EMADS: AN EXTENDIBLE MULTI-AGENT DATA MINER

Kamal Ali Albashiri, Frans Coenen, and Paul Leng

Abstract In this paper we describe EMADS, an Extendible Multi-Agent Data mining System. The EMADS vision is that of a community of data mining agents, contributed by many individuals, interacting under decentralised control to address data mining requests. EMADS is seen both as an end user application and a research tool. This paper details the EMADS vision, the associated conceptual framework and the current implementation. Although EMADS may be applied to many data mining tasks; the study described here, for the sake of brevity, concentrates on agent based data classification. A full description of EMADS is presented.

Keywords Multi-Agent Data Mining (MADM), Classifier Generation.

1 Introduction

Multi-Agent Systems (MAS) offer a number of general advantages with respect to Computer Supported Cooperative Working, distributed computation and resource sharing. Well documented advantages [1] include:

1. Decentralised control.
2. Robustness.
3. Simple extendability.
4. Sharing of expertise.
5. Sharing of resources.

Decentralised control is, arguably, the most significant feature of MAS that serves to distinguish such systems from distributed or parallel approaches to computation.

Kamal Ali Albashiri, Frans Coenen, and Paul Leng
Department of Computer Science, The University of Liverpool,
Ashton Building, Ashton Street, Liverpool L69 3BX, United Kingdom
e-mail: {ali,frans,phl}@csc.liv.ac.uk

Decentralised control implies that individual agents, within a MAS, operate in an autonomous manner and are (in some sense) self deterministic. Robustness, in turn is a feature of the decentralised control, where the overall system continues to operate even though a number of individual agents have "crashed". Decentralised control also supports extendability in that additional functionality can be added simply by including further agents. The advantages of sharing expertise and resources are self evident. The advantages offered by MAS are particularly applicable to Knowledge Discovery in Data (KDD) where a considerable collection of tools and techniques are current. MAS also has some particular advantages to offer with respect to KDD, and particularly data mining, in the context of sharing resources and expertise. KDD is concerned with the extraction of hidden knowledge from data. Very often data relevant to one search is not located at a single site, it maybe widely-distributed and in many different forms. There is a clear advantage to be gained from an organisation that can locate, evaluate and consolidate data from these diverse sources. KDD has evolved to become a well established technology that has many commercial applications. It encompasses sub-fields such as classification, clustering, and rule mining. Research work in these fields continues to develop ideas, generate new algorithms and modify/extend existing algorithms. A diverse body of work therefore exists. KDD research groups and commercial enterprises, are prepared (at least to some extent) to share their expertise. In addition, many KDD research groups have made software freely available for download[1]. This all serves to promote and enhance the current "state of the art" in KDD. However, although the free availability of data mining software is of a considerable benefit to the KDD community, it still require users to have some programming knowledge — this means that for many potential end users the use of such free software is not a viable option. One of the additional advantages offered by a MAS approach is that it would support greater end user access to data mining techniques.

A second advantages offered by MAS, in the context of data mining, is that of privacy and (to an extent) security. By its nature data mining is often applied to sensitive data. MAS allows data to be mined remotely. Similarly, with respect to data mining algorithms, MAS can make use of algorithms without necessitating their transfer to users, thus contributing to the preservation of intellectual property rights.

In this paper the authors propose the Extendible Multi-Agent Data mining System (EMADS). The EMADS vision is that of an anarchic collection of persistent, autonomous (but cooperating) KDD agents operating across the Internet. Individual agents have different functionality; the system currently comprises data agents, user agents, task agents, mining agents and a number of "house-keeping" agents. Users of EMADS may be data providers, data mining algorithm contributors or miners of data. The provision of data and mining software is facilitated by a system of *wrappers*. Users wishing to obtain (say) classifiers or collections of patterns, need have no knowledge of how any particular piece of data mining software works or the location of the data to be used. The operation of EMADS is illustrated in this paper

[1] See for example the Weka Tool Kit $http://www.cs.waikato.ac.nz/ml/weka/$, and the LUCS-KDD Software Library $http://www.csc.liv.ac.uk/\tilde{f}rans/KDD/Software/$

through the application of a collection of classifier data mining agents to a number of standard “benchmark” data sets held by data agents.

