplete text database cleaning approach is described in this section.

Algorithm:

```
Incomplete-Clean()
Begin
  If Detect = 'Incomplete' Then
        // identify the incomplete database
   // the records with no value or default value are considered
   // as incomplete data and such records are extracted for
  //further processing

   Compute-attribute-missing(); // find out the att-val missing
                                //    record
   Usability();                 // compute usability of the record
   Degree-of-incompleteness(); // determine the degree of
                                // incompleteness
   If AMR = 'low' Then    //    AMR < ℓ [threshold]
    //   if the degree of incomplete incompleteness is low then
    // it is meaningful fill the missing information
   For each of the missing value attributes do
   Begin
      If Replace = 'regular' Then
          Regular-substitute-method();
           // This method computes the filling value by finding
           //  the most often occurred value in the attribute
      Else
        // use other method to replace the missing values
       // requested by the user
          Execute-app-method ();
     End if
     End;
   Else   // { AMR is high }
    // usability of the record is very low and the degree of
   // incompleteness is high
     Purge-record ();          // deletes the record
   End if
 Else  // { Detect ≠ Incomplete }
        // Agent learns not to perform any operation
// all the information present in the text database is complete
End if
End.
```

The decision tree based incomplete data cleaning of text database algorithm learns the cleaning task by identifying the attribute values and executes appropriate methods to initiate the data cleaning action by filling the missing values with appropriate values depending upon the input.

In the Data Cleaning Multi-Agent Architecture, KMA uses the Machine learning based Decision tree learning approach to implement the text database cleaning [27]. Table 1 is used as the training example for the agent to perform the incomplete data cleaning process [28]. In the Incomplete Data Cleaning learning process, KMA uses the node attributes 'DETECT', 'AMR' and 'REPLACE' to perform decision tree based Incomplete data cleaning (Fig 6). Based on the value present in the training example the KMA identifies and executes appropriate method to perform the text database cleaning [21]. The Agent learning algorithm is tested with the samples present in the training example (Table 1).

5.3 Decision Tree construction process for Text Database Cleaning of Incomplete Data

Decision tree construction process makes use of the ID3 (Induction Decision Tree Version 3) machine learning inductive algorithm. This algorithm requires computing the measures *entropy* and *information gain*[26]. The training data used in the decision tree construction process is built by using the set of node attributes along with its possible values. ID3 algorithm uses this training data as input and constructs the decision tree by placing the attribute node in its place by using the measures entropy and information gain. The entropy and information gain are computed using the following formulae.

$$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Evaluation of the intermediate nodes is done by an Evaluation function that applies the Deterministic Competitive Learning algorithm (DCL) concept. The algorithm found to be more suitable for Boolean valued function and is used in the evaluation function. The DCL algorithm changes its state and moves to one more level down in the decision tree if the Evaluation Function returns true value otherwise it remains in the same state. In other words, the algorithm learns only if the status of the output has been changed compared to the previous iteration. DCL updates the weights using the following procedure [4].

$W_j[n+1] = W_j[n] + S_j(y_j)(x - W_j[n])$

Where W is the weight vector and

$S_j(y_j) = \{ 1,$ if j is the winner

$\{ 0,$ otherwise.

The Decision Tree construction process for the Text database cleaning of Incomplete data is described below in detail.

Decision tree construction process needs to compute the measures entropy and information gain for each of the associated node attributes. After computing the measure values for all the node attributes, the algorithm chooses the node attribute with higher information gain as its root node.

The children of the root node are obtained by recursively applying the algorithm to the rest of the node attributes. The Learning function is computed using the Decision Tree and is used in the evaluation function to determine the action plan.

S.No	DETECT	AMR	REPLACE	LEARN
S1	Incomplete	Low	Regular	Yes
S2	Incomplete	Low	Other	No
S3	Incomplete	High	Regular	No
S4	Incomplete	High	Other	No
S5	Complete	Low	Regular	Yes
S6	Complete	Low	Other	Yes
S7	Complete	Low	Regular	Yes

