Adaptive Knowledge Discovery for Decision Support in Intensive Care Units

PEDRO GAGO[1] AND MANUEL FILIPE SANTOS[2]
[1]Escola Superior de Tecnologia e Gestão
Instituto Politécnico de Leiria
Leiria
PORTUGAL
pgago@estg.ipleiria.pt
[2] Departamento de Sistemas de Informação
Universidade do Minho
Guimarães
PORTUGAL
mfs@dsi.uminho.pt

Abstract: - Clinical Decision Support Systems (CDSS) are becoming commonplace. They are used to alert doctors about drug interactions, to suggest possible diagnostics and in several other clinical situations. One of the approaches to building CDSS is by using techniques from the Knowledge Discovery from Databases (KDD) area. However using KDD for the construction of the knowledge base used in such systems, while reducing the maintenance work still demands repeated human intervention. In this work we present a KDD based architecture for CDSS for intensive care medicine. By resorting to automated data acquisition our architecture allows for the evaluation of the predictions made and subsequent action aiming at improving the predictive performance thus enhancing adaptive capacities.

Key-Words: - Clinical Decision Support, Intelligent Decision Support Systems, Knowledge Discovery, Intensive care, Ensembles, Multi-Agent Systems

1 Introduction

Clinical Decision Support Systems (CDSS) are computer systems designed to impact decision making about individual patients at the point in time that these decisions are made [1]. Such systems may help physicians perform a large number of tasks related to patient care. Some of the most important factors associated with their success [2] [29] are:

- providing alerts/reminders automatically as part of the workflow;
- providing the suggestions at a time and location where these decisions are being made;
- providing actionable recommendations; and
- computerizing the entire process.

Thus, integration into both the culture and process of care is going to be necessary for these systems to be optimally used. In fact, there are several issues that must be addressed in order to promote the use of CDSS. Some are issues of how the data are entered. Others are related to the development and maintenance of the knowledge base. Finally, there are also physician motivation issues.

An increasing number of Intensive Care Units (ICUs) are using computer applications in their daily routine [3]. Those applications usually encompass data integration, where data from several sources is combined in a central repository. Moreover, the availability of medical data in digital support allows for the progressive replacement of the traditional paper based records with the associated gains in quality of care [3-6]. In addition, the availability of data may ease the development of CDSS through the use of Knowledge Discovery from Databases (KDD) techniques. However, there are issues with such DSSs as there are several problems regarding data availability and quality and also there is the need for frequent system updates in order to maintain a good performance.

In this paper we present the work we have been developing in building one such CDSS. The architecture we propose allows the development of (semi) autonomous CDSSs as data acquisition, pre-processing and model updates can be performed without human intervention.

Our system (INTCare) is being tested in a real environment, the Intensive Care Unit (ICU) of Hospital Geral de Santo António (HGSA), Oporto, north of Portugal.

The following sections will introduce the problem definition and the state of the art, followed by the INTCare system formal description in terms of the Agent-Based paradigm. Then, the obtained results will be presented by means of their usability and efficiency. Next, a discussion will be set over the critical aspects of the INTCare system. Finally, the paper will be concluded, showing also perspectives of future work.

2 Background

2.1 Clinical Decision Support Systems

One can now find CDSS being used with very diverse objectives, including [7]:

- alerts and reminders;
- diagnostic assistance;
- therapy critiquing and planning;
- prescribing decision support;
- information retrieval;
- image recognition and interpretation.

Nevertheless, CDSS are not as widespread as would be expected given the potential benefits [8]. Some insights into the reasons behind some CDSS failures are reported in [9] and include several ideas with one of the most striking being that "the critical impediment to the development of decision programs useful in medicine lays in the impossibility of developing an adequate database and an effective set of decision rules". Also, CDSS that require significant data entry are not likely to be used on a regular basis [10]. The architecture we propose attempts to circumvent these difficulties includes an agent for automatic data gathering from the available sources such as bedside monitors or the patient's Electronic Health Record (EHR). Moreover, by using Knowledge Discovery in Databases (KDD) techniques INTCare is able to use the available data in order to automatically build and/or maintain the knowledge base.

2.2 Knowledge Discovery from Databases and Data Mining

The interest in Knowledge Discovery from Databases (KDD) and Data Mining (DM) arose due to the rapid emergence of electronic data management methods. In 1997, the Gartner Group suggested that DM is one of the top five key technologies that will have a major impact in the industry within the next years [11]. In effect, these techniques are now widespread and applications can be found in diverse areas.