The paper is organised as follows. A brief review of some related work on Multi-Agent Data Mining (MADM) is presented in Section 2. The conceptual framework for EMADS is presented in Section 3. The current implementation of EMADS, together with an overview of the wrapper principle is given in 4. The operation of EMADS is illustrated in Section 5 with a classification scenario. Some conclusions are presented in Section 6.

2 Previous Work

There are a number of reports in the literature of the application of Agent techniques to data mining. Some example systems are briefly presented here. One of the earliest references to MADM is Kargupta et al. [2] who describe a parallel data mining system (PADMA) that uses software agents for local data accessing and analysis, and a Web based interface for interactive data visualisation. PADMA has been used in medical applications. Gorodetsky et al. [3] correctly consider that the core problem in MADM is not the data mining algorithms themselves (in many case these are well understood), but the most appropriate mechanisms to allow agents to collaborate. Gorodetsky et al. present a MADM system to achieve distributed data mining and, specifically, classification. They describe a distributed data mining architecture and a set of protocols for a multi-agent software tool. Peng et al. [4] present an interesting comparison between single-agent and multi-agent text classification in terms of a number of criteria including response time, quality of classification, and economic/privacy considerations. Their results indicate, not unexpectedly, in favour of a multi-agent approach.

Agent technology has also been employed in *meta-data mining*, the combination of results of individual mining agents. One example is meta classification, also sometimes referred to as meta-learning, this is a technique for generating a *global* classifier from *N* distributed data sources by first computing *N base* classifiers which are then collated to build a single *meta* classifier (see for example [14]). The meta-learning strategy offers a way to mine classifiers from homogeneously distributed data.

Perhaps the most mature agent-based meta-learning systems are: JAM [5], BODHI [6], and Papyrus [7]. In contrast to JAM and BODHI, Papyrus can not only move models from site to site, but can also move data when that strategy is desired. Papyrus is a specialised system which is designed for clustering while JAM and BODHI are designed for data classification. Basically, these systems try to combine local knowledge to optimise a global objective. The major criticism of such systems is that it is not always possible to obtain an exact final result, i.e. the global knowledge model obtained may be different from the one that might have been obtained by applying the one model approach to the same data.

It should be noted that the domains of distributed and multi-agent data mining tend to overlap, with much discussion amongst authors as to what a MADM system is. In this paper the authors concur with Wooldridge's [1] definition of what an agent is as itemised in Section 1.

3 The EMADS Conceptual Framework

Conceptually EMADS is a hybrid peer to peer agent based system comprising a collection of collaborating agents that exist in a set of *containers*. Agents may be created and contributed to EMADS by any EMADS user/contributor. One of these containers, the *main container*, holds a number of house keeping agents that have no direct connection with MADM, but provide various facilities to maintain the operation of EMADS. In particular the main container holds an Agent Management System (AMS) agent and a Directory Facilitator (DF) agent. The terminology used is taken from the JADE (Java Agent Development) [9] framework in which EMADS is implemented (JADE implementation details are discussed further in Section 4). Briefly the AMS agent is used to control the life cycles of other agents in the platform, and the DF agent provides an agent *lookup service*. Both the main container and the remaining containers can hold various MADM agents. Note that the EMADS main container is located on the EMADS host organisation site (currently The University of Liverpool in the UK), while the other containers may be held at any other sites world wide.

Other than the house keeping agents held in the main container EMADS currently supports four categories of MADM agents:

1. **User Agents**: User agents are the interface agents that connect users to EMADS. User agents allow users to pose requests and receive responses to such requests. Individual users create and launch their own EMADS user agents, which reside in the users' EMADS containers and are hosted at the users' site [2]. User agents interact with task agents (see below) in order to process data mining requests.
2. **Task Agents**: Task agents are specific temporary agents that are automatically created by user agents to address specific data mining requests. Task agents are located at the user's site and persist till the response to the associated requests is complete. A user can cause any number of task agents to be created. The nature of individual task agents depends on the nature of the requests, for example a classification task agent will be launched to respond to a classification request while (say) a meta Association Rule Mining task agent will be launched to respond to a meta-ARM request. Individual task agents posses meta-knowledge about data mining processes, which in turn define the methodology/approach best suited to respond to a particular data mining request; this includes input format requirements for specific data mining agents (see below). This meta-knowledge is used

[2] The EMADS user software is available from the EMADS mediator site at *http://www.jade.csc.liv.ac.uk/*, although currently EMADS is only available to local users

in initiate and execute a required data mining process. Task agents are also responsible for communication to/from data mining agents, and (if appropriate) the activation and synchronisation of data mining agents. To execute a data mining process a task agent typically seeks the services of a group of data mining and data agents (see below) to obtain the desired result and return it to the user agent.