Table 1 Sample training data for Incomplete data cleaning

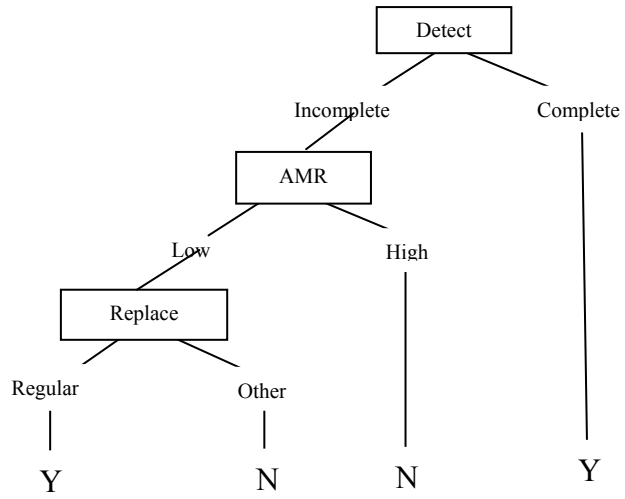


Fig 6 Decision Tree Learning Approach for Text Database Cleaning of Incomplete Data

Sample data present in Table1 is used to train the KMA to learn while cleaning the incomplete data [28]. To train the KMA the sample training data considers attributes *Detect*, *AMR* (Attribute Missing in Record), and *Replace* for decision tree construction. KMA initiates the learning process from the root node (DETECT) of the decision tree, reads the value of the training example and determines the action. If the value of the root node attribute DETECT is 'Incomplete' then KMA confirms the Incomplete data cleaning method and initiates the cleaning action by moving one level down in the decision tree and reaches the node AMR. The KMA determines not to perform any action / operation for the value 'Complete' of the root node DETECT. After reaching the node AMR, KMA reads the value of the node AMR from the training example. If the value of the node AMR is 'Low' then KMA learns to perform incomplete data cleaning process and moves one level down in the decision tree from the current node and reaches the node REPLACE at the next level. Deleting the Record from the database is done as action by the KMA, if the value of the node AMR is 'High'. The node *Replace* has values *regular* and *others* to choose either regular substitution method or other methods. If the value of the node Replace is regular then it chooses the method regular() to perform regular substitution method to fill the missing values. The value *others* of the node Replace enables the agent to any other method to fill the missing data. The decision variable *Learn* has two values namely 'yes' and 'No' to indicate the learning status of the agent.

6. INTELLIGENT MULTI-AGENT SYSTEM ARCHITECTURE FOR DATA CLEANING

Intelligent Multi-Agent System Data Cleaning (IMASDC) Architecture combines the features of Multi-Agent data cleaning Architecture, MAS and MAS-L frameworks to carry out data cleaning process (Fig 7). The MAS and MAS-L frameworks are designed using the object oriented concept and IMASDC Architecture makes use of them to improve the performance of the cleaning task. Agents present in the IMASDC Architecture inherit all the classes and interfaces from the *AgentClass* of the MAS framework, and the learning aspect is initiated by using the intelligent class of the MAS-L Framework. These agents use the *InteractionProtocol* to interact with the other agents and make use of the *AgentCommunicationLayer* to implement the communication protocol to interact with the data cleaning system. All communications are done through the *InteractioProtocol* that uses *AgentBlackBoardinformation* to exchange the message. *Blackboard* is used to establish communication between the agents available in the data cleaning architecture.

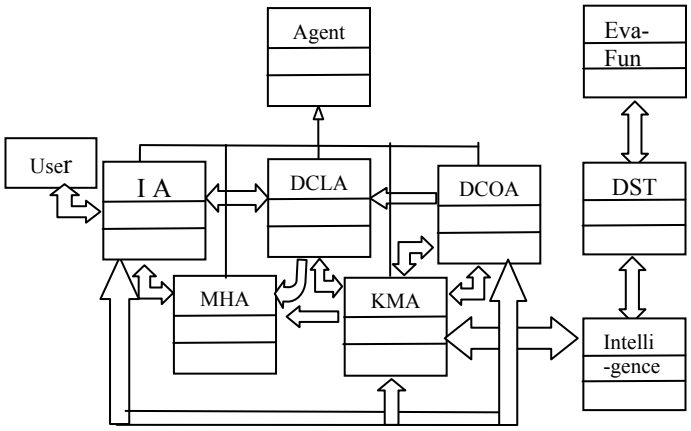


Fig 7. IMASDC Architecture using in Multi-Agent Data Cleaning Architecture, MAS and MAS-L Frameworks

Agent's communication is visualized through Black Board and is used as a common platform for all the agents to communicate [29]. Interface Agent uses the *AgentInterface* to initiate other agents. MHA uses *Processmessagethread* and

AgentMessage of the MAS framework to prepare and handle the message. KMA initiates the *Intelligenceclass* of MAS-L framework to enable the learning activity. The intelligence class in turn calls the *DecisionSearchTree* class which constructs the nodes for the Decision Tree using the ID3 machine learning algorithm.