Within the Medicine arena, huge databases, with large, complex and multi-source information (e.g. text, images or numerical data), are commonplace. However, human experts are limited and may overlook important details. Furthermore, the classical data analysis (e.g. logistic regression) breaks down when such vast amounts of data are present. Hence, an alternative is to use automated discovery tools to analyze the raw data and extract high level information for the decision-maker [12].

More recently, there has been an emergent DM research area that involves the use of ensembles for supervised learning, where a set of classification/regression models are combined in some way to produce an answer [13].

Knowledge discovery is a priority, constantly demanding for new, better suited efforts. Systems or tools capable of dealing with the steadily growing amount of data presented by information system, are in order [27].

3 CDSS Architecture

In this section we present our architecture for decision support in the intensive care unit.

Most CDSS have three main components: the knowledge base, the inference or reasoning engine and a mechanism to communicate with the user [14]. However, such architecture usually demands manual updates to the knowledge base and may require extensive data entry by the users of the system. Moreover, the need for manual updates to the knowledge base is one of the factors for CDSS failure [8]. Also, doctors tend not to use an application that demands them to repeat data entry [15].

In this architecture we opted to include a component that is responsible for data acquisition and pre-processing. This component will connect to whatever data sources are available and populate a data repository that can be used for automatic updates to the knowledge base. It is presented as an autonomous component as it is fundamental to autonomous operation. The architecture is presented in Figure 1.

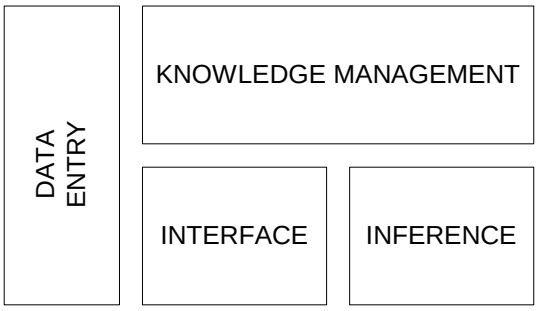

Figure 1 – The CDSS architecture

Data entry is of utmost importance for the correct operation of INTCare.

The Knowledge Management component maintains the prediction models and also includes allowing the evaluation of the predictions made. Moreover, a performance evaluation sub-component monitors the accuracy of the predictions made and when necessary will trigger model updates. We advocate the use of ensembles of models as they have been shown to perform better than single models [13] and allow for smooth updates. In fact, replacing a model in an ensemble is much less disruptive than having a single predictive model and replacing it for another.
The inference component responds to requests from the user (via the interface component). It retrieves the necessary data from the available sources and uses the adequate predictive models from the knowledge management component in order to present the user with an answer. The interface component allows the physicians to ask for predictions for specific patients.

A system based in this architecture is capable of functioning correctly when there is no information regarding the quality of the predictions made. However, if such information is made available, there are mechanisms that may force an automatic update of the predictive models being used thus allowing for an improved performance.

4 INTCare System

The proposed architecture was implemented in a real setting in order to evaluate its usefulness. The chosen location was an ICU of the Hospital de Santo António in Oporto, Portugal. It is a good location as there is a wealth of available data that can be collected either from the bed-side monitors or from the EHR.
Our initial goal was that of predicting hospital mortality for ICU patients, given some data from the first 24 hours after ICU admission. As discharge data is entered into the EHR it is possible to (automatically) evaluate the predictions made.

The INTCare (Figure 2) is an agent based system, composed by several semi-autonomous agents in charge for the functionalities inherent to the system. Formally, the INTCare system is defined as a tuple:

$$\Xi \equiv < C_{INTCare}, \Delta_{INTCare}, a_{gat}, a_{vsa}, a_{cade}, a_{ada}, a_{pp}, a_{cde}, a_{dm}, a_{ens}, a_{pf}, a_{mi}, a_{dr}, a_{pd}, a_{sc}, a_{int}, a_{ic} >$$

where:
$C_{INTCare}$ is the context and corresponds to a logical theory, represented as a triple <Lg, Ax, D>, where Lg stands for an extension to the language of programming logic, Ax is a set of axioms over Lg, and D is a set of inference rules;
$\Delta_{INTCare}$ is the set of bridge rules defining the interaction among the systems' components (the agents);
a_{gat}, …, a_{ic} are the system's agents.