3. **Mining Agents**: Mining agents are an implementation of a specific data mining technique or algorithm. Mining agents contain the methods for initiating and carrying out a data mining activity and communicating results back to the appropriate task agent. Note that to release the full potential of EMADS mining agents, in either the same or different containers, typically collaborate to resolve some data mining task; although they are not obliged to so. Data mining agents are contributed by any EMADS *developer*, and reside in their owner's EMADS container hosted at the owner's site.
4. **Data Agents**: An agent, located at a local site, that holds meta-data about specified data sources held at the same site. The data may be a single data set, part of a data set or a number of data sets. Data agents are provided by EMADS *data contributors*. One of the advantages offered by data agents is that of privacy preservation.

A high level view of the EMADS conceptualisation showing the various categories of agents and their interaction is given in Figure 1. The figure shows a mediator host (main container) and three local hosts (local containers). The mediator host holds a AMS and a DF agent. One of the local hosts has a user and a task agent, while the other two hosts hold data and mining agents.

It should be noted that EMADS containers may contained both mining and data agents simultaneously as well as user agents. It should also be noted that data mining and data agents are *persistent*, i.e. they continue to exist indefinitely and are not created for a specific data mining exercise. Communication between agents is facilitated by the EMADS network.

3.1 EMADS End User Categories

EMADS has several different modes of operation according to the nature of the *participant*. Each mode of operation (participant) has a corresponding category of user agent. Broadly, the supported categories are as follows:

- **EMADS Users**: These are participants, with restricted access to EMADS, who may pose data mining requests.
- **EMADS Data Contributors**: These are participants, again with restricted access, who are prepared to make data available to be used by EMADS mining agents.
- **EMADS Developers**: Developers are EMADS participants, who have full access and may contribute data mining algorithms.

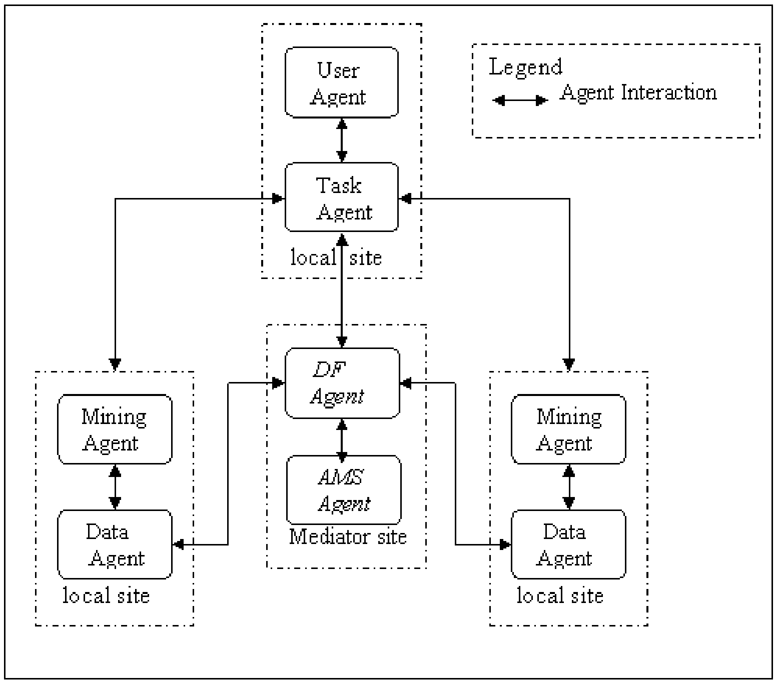

Fig. 1 High level view of EMADS conceptual framework.