6.1 Text Database cleaning of Incomplete data Using IMASDC Architecture

User initiates the data cleaning task by providing the source file name and the details necessary to perform the cleaning task. Multi-agent present in the data cleaning architecture receives the user provided information and determines the action with the help of the KMA. KMA uses its learning capability to initiate intelligence component and to determine the action plan. Incomplete database cleaning process uses IMASDC Architecture to fill the missing values. KMA present in the architecture uses specific key value to initiate the intelligence component to construct appropriate decision trees. The Incomplete data cleaning approach considers the attributes DETECT, AMR and REPLACE for the decision tree construction. Table 1 is used as a training sample to train the KMA and to plan for further action plan. KMA initiates the search process from the root of the decision tree and makes use of the evaluation function to determine the action. In the decision tree search process agent learns to move from the current node to one more level down in the decision tree or remain in the same node based on the value returned by the evaluation function. Text database cleaning using the IMASDC Architecture is described in detail as follows.

Interaction Agent (IA) receives the user input and sends it to the Data Collection Agent (DCOA) for further processing. DCOA sends the user details to the KMA to verify the existence of the user provided information and the method needed for the text database cleaning task. KMA makes use of the user input (file name and key word) and determines the action by verifying its knowledgebase by using the keyword and the file name. KMA checks its knowledgebase for the existence of cleaned information related to the input file. If it is available then KMA sends the cleaned details to the Data Cleaning Agent (DCLA) for forwarding to IA. If the cleaned information related to the input file is not available then the given input file is taken up for cleaning and the KMA determines one of the suitable cleaning methods. If the keyword is *Incomplete,* KMA selects the method Incomplete-Clean() method to fill the missing values of the given text database. Once the selection of the cleaning method is over, KMA provides the source file, method needed to the DCLA to initiate the cleaning action. When the cleaning task is over DCLA sends the cleaned information to the KMA to update its knowledgebase and also sends the cleaned details to the IA as the output. IA receives the output file from the DCLA and sends the cleaned file to the user. During the text database cleaning task Message Handling Agent (MHA) present in the architecture initiates interaction dialogues or messages to exchange messages among the agents.

6.2. Implementation

User provides the source file name (Patient.DB) and the keyword to the Interface Agent. Interface agent sends the details to the DCOA and it forwards the same detail to the KMA. KMA sends the source file to be cleaned along with the keyword to the Intelligence Component. Intelligence component (IC) checks the keyword and chooses incomplete-clean decision tree. After selecting the decision tree IC sends the information to the concerned decision tree and initiates the search process from its root node. Decision Search Tree component forwards the source file (*.DB extension) to the evaluation function to determine the cleaning action. Evaluation function makes use of the keyword to check the availability of the file with ***.IDC.DB** extension for the keyword 'incomplete'. ***.IDC.DB** stands for Incomplete Data Cleaning database. If the files with the extensions is present the evaluation function return 0 (without changing state) and the decision tree chooses the 'other' option to determine not to perform any cleaning operation. Decision tree forwards the cleaned file name (with ***.IDC.DB** extension) to the intelligence component. Intelligence component forwards the file to the KMA. KMA sends the cleaned file to the DCLA for further processing. KMA initiates the MHA to prepare a message about the file existence and MHA in turn sends the 'file existence' message to the user via IA.

If the file with extension ***.DB** is present and the keyword given by the user is *Incomplete* then the evaluation function chooses Incomplete-clean decision tree (Fig 6). Searching starts from the root node DETECT to initiate the cleaning action. Decision tree component makes use of the training example (Table 1) record to determine the action.

The evaluation function reads the value of the attribute as '*Incomplete*' from the training example and verifies it with the branch values of the current node. Left branch of the decision tree matches with the value present in the training example. Hence Evaluation function changes its state and returns the value 1 and moves to the left branch of the decision tree and it reaches the node AMR.

Now the evaluation function reads the value of the node attribute AMR. If the value is 'Low' then the evaluation function returns value 1. The decision tree reaches the node REPLACE by moving one level down from the current node. The decision tree sends the source file name to perform incomplete data cleaning operation to the intelligence component. . Intelligence component in turn sends the same to the KMA and KMA forwards the details to the DCLA to detect the missing values from the source text database file.