4.1. Description of the Agents

The INTCare agents follow the subordinated architecture [16], they are social computational entities, semi-autonomous, reactive, with internal-state, and pre-defined goals incited by the system creators, whose activity contributes to the goal of the overall system. In a technical perspective the INTCare agents are high granularity objects aggregating a great number of capacities.

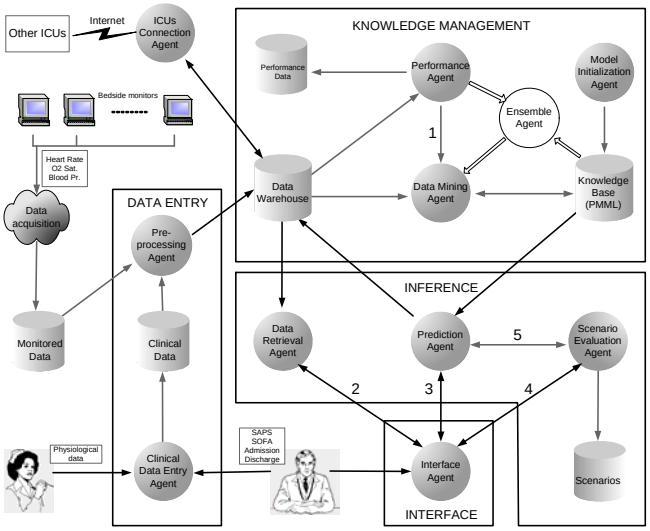

Figure 2 - The INTCare system

The social model is static, pre-defined, and incorporated into the system by the developers. The intelligent behavior, the accuracy, the robustness, the

flexibility and efficiency of this kind of system emerges from the agents and their interaction. The overall behavior is the combined result of its purpose and its interaction with the environment [17]. The next lines describe the agents' functionalities:

Gateway *(a_{gat})* is responsible to capture the vital signal data from bedside monitors. This data is packed into HL7 messages and sent to the Vital Signs Acquisition Agent;

Vital Signs Acquisition *(a_{vsa})* is an AIDA process that parses the HL7 messages, extracts information blocks and stores them in the database tables: **UCI_PATIENT_HL7.**

The HL7 message starts with the header "MSH" and it is separated with "|" and "^". This agent needs to split the message into individual data information. The data is verified and, if the information is correct, the agent performs the next steps. For the PID, PV1, OBR and OBX variables, it reads the information from the gateway splits the hl7 message and gets the required data to database.

For optimization purpose, if more than one message is received within one minute, an algorithm is applied so that only one message per minute is stored in the database. Due to the high number of null values, it was necessary to perform some optimization. The reason for this high rate is that the system can look for the values of physiological parameters more than once per minute. When this happens the gateway may get more than one message in the same minute. Some of these messages will have null values for the parameters that had been correctly collected in the previous time.

The solution found was to create an algorithm to gather all the values read in the same minute and for the same parameter and calculate the correspondent average. If it receives more than one value, it will store the average; if it can't collect any value in this minute, the field stays as a blank (null). Otherwise, it stores the single value collected. In conclusion, when the number of messages received per minute is greater than 1, it calculates the average of each one of the variables and inserts them into the database as a single record:

```
MSG_Data_Average algorithm
    If count(msg_per_min)>1 Then
        avg(msg_obx_result)
        Insert into database (avg_msg_obx_result)
    Else
        Insert into database (msg_obx_result)
```

With this algorithm the number of nulls present in the tables is reduced in 52% (from 60% to 8%).

Clinical Analysis Data Entry (a_{cade}) is responsible for capturing the clinical data from the lab results that are done in the hospital;

Clinical Data Entry (a_{cde}) is responsible for capturing the clinical data from the medical and nursing staff [24], especially from nursing records;

AIDA (a_{ada}) is an agency to archive and to disseminate medical exams and results. This agency will supply the lab results and nursing records through the clinical analysis data entry agent and clinical data entry agents [28];

Pre-Processing (a_{pp}) agent is responsible for the correct linking of all the values in order to create a valid (even if limited in scope) medical record for the patient [24] . This agent is in charge of solving some data acquisition problems.