Note that in each case, before interaction with EMADS can commence, appropriate software needs to be downloaded and launched by the participant. Note also that any individual participant may be a user as well as a contributor and/or developer.

Conceptually the nature of EMADS data mining requests, that may be posted by EMADS users, is extensive. In the current implementation, the following types of generic request are supported:

- Find the ”best” classifier (to be used by the requester at some later date in off line mode) for a data set provided by the user.
- Find the ”best” classifier for the indicated data set (i.e. provided by some other EMADS participant).
- Find a set of Association Rules (ARs) contained within the data set(s) provided by the user.
- Find a set of Association Rules (ARs) contained within the indicated type of data set(s) (i.e. provided by other EMADS participants).

A “best” classifier is defined as a classifier that will produce the highest accuracy on a given test set (identified by the mining agent) according to the detail of the request. To obtain the “best” classifier EMADS will attempt to access and communicate with as many classifier generator data mining agents as possible and select the best result. The classification style of user request will be discussed further in Section 5 to illustrate the operation of EMADS in more detail.

The Association Rule Mining (ARM) style of request is not discussed further in this paper. However,the idea here was that an agent framework could be used to implement a form of Meta-ARM where the results of the parallel application of ARM to a collection of data sets, with not necessarily the same schema but conforming to a global schema, are combined. Details of this process can be found in Albashiri et al. [8].

4 The EMADS Implementation

EMADS is implemented using the JADE framework. JADE is FIPA (Foundation for Intelligent Physical Agents) [10] compliant middleware that enables development of peer to peer applications based on the agent paradigm. JADE defines an agent platform that comprises a set of containers, which may be distributed across a network as in the case of EMADS. A JADE platform includes a main container in which is held a number of mandatory agent services. These include the AMS and DF agents whose functionality has already been described in Section 3. Recall that the AMS agent is used to control the lifecycles of other agents in the platform, while the DF agent provides a lookup service by means of which agents can find other agents. When a data mining or data agent is created, upon entry into the system, it announces itself to the DF agent after which it can be recognised and found by other agents.

Within JADE agents are identified by name and communicate using the FIPA Agent Communication Language (ACL). More specifically, agents communicate by formulating and sending individual messages to each other and can have “conversations” using interaction protocols that range from query request protocols to negotiation protocols. ACL message communication between agents within the same container uses event dispatching. Message communication between agents in the same JADE platform, but in different containers, is founded on RMI. Message communication between agents in different platforms uses the IIOP (Internet Inter-ORB Protocol). The latter is facilitated by a special Agent Communication Channel (ACC) agent also located in the JADE platform main containers.

Figure 2 gives an overview of the implementation of EMADS using JADE. The figure is divided into three parts: at the top are listed N user sites. In the middle is the JADE platform holding the main container and N other containers. At the bottom a sample collection of agents is included. The solid arrows indicates a “belongs to” (or “is held by”) relationship while the dotted arrows indicate a “communicates with” relationship. So the data agent at the bottom left belongs to $container$ 1 which in turn belongs to $User\ Site$ 1; and communicates with the *AMS agent* and (in this example) a single *mining agent*.

The principal advantage of this JADE architecture is that it does not overload a single host machine, but distributes the processing load among multiple machines. The results obtained can be correlated with one another in order to achieve computationally efficient analysis at a distributed global level.