P.No	Gen-der	Visit Date	Heart Rate	Sys -BP	Dia BP	Dia Code	Adv Event
12	M	12/12/2008	80	120	100	101	0
101	M	10/4/2008	71		65	111	1
15	M	12/12/2008	53	85	65	201	1
45	M	15/3/2009	62	125		141	1
26	M		55		65		0
18	F	12/12/2008	58	115			
202	F	15/3/2009	55	90	70	121	1
101	M	10/4/2008	71		65	111	1
220	M	16/7/2009	59	100	70	141	0
145		10/4/1008				151	
15	M	12/12/2008	53	85	65	201	1
17	M	12/12/2008	85			100	1
202		15/3/2009		90	70	121	1
52	F	12/12/2008	76	110		121	0

For each of the record present in the text database, the measure *usability* of the record is computed. DCLA returns the records with missing values (missing values may either physically not present or it has default values) that are stored in the file with ***.MIS.DB** extension to the KMA. KMA stores this intermediate version of the (missing data) file and sends it to the intelligence component. Intelligence component in turn checks the usability of the missing records in ***.MIS.DB** extension file. If the usability of the record is *low* then the intelligence component enables the decision tree to determine the action plan. Decision tree initiates the evaluation function to determine the action. Evaluation function verifies the training example value for the attribute REPLACE. If the value of this node attribute is '*Regular*' the evaluation function returns the value 1 to the decision tree and the decision tree moves one level down. Now the decision tree reaches the leaf node and returns the action plan [*to fill the missing values by regular substitution method*] to the intelligence component. Intelligence component forwards this information to the KMA and KMA in turn passes the information to DCLA. DCLA executes the method provided by the KMA to fill the missing values. DCLA returns the cleaned (filled text database) file and name the file with *.**IDC.DB** extension to the KMA to update its knowledgebase. KMA receives file and stores the file in its knowledgebase and deletes the intermediate version of ***.MIS.DB** file. Also DCLA sends the cleaned output file to the Interface Agent and it sends the same to the user.

If the value of the node attribute AMR is '*High*' then the evaluation function returns value 0 without changing its state. This means that the record has more number of missing attribute values and is considered as useless record / tuple. Since the branch of this node value 'high' is a leaf node it returns the action plan as delete the tuple / record. This action is sent as a message to the intelligence component which in turn sends this message to the KMA. KMA forwards this message to the DCLA to perform record removal operation. The resultant of the process is stored with the extension ***.IDC.DB.**

6.3 Sample Text database cleaning of Incomplete data and Cleaned Text database

P.No	Gen-der	Visit Date	Heart Rate	Sys-BP	Dia BP	Dia Code	Adv Event
12	M	12/12/2008	80	120	100	101	0
101	M	10/4/2008	71	**105**	65	111	1
15	M	12/12/2008	53	85	65	201	1
45	M	15/3/2009	62	125	**70**	141	1
26	M	**12/12/2008**	55	**105**	65	**125**	0
18	F	12/12/2008	58	115	**70**	**125**	**1**
202	F	15/3/2009	55	90	70	121	1
101	M	10/4/2008	71	**105**	65	111	1
220	M	16/7/2009	59	100	70	141	0
15	M	12/12/2008	53	85	65	201	1
17	M	12/12/2008	85	**105**	**70**	100	1
202	**M**	15/3/2009	**64**	90	70	121	1
52	F	12/12/2008	76	110	**70**	121	0

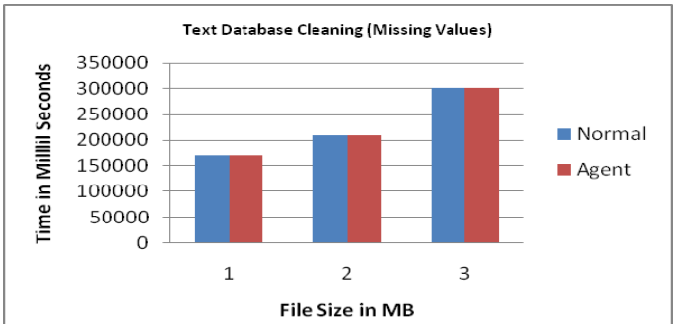
Fig 8. Filling the Missing values

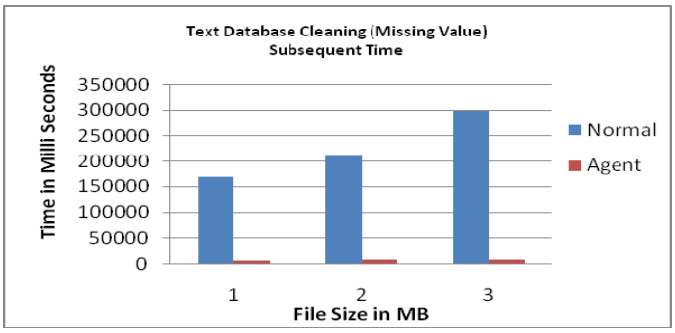
(Patient number 145 is deleted because the AMR is very high and other records with missing value is filled and displayed in bold face)