Before data is consolidated in the data warehouse, the agent verifies the data in order to remove null values and correct the values that are out of range. It proceeds with the copy of the values received from bedside monitors, electronic nurse records and lab results, examines them and derives new fields. This agent analyzes all values acquired and only puts in the data warehouse the values that are acceptable and minimally correct;

Data Mining *(a_{dm})* is responsible for the retrieval of the relevant data in order to make possible the application of AI algorithms to train new models (train), whenever requested by the Performance (a_{pf}) agent, storing them into the Knowledge Base. After training, the models are stored in the Knowledge Base;

Ensemble agent *(a_{ens})* intended to enhance predictive performance by combining several models (e.g. using a majority voting scheme) in order to produce an efficient answer [13].

Two different ensemble evolution strategies were evaluated and compared to a "traditional" ensemble (Configuration A). In the first one (Configuration B) the model weights are changed after the evaluation of each batch of records. The weights update procedure is also included in the second configuration (Configuration C). Also, in this configuration new models are created using the records in each batch and the poor performing

models (those with negative weights) are removed from the ensemble.

In order to investigate the effect of evaluating batches of records of different sizes we tested batches of 10, 20, 50, 100 and 200 records.
The rationale behind this is that bigger batches may reduce the responsiveness of the system and thus lower its tendency for over training.
In configurations B and C, in order to evaluate the effects of allowing changes in the model's weights we decided to adjust the weights after each prediction. The models that had made a correct prediction had their weights increased. Those who failed had their weights reduced. We started by doing this after each prediction, but then investigated whether evaluating batches of records before making the weights updates would lead to better results. We tested updating the weights after each batch of 10, 20, 50, 100 and 200 records. After each batch of predictions (of size N) we had the number of correct predictions (C). First we calculated the fraction of the increment for each weight:

$$P = \frac{C - \frac{N}{2}}{\frac{N}{2}}$$

where P is positive if the model is correct in more than half of the records in the batch being considered. It is zero for those cases where exactly half of the answers are correct and it is negative for the remaining cases. If all the predictions are correct, the value for P is 1, if all are wrong the value is -1. The weight update is then the result of equation 2 where wi stands for the weight of model i. (By dividing P by 10*number of models we assure that the weight changes are not too abrupt (the initial weights are set at $\frac{1}{number\ of\ \mathrm{mod}els}$)):

$$w_i = w_i + P\frac{1}{10 * number\ of\ \mathrm{mod}els}$$

In configuration C new models were added to the ensemble after the evaluation of each batch of records. Two new models are created: one decision tree and one neural network. Both are trained on that batch of records using the same method described above for the initial training of the ensemble. The correspondent weight is equal to the average of the weights of the other models in the ensemble. Next, we present the algorithm (as in DWM, we used the term "expert" instead of "model"):

Weight Updates Algorithm

$\{\mathbf{x}, y\}$: training data
$\{e, w\}_m^1$: experts and their weights
p : number of records in each batch
δ_i : fraction of weight increment for expert $_i$

num = 0
for i = 1,...,n
  $answer \leftarrow 0$
  for j = 1, ..., m
    predicted $\leftarrow$ Classify(e $_j, x_i$)
    if (predicted = y_i)
      num $_j \leftarrow$ num $_j + 1$
    end if
    answer $\leftarrow$ answer + predicted * w_j
  end for
  output answer
  if (i mod $p = 0$)
    $\boldsymbol{\delta} \leftarrow$ Increment(num, p)
    $\mathbf{w} \leftarrow$ UpdateWeights($\mathbf{w}, \boldsymbol{\delta}$)
    $\{e, w\} \leftarrow$ DeleteNeg($\{e, w\}$)
    $m \leftarrow m + 1$
    $e_m \leftarrow$ Create-DecTree(precords)
    $w_m \leftarrow \bar{w}$
    $m \leftarrow m + 1$
    $e_m \leftarrow$ Create-NNetwork(precords)
    $w_m \leftarrow \bar{w}$
    $\mathbf{w} \leftarrow$ Normalize($\mathbf{w}$)
  end if
end for