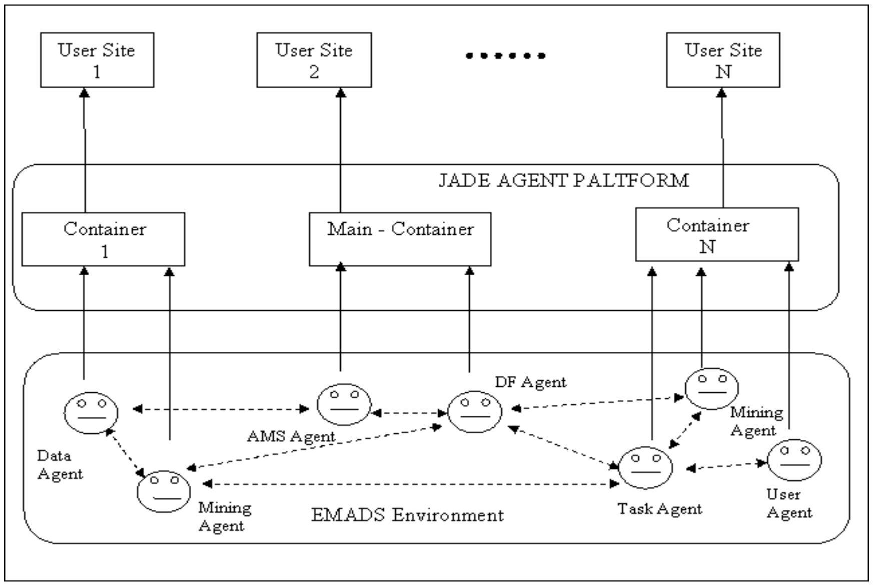

Fig. 2 EMADS Architecture as Implemented in Jade

4.1 EMADS Wrappers

One of the principal objectives of EMADS is to provide an easily extendible framework that could easily accept new data sources and new data mining techniques. In general, extendibility can be defined as the ease with which software can be modified to adapt to new requirements or changes in existing requirements. Adding a new data source or data mining techniques should be as easy as adding new agents to the system. The desired extendability is achieved by a system of wrappers. EMADS wrappers are used to "wrap" up data mining artifacts so that they become EMADS agents and can communicate with other EMADS agents. As such EMADS wrappers can be viewed as agents in their own right that are subsumed once that have been integrated with data or tools to become data mining agents. The wrappers essentially provide an application interface to EMADS that has to be implemented by the end user, although this has been designed to be a fairly trivial operation. Two broad categories of wrapper have been defined: (i) data wrappers and (ii) tool wrappers. Each is described in further detail in the following two sections.

4.1.1 Data Wrappers

Data wrappers are used to "wrap" a data source and consequently create a data agent. Broadly a data wrapper holds the location (file path) of a data source, so that it can be accessed by other agents; and meta information about the data. To assist end users in the application of data wrappers a data wrapper GUI is available. Once created, the data agent announces itself to the DF agent as consequence of which it becomes available to all EMADS users.

4.1.2 Tool Wrappers

Tool wrappers are used to “wrap” up data mining software systems and thus create a mining agent. Generally the software systems will be data mining tools of various kinds (classifiers, clusters, association rule miners, etc.) although they could also be (say) data normalisation/discretisation or visualisation tools. It is intended that EMADS will incorporate a substantial number of different tool wrappers each defined by the nature of the desired I/O which in turn will be informed by the nature of the generic data mining tasks that it us desirable for EMADS to be able to perform. Currently the research team have implemented two tool wrappers:

1. The binary valued data, single label, classifier generator.
2. The meta AR generator.

Many more categories of tool wrapper can be envisaged. Mining tool wrappers are more complex than data wrappers because of the different kinds of information that needs to be exchanged. For example in the case of a “binary valued, single label, classifier generator” wrapper the input is a binary valued data set together with meta information about the number of classes and a number slots to allow for the (optional) inclusion of threshold values. The output is then a classifier expressed as a set of Classification Rules (CRs). As with data agents, once created, the data mining agent announce themselves to the DF agent after which they will becomes available for use to EMADS users.

5 EMADS Operation: Classifier Generation

In this section the operation of EMADS is illustrated in the context of a classifier generation task; however much of the discussion is equally applicable to other generic data mining tasks such as clustering and ARM. The scenario is that of an end user who wishes to obtain a ”best” classifier founded on a given, pre-labelled, data set; which can then be applied to further unlabelled data. The assumption is that the given data set is binary valued and that the user requires a single-label, as opposed to a multi-labelled, classifier. The request is made using the individual’s user agent which in turn will spawn an appropriate task agent.

For this scenario the task agent identifies mining agents that hold single labelled classifier generators that take binary valued data as input. Each of these mining agents is then accessed and a classifier, together with an accuracy estimate, requested. The task agent then selects the classifier with the best accuracy and returns this to the user agent.

The data mining agent wrapper in this case provides the interface that allows input for: (i) the data; and (ii) the number of class attributes (a value that the mining agent cannot currently deduce for itself) while the user agent interface allows input for threshold values (such as support and confidence values). The output is a classifier together with an accuracy measure. To obtain the accuracy measures the classifier generator (data mining agent) builds the classifier using the first half of the input data as the "training" set and the second half of the data as the "test" set. An alternative approach might have been to use Ten Cross Validation (TCV) to identify the best accuracy.