6.4. Analysis

Text database cleaning is performed in terms of incomplete data cleaning method. Text database cleaning is analyzed in terms of agent based and normal cleaning. It is observed that agent based text database cleaning produces a better performance than the normal text database cleaning method. The intelligent behavior of agents largely reduces the time required for cleaning the text database. The task of the DCLA involves the process of filling the missing values in a given text database and using it for further reference. But the reference to the already cleaned information related to a given

file reduces the task of DCLA. KMA with the learning capability of knowing the available cleaned information related to a file ensures improvement in the performance of the text database cleaning system.





The above two charts are the outcome of the analysis made over normal cleaning and agent based cleaning. Agent based text database cleaning is found to be much better than the normal cleaning procedure. Only for the first time the agent based text database cleaning and the normal text database cleaning procedure requires the same amount of time. In the subsequent process the agent makes use of its previous experience (stored in its knowledgebase) for cleaning. Use of Multi-Agents in the data cleaning architecture enormously reduces the computation time and the performance of the system is improved by the learning behavior of the agent. It is observed that IMASDC Architecture based text database cleaning gives performance over non agent based (normal) text database cleaning.

7. Conclusion

This paper proposes Intelligent Multi-Agent Data Cleaning (IMASDC) Architecture that uses the Object oriented frameworks MAS and MAS-L. Functionalities and the behaviour of the agents present in the architecture are described in detail. Among the agents present in the architecture KMA is designated to be the learning agent. To implement the learning characteristics of KMA a learning algorithm based on the machine learning tool decision trees was used. Text database cleaning using IMASDC Architecture brings the data cleaning system to have intelligence behaviour and improves the functionality of the system with higher performance than ordinary systems. Future work on this paper may use the IMASDC architecture to clean the text data, Email data and biological database.

Reference

[1] Ayse Yasemin SEYDIM, "Intelligent Agents : A Data Mining Perspective", CiteSeer-IST Scientific Literature Digital Library, 1999.

[2] Alex Bordetsky, "Agent-Based Support for Colloborate Data Mining in System Management", Proceedings of the 34th Hawaii international conference on System Science, 2001,vol ©IEEE, ISBN 0-7695-0981-9/01.

[3] Dae Su Kim, Chang Suk Kim, Kee Wook Rim, "Modeling and Design of Intelligent Agent System", International Journal of Control Automation and Systems, 2003, Vol 1 No 2.

[4] Dong-Chul Park,"Centroid Neural Network for Unsupervised Competitive Learning", IEEE Transactions on Neural Networks, 2000, Vol 11 © IEEE, ISBN S1045-9227(00)02998-2.

[5] Erhard Rahm, Hong Hai Do, "Data Cleaning: Problems and Current Approaches", conference on data cleaning,2000.

[6] Fayad. M, D.Schmidt, "Building Application Frameworks: Object-Oriented Foundations of Design",John Wiley & Sons,1999.

[7] Feldman R , Fresji M, Hirsh H, Aumann Y, Liphstat O, Schlter Y , Rajman M, "Knowledge Management : A Text Mining Approach" ,Proceedings of the 2nd International conference on Practical Aspects of Knowledge Management(PAKM98),1998.

[8] Ferber, J, "Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence", Addison-Wesley Pub Co, 1999.

[9] Garcia, A., Silva, V., Lucena, C., Milidiú, R. "An Aspect-Based Approach for Developing Multi-Agent Object-Oriented Systems", Simpósio Brasileiro de Engenharia de Software Rio de Janeiro, 2001.

[10] Garcia, A., Lucena, C. J., Cowan, D.D., "Engineering Multi-Agent Object-Oriented Software with Aspect-Oriented Programming", Elsevier, 2001.

[11] Garcia, A., Lucena, C. J.,"An Aspect-Based Object-Oriented Model for Multi-Agent Systems", 2nd Advanced Separation of Concerns Workshop at ICSE-2001, 2001.

[12] Garro, A., Palopoli, L.,"An XML Multi-Agent System for e-Learning and Skill Management", Third International Symposium on Multi-Agent Systems-Large Complex Systems and E-Businesses (MALCEB-2002), 2002.