The function Increment computes the value of the increment as described in 1 and UpdateW eights updates the experts weights as indicated in 2. DeleteNeg removes from the ensemble all experts with negative weights. Moreover, Create-DecTree creates a decision tree and Create-NNetwork creates a neural network. Both functions use the records from the last evaluated batch. Finally, the function Normalize normalizes the experts' weights (so that the sum of the weights is 1). The system outputs its prediction (answer) for the record under evaluation.
To evaluate the results we used the average of the values of the area under the Receiver Operating Characteristic curve (AUC ROC) obtained after 30 runs of each experiment. The ROC curves are often used in the medical area to evaluate computational models for decision support, diagnosis and prognosis [30, 31]. A model presenting an AUC of 1 has perfect discriminative power (perfect predictive

ability) while a value of 0.5 corresponds to random guessing.
We divided the available data into two mutually exclusive datasets. Models were created using the first dataset. Those models were then evaluated using the second dataset. Several experiments were conducted with different parameter settings with regard to the frequency of weights updates and the possibility of creating of new models.
We started by evaluating the performance of the static ensemble (no changes are made to the ensemble during the evaluation of the records). With no changes in the ensemble composition the AUC ROC was 78.80% ±0.93 %.
Next we tested configurations B and C investigating the effect of using different intervals for evaluation before the weights were changed. Different configurations include weight changes after every record was evaluated or after batches of 10, 20 , 50, 100 or 200 records. In Table 1 we present the AUC under the ROC curve for each of the configurations considered, allowing us to see the ensembles with better discrimination capabilities (better able to distinguish between the patients with outcome 0 or 1).

Table 1 - Results for the ensemble evolution (% of AUC ROC).

Config.	B	C
1	78.79±0.93	–
10	78.72±0.93	85.05±0.07
20	78.61±0.93	84.83±0.10
50	78.44±0.92	84.58±0.15
100	78.03±0.92	84.41±0.21
200	77.45±0.91	82.94±0.33

B - Weight updates; C - Weight updates and model creation and elimination

These results seem to indicate that it is best to update the models weights immediately after the evaluation of each record. In order to further clarify this point we decided to evaluate the ROCs in a segmented manner. Considering batches of 200 records and computing the AUC ROC for each of the batches we got the results that are show in Figure 3. The graph shows an example of the evolution of the partial AUC ROC values for one of the experiment runs.

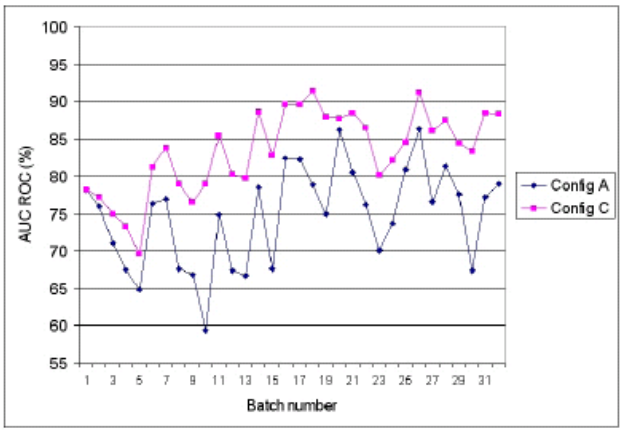

Figure 3 - Evolution of AUC ROC values over time

Performance (a_{pf}) agent continually consults, in a proactive way, the the Data Warehouse for updates that allow statistics collection (e.g. discharge data that may or may not confirm a prediction made), as a base to calculate a set of assessment parameters maintained in the Performance Database. The evaluation metrics include classification accuracy, sensitivity and specificity values [18]. These statistics are updated every time new relevant information is collected. Whenever the collected statistics show that the performance has fallen below a predefined threshold (a configuration parameter) a new model is requested in order to allow the replacement of the poor performing one;

Model Initialization (a_{mi}) agent populates the Knowledge Base with the models obtained from off-line training. This agent is currently used only when first starting INTCare;

Data Retrieval (a_{dr}) agent is an information agent, whose only objective is that of retrieving, from the Data Warehouse, the information requested by the interface agent and returning it;
Prediction (a_{pd}) agent answers user questions by applying the adequate models contained in the Knowledge Base to the data stored in the Data Warehouse. Next, results are sent back to the calling agent and if the calling agent is the interface agent, the prediction made is recorded into the Data Warehouse;

Scenario Evaluation (a_{sc}) agent makes it possible to the doctor to create and evaluate what-if scenarios. After receiving the data from the interface agent, the scenario evaluation agent requests a forecast from the prediction agent, the scenario is then stored in the Scenarios Database and the result is sent back to the interface agent;

Interface(a_{int}) agent allows (web-based) interaction with the system by providing an easy way for doctors to request prognostics and evaluate scenarios. Whenever new data is needed, this agent messages the data retrieval agent;

Connection Agent *(a_{ic})* is in charge for the knowledge interchange among the various instances of the INTCare system running in the ICUs.