From the literature there are many reported techniques available for generating classifiers. For the scenario the authors used implementations of eight different algorithms [3]:

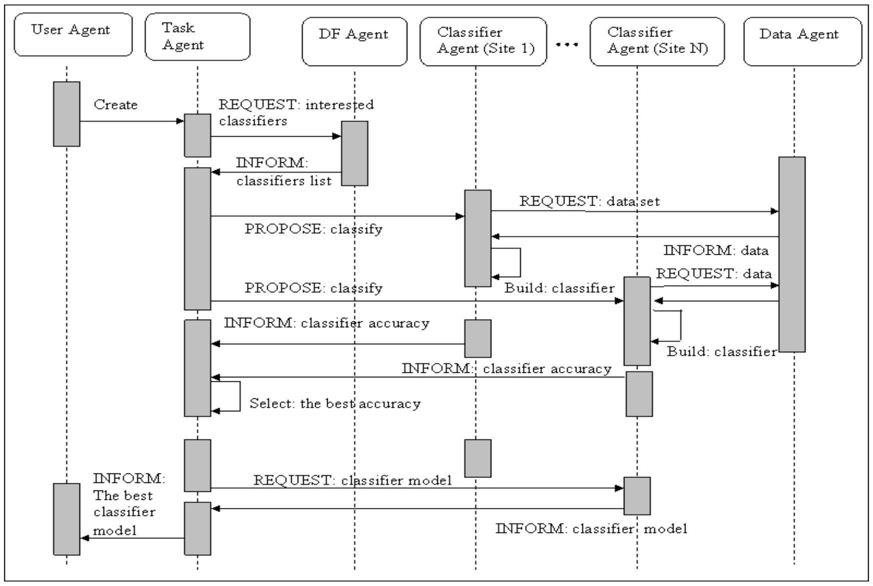
User Agent
Task Agent
DF Agent
Classifier Agent (Site 1)
Classifier Agent (Site N)
Data Agent
Create
REQUEST: interested classifiers
INFORM: classifiers list
REQUEST: data set
PROPOSE: classify
INFORM: data
REQUEST: data
PROPOSE: classify
Build: classifier
INFORM: classifier accuracy
Build: classifier
INFORM: classifier accuracy
Select: the best accuracy
INFORM: The best classifier model
REQUEST: classifier model
INFORM: classifier model

Fig. 3 Classification Task Sequence Diagram.

1. FOIL (First Order Inductive Learner) [11] the well established inductive learning algorithm for generating Classification Association Rules (CARs).
2. TFPC (Total From Partial Classification) CAR generator [12] founded on the P- and T-tree set enumeration tree data structures.
3. PRM (Predictive Rule Mining) [15] an extension of FOIL.
4. CPAR (Classification based on Predictive Association Rules) [15] a further development from FOIL and PRM.

[3] taken from the LUCS-KDD repository at *http://www.csc.liv.ac.uk/~frans/KDD/Software/*

5. IGDT (Information Gain Decision Tree) classifier, an implementation of the well established decision tree based classifier using most information gain as the “splitting criteria”.
6. RDT (Random Decision Tree) classifier, a decision tree based classifier that uses most frequent current attribute as the “splitting criteria” (so not really random).
7. CMAR (Classification based on Multiple Association Rules) is a Classification Association Rule Mining (CARM) algorithm [16] .
8. CBA (Classification Based on Associations) is a CARM algorithm [17].

These were placed within an appropriately defined tool wrapper to produce eight (single label binary data classifier generator) data mining agents. This was a trivial operation indicating the versatility of the wrapper concept.

Thus each mining agent’s basic function is to generate a classification model using its own classifier and provide this to the task agent. The task agent then evaluates all the classifier models and chooses the most accurate model to be returned to the user agent. The negotiation process amongst the agents is represented by the sequence diagram given in Figure 3 (the figure assumes that an appropriate data agent has ready been created). In the figure includes *N* classification agents. The sequence of events commences with a user agent which spawns a (classification) task agent, which in turn announces itself to the DF agent. The DF agent returns a list of classifier data mining agents that can potentially be used to generate the desired classifier. The task agent then contacts these data mining agents who each generate a classifier and return statistical information regarding the accuracy of their classifier. The task agent selects the data mining agent that has produced the best accuracy and requests the associated classifier, this is then passed back to the user agent.