[13] Gerhard Weiss, "Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence", The MIT Press, 1999.

[14] Helena Galhardas, Daniela Florescu, Dennis Shasha, Eric Simon, Christian-Augustin Saita, "Declarative Data Cleaning: Lanaguage, Model and Algorithms", Proceedings of the 27th VLDB conference, 2005.

[15] Ioerger, T. R. He, L. Lord, D. Tsang, P , "Modeling Capabilities and Workload in Intelligent Agents for Simulating Teamwork", Proceedings of the Twenty-Fourth Annual Conference of the Cognitive Science Society (CogSci-02),2002, PP482-487.

[16] Jiawei Han , Micheline Kamber, "Data mining: Concepts and Techiniques" , Morgan Kaufmann Publishers- Elsevier, 2001.

[17] Jie Tang, Hang Li, Yunbo Cao, Zhaohui Tang , "Email Data Cleaning", Proceedings of the KDD'05 , 2005, vol © ACM, ISBN 1-59593-135-X/05/0008.
[18] Kollayut Kaewbuadee, Yaowadee Temtanapat, Ratchata Peachavanish, "Data Cleaning using FD from Data Mining process", Proceedings of conference, 2003.

[19] Margaret H Dunham, Sridhar S " Data Mining – Introductory and Advanced Topics" Pearson Education, Inc., Copyright ©2003, ISBN 81-7758-785-4

[20] Massimo Cossentino, Antonio Chella and Umberto Lo Faso, "Designing agent based systems with UML" ,international conference on agents, 2006.

[21] Raymond J Mooney , Razvan Bunescu, "Mining knowledge from text using information extraction", SIGKDD Explorations, 2003, vol 7 issue 1, pp 3-10.

[22] Russell, S., Norvig, P., "Artificial Intelligence, A Modern Approach", Prentice-Hall, 1995.

[23] Sardinha, J.A.R.P., Ribeiro, P.C., Lucena, C.J.P., Milidiú, R.L., "An Object-Oriented Framework for Building Software Agents", Journal of Object Technology,2003, Vol 2No.1.

[24] Sardinha, J.A.R.P., Milidiú, R.L., Lucena, C.J.P., Paranhos P , "An Object-Oriented Framework for Building Intelligence and learning properties in Software Agents",Journal of object Technology, 2004.

[25] Shahram Rahimi and Norman F. Carver , "A Multi-Agent Architecture for Distributed Domain-Specific Information Integration", Proceedings of the 38th Hawaii International Conference on System Sciences,2005, vol ©IEEE, ISBN 0-7695-2268-8/05.

[26] Soman K P, Shyam Diwakar, Ajay. V, "Insight into Data Mining Theory and Practice", PHI, 2008, ISBN 978-81-203-2897-6.

[27] Stader, J., Macintosh, A., "Capability Modeling and Knowledge Management- In Applications and Innovations in Expert Systems VII", 19th Int Conf on Knowledge-Based Systems and AAI Springer-Verlag, 1999, pp 33-50 ISBN 1-85233-230-1.

[28] Symeonidis A L, Chatzidimitriou K C, Athanasiadis I N, Mitkas P A , "Data Mining for agent reasoning : A synergy for training agents", Engineering Applications of Artificial Intelligence- Elsevier, 2007.

[29] Tobin J Lehman, Stephen W. McLaughry, Peter Wyckoff, "T Spaces : The next Wave", Proceedings of international conference on Machine Learning, 1999.

[30] Tom M Mitchell, "Machine Learning", McGrawHill,1997, ISBN 0070428077.

[31] Weiss, G., "Multiagent systems: a modern approach to distributed artificial intelligence", The MIT Press, 2000.
[32] William E Winkler, "Data Cleaning Methods", Conference SIGKDD'03 , 2003, vol © ACM, ISBN 1-58113-000.

[33] Winston, PH. "Artificial Intelligence", Addison Wesley, 1992.

[34] Wooldridge, M, Jennings, N. R., Kinny, D."The Gaia Methodology for Agent-Oriented Analysis and Design",Kluwer Academic Publishers,2000.

[35] Zili Zhang, Chengqi Zhang and Shichao Zhang, "An Agent-based hybrid framework for database mining", Taylor & Francis Group Applied Artificial Intelligence, 2003,pp 17:383-398.

[36] Zhang Jin , "Research on Data Cleaning in Data Acquisition", conference on data mining, 2001.