4.2 Agents' messaging

The system has various agents responsible for the necessary tasks related to the data acquisition process.

The a_{vsa} agent processes the monitored data. When the gateway receives the vital signs from the monitors, sends an HL7 message (*M1*) to the vital signs acquisition agent.

Next, we can see an example of a HL7 message:

MSH|^~\&|*DHV* |***h2***|***h3***|***h4***|||*ORU^R01*|***h1***|*P*|*2.3.1*
PID|*1*||***d1***||***d2***
PV1|*1*|*U*|***v1***
OBR|*1*|||*DHV*|||***r1***|
OBX|***x2***|*NM*|***x3^x4^^^x5***||***x6***|***x7***|||||*R*||||***x1^ v1***||

Table 2 presents the meaning of each variable involved in the exchange of messages between the agents' a_{gat} and a_{vsa}.

Table 2 - HL7 message variables

h1	Version ID
h2	Sending Facility
h3	Receiving Application
h4	Receiving Facility
d1	Patient ID (Internal ID)
d2	Patient Name
v1	Assigned Patient Location
r1	Observation Date/Time
x1	Producer's ID
x2	Value Type
x3	Observation Identifier (cod)
x4	Observation Identifier (cod2)
x5	Observation Identifier (descp)
x6	Observation Value
x7	Units

The a_{ada} agent exchanges messages with the a_{cad} agent. When the a_{ada} agent receives lab results, it sends a message (*M2.1*) notifying that new data is available. The a_{cad} agent reads the message and sends one (*M2.2*) with the requested variables (Table 3, column 4). Finally, the a_{ada} agent sends the message (*M2.3*) with the required data.

When the nursing records are filled in, a_{ada} agent sends a message (*M3.1*) to the a_{cde} agent informing about the new data (Table 3, column 2). A_{cde} agent sends a message to the a_{ada} agent with the requested data and a_{ada} agent sends back a message with the required data.

Table 3 - Clinical Variables

e1	Urine Output	c1	Billirubin
e2	Glasgow	c2	Creatinine
e3	Amines	c3	Blood Platelets
e4	SOFA		

Figure 4 – Sequence diagram of the messagessummarizes the agents' messaging process described above.

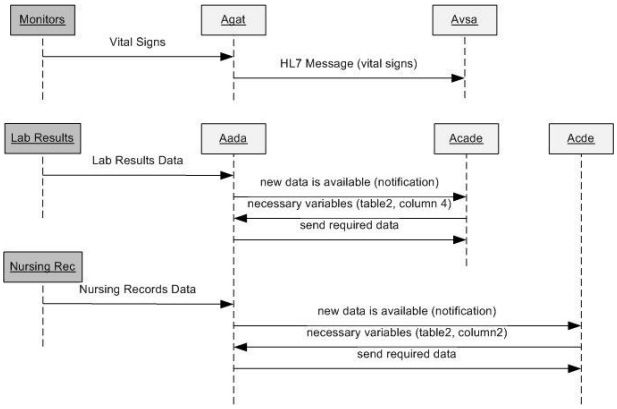

Figure 4 – Sequence diagram of the messages

5 Deployment and Experimentation

In this section we describe both the deployment process and some of the experiments we have conducted.

INTCare is an application programmed in Java that utilizes the WEKA library [19]. Its deployment was controlled, having started with data collection from a single bed and is now and has evolved to the current situation where data is collected from all the beds in the ICU unit. Data collection depends on proprietary software from the suppliers of the bed-side monitors. Also, the system connects to the hospital network to access the relevant clinical records, thus reducing the need for data entry by the physicians.

After some initial work with offline data [20, 21] we tested the INTCare system using data collected via the bed-side monitors [22]. It was a somewhat limited experience as it was performed on a small dataset as at the time data acquisition was available for a single ICU bed. Nevertheless, both the results and the physicians' comments were encouraging.
Other experiments regarded how show the model updates be performed so that the system is able to function without the need for human intervention.
Two different experiments [23, 24] showed that good performance can be achieved through the use of ensembles that are updated following a procedure based on the Dynamic Weighted Majority [25].