Table 1 Classification Results

Data Set	Classifier	Accuracy	Generation Time (sec)
connect4.D129.N67557.C3	RDT	79.76	502.65
adult.D97.N48842.C2	IGDT	86.05	86.17
letRecog.D106.N20000.C26	RDT	91.79	31.52
anneal.D73.N898.C6	FOIL	98.44	5.82
breast.D20.N699.C2	IGDT	93.98	1.28
congres.D34.N435.C2	RDT	100	3.69
cylBands.D124.N540.C2	RDT	97.78	41.9
dematology.D49.N366.C6	RDT	96.17	11.28
heart.D52.N303.C5	RDT	96.02	3.04
auto.D137.N205.C7	IGDT	76.47	12.17
penDigits.D89.N10992.C10	RDT	99.18	13.77
soybean-large.D118.N683.C19	RDT	98.83	13.22
waveform.D101.N5000.C3	RDT	96.81	11.97

Note that the users make the data that they desire to be mined (classified) available by launching their own data agents (which in turn publish their name and description using the DF agent as described above). The data sets used for the illustration were taken from the UCI machine learning data repository [18]. To simplify the scenario these data sets were preprocessed so that they were discretized/normalized into a binary form [4]. It should be noted here that the research team is currently implementing a normalisation/discretisation agent.

The results from a sequence of user requests, using different data sets, are presented in Table 1. Each row in the table represents a particular request and gives the name of the data set, the selected best algorithm, the best accuracy and the total EMADS execution time from creation of the initial task agent to the final classifier being returned to the user agent. The naming convention used in the Table is that: *D* equals the number of attributes (after discretisation/normalisation), *N* the number of records and *C* the number of classes (although EMADS has no requirement for the adoption of this convention).

The results demonstrate firstly that EMADS works (at least in the context of the current scenario). Secondly that operation of EMADS is not significantly hindered by agent communication overheads, although this has some effect. The results also reinforce the often observed phenomena that there is no single best classifier generator suited to all kinds of data set.

6 Conclusions and Future Work

This paper describes EMADS, a multi-agent framework for data mining. The principal advantages offered are that of experience and resource sharing, flexibility and extendibility, and (to an extent) protection of privacy and intellectual property rights. The paper presents the EMADS vision, the associated conceptualisation and the JADE implementation. Of note are the way that wrappers are used incorporate existing software into EMADS. Experience indicates that, given an appropriate wrapper, existing data mining software can be very easily packaged to become an EMADS data mining agent. The EMADS operation is illustrated using a classification scenario.

A good foundation has been established for both data mining research and genuine application based data mining. The current functionality of EMADS is limited to classification and Meta-ARM. The research team is at present working towards increasing the diversity of mining tasks that EMADS can address. There are many directions in which the work can (and is being) taken forward. One interesting direction is to build on the wealth of distributed data mining research that is currently available and progress this in an MAS context. The research team are also enhancing the system's robustness so as to make it publicly available. It is hoped that once the system is live other interested data mining practitioners will be prepared to contribute algorithms and data.

[4] The discretized data sets are available at $http://www.csc.liv.ac.uk/\tilde{}frans/KDD/Software/LUCS-KDD-DN/DataSets/dataSets.html$

References

1. Wooldridge, M. (2003). *An Introduction to Multi-Agent Systems*. John Wiley and Sons (Chichester, England).
2. Kargupta, H., Hamzaoglu, I. and Stafford B. (1997). *Scalable, Distributed Data Mining Using an Agent Based Architecture.* Proceedings of Knowledge Discovery and Data Mining, AAAI Press, 211-214.
3. Gorodetsky, V., Karsaeyv, O., Samoilov, V. (2003). *Multi-agent technology for distributed data mining and classification.* Proc. Int. Conf. on Intelligent Agent Technology (IAT 2003), IEEE/WIC, pp438-441.
4. Peng, S., Mukhopadhyay