6 Contributions and limitations

In order to be successful CDSS must merge into the existing clinical environment and be as inconspicuous as possible to physicians and nursing staff. Manual data entry must be kept to an absolute minimum avoiding the need for duplicate data entry. Moreover, the knowledge base must be kept up to date. By connecting to the existing electronic records INTCare greatly reduces the need for manual data entry from the medical staff.
Moreover, INTCare works in a non-intrusive way as it provides predictions on request and does not burden physicians with unwanted alarms and recommendations.
Also, the automatic updates to the knowledge base lower the maintenance costs and allow the system to maintain a good predictive performance.
With these characteristics INTCare is a system that blends into the existing environment and operates without placing any extra requirements on the medical staff.
As a byproduct, the deployment of INTCare allows for data collection that may be used in future medical studies.

The architecture allows for the growth of the number of predictive objectives through simple changes in the agents.
A computer based system has other advantages such as not being subject to between-physician variability in prognostic skill or beliefs regarding the construct of "futility". Furthermore model-based predictions may be more equitable because they do not incorporate value-based judgments regarding the ''worth'' of one life over another [26].

7 Conclusions and further work

Clinical Decision Support Systems are potentially very useful having being used in conjunction with CP Order Entry. Current uses include alerts for drug interaction, etc.
Reports on the effective use of such systems stress the need for integration into the existing environment and the difficulties found when there is a need for knowledge base updates.
The INTCare architecture allows for the development of CDSS that blend into the existing systems as data acquisition is done by automatically connecting to existing systems and knowledge base updates are automatically performed in response to self evaluation. The resulting system can be deployed as part of the informatics infrastructure with prediction results being shown to doctors when they require them. Regardless if this, the initial stages of deployment required great effort so that automatic data collection could work as intended. Indeed, nurses were asked to adhere more strictly to guidelines regarding the use of bed-side monitors and authorizations had to be obtained so that access to data over the hospital's networks could be possible.
The major contributions of this approach are:

- this architecture is being tested in a real environment that demands INTCare to be customized in order to answer physician needs;
- it allows for a seamless integration into the work processes in the ICU thus increasing the chances that it will be used in daily work;
- the use of ensembles of diverse predictive models allows for a better predictive performance;
- mechanisms for automatic update of the models in the knowledge base are included;
- the overall KDD process is implemented in an automatic manner.

At the present time there are still some INTCare agents to be fully implemented, as is the case with the scenario evaluation agent. Future work includes the implementation of these agents and the evaluation of other ensemble methods.

Acknowledgements

The INTCare project is financially supported by FTC (PTDC/EIA/72819/2006).

References:

[1] Berner, E. S. and La Lande, T. J., Berner, E. S. (ed.) *Clinical Decision Support Systems - Theory and Practice, Overview of Clinical Decision Support Systems*, Spinger, 2007, pp. 3-22.

[2] Kawamoto, K., Houlihan, C.A., Bala, E.A., Lobach, D.F., *Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success,* BJM, 2005.

[3] Manjoney, R., *Clinical information systems market - An insider's view*, Journal of Critical Care, Vol. 19, No. 4, 2004, pp. 215-220.

[4] Zanier, E., Ortolano, F., Ghisoni, L., Colombo, A., Losappio, S. and Stocchetti, N., *Intracranial pressure monitoring in intensive care: clinical advantages of a computerized system over manual recording*, Critical Care, 2007, Vol. 11, R7.

[5] Bosman, R., Rood, E., Oudemans-van Straaten, H., Van der Spoel, J., Wester, J. and Zandstra, D., *Intensive care information system reduces documentation time of the nurses after cardiothoracic surgery*, Intensive Care Medicine, Vol 29, 2003, pp. 83-90.

[6] Fraenkel, D. J., Cowie, M. and Daley, P. *Quality benefits of an intensive care clinical information system*, Critical Care Medicine, Vol. 31, No. 1, 2003, pp. 120-125.

[7] Coiera, E. Guide to Health Informatics – 2nd Ed., Hodder Education, 2003.

[8] Carter, J. Berner, E. S. (ed.) *Clinical Decision Support Systems - Theory and Practice Design and Implementation Issues*, Spinger, 2007, pp. 64-98.

[9] Engle R.L. Jr., *Attempts to use computers as diagnostic aids in medical decision making: a thirty-year experience*. Perspect Biol Med, Vol 35, No. 2, 1992, pp.207–219

[10] Clayton, P.D. and Hripcsak, G. *Decision Support in Healthcare*, Int. J. Biomed Comput, Vol. 39, 1995, pp. 59-66.

[11] Lee S. and Siau K., *A review of data mining techniques*, Industrial Management & Data Systems, Vol. 101, No. 1, 2001, pp. 41-46.

[12] Hand D., Mannila H., Smyth P., *Principles of Data Mining*, MIT Press, Cambridge, MA, 2001.

[13] Dietterich T., *Ensemble methods in machine learning*, In: Kittler and Roli (eds), Multiple Classifier Systems, 2001, LNCS 1857, Springer, pp 1-15.

[14] Tan J.K.H., and Sheps, S. *Health decision support systems*, Aspen Publishers, Inc., 1998.

[15] Clayton PD, Hripcsak G. *Decision support in healthcare*, Int J Biomed Comput, Vol. 39, No.1, 1995 pp.59–66.

[16] Weiss G., *Multiagent Systems A Modern Approach to Distributed Artificial Intelligence*, MIT Press, Cambridge MA, 1999.

[17] Vahidov R., Kersten G., *Decision station: situating decision support systems*, Decision Support Systems, Vol. 38, No. 2, 2004, pp. 283-303.

[18] Silva A., Cortez P., Santos M.F., Gomes L., Neves J., *Multiple organ failure diagnosis using adverse events and neural networks*, In: Seruca I. et al. (eds), Enterprise Information Systems VI, 2005, Springer.

[19] Ian H. W. and Eibe F., *Data Mining: Practical machine learning tools and techniques*, *2nd Edition*, Morgan Kaufmann, San Francisco, 2005.

[20] Santos M.F., et al., *Augmented Data Mining over Clinical Databases: Using Learning Classifier Systems*, In: Proceedings of the Fourth International Conference on Enterprise Information Systems (ICEIS 2002), INSTICC, pp. 512-516.

[21] Santos M.F., Pereira J., Silva A., *A cluster framework for data mining models – an application to intensive medicine*, In: Chen C-S. et al. (eds) Proceedings of the seventh International Conference on Enterprise Information Systems (ICEIS 2005), INSTICC, pp. 163-168.

[22] Gago, P., Santos, M. F. , Silva, A., Cortez, P., Neves, J. and Gomes, L., *INTCare: A Knowledge Discovery based Intelligent Decision Support System for Intensive Care Medicine*, Journal of Decision Systems, Special issue on Design, Building and Evaluation of Intelligent DMSS, Lavoisier, Vol. 14 No. 3, 2005, pp. 241-259.

[23] Gago, P., Silva, A. and Santos, M. F., *Adaptive Decision Support for Intensive Care*, EPIA Workshops, Springer, 2007, pp. 415-425.

[24] Gago, P. and Santos, M. F., *Towards an Intelligent Decision Support System for Intensive Care Units*, ECAI Workshop on Supervised and Unsupervised Ensemble Methods and their Applications, Greece, 2008, pp. 21-25.

[25] Kolter, J. Z. and Maloof, M. A., *Dynamic Weighted Majority: A New Ensemble Method for Tracking Concept Drift*, Proceedings of the Third IEEE International Conference on Data Mining, IEEE Press, 2003, pp. 123-130.

[26] Barnato A. and Angus D., *Value and role of intensive care unit outcome prediction models in end-of-life decision making*. Crit Care Clin, Vol. 20, No. 3 , 2004, pp. 345-362.

[27] A. Lourenco and O. Belo, "Promoting agent-based knowledge discovery in medical intensive care units," WSEAS Transactions on Computers, vol. 2, pp. 403-408, 2003.
[28] A. Abelha, et al., "Health data management in the medical arena," WSEAS-Transactions-on-Computers, 2004.
[29] M. Pereira, et al., "Computer aided monitoring system of intensive care unit patients," WSEAS Transactions on Information Science and Applications, vol. 4, pp. 78-84, 2007.
[30] Thomas A. Lasko, Jui G. Bhagwat, Kelly H. Zou, and Lucila Ohno-Machado. The use of receiver operating characteristic curves in biomedical informatics. J. of Biomedical Informatics, 38(5):404–415, 2005.
[31] MH Zweig and G Campbell. Receiver-operating characteristic (roc) plots: a fundamental evaluation tool in clinical medicine. Clin Chem, 39(4):561–577, 1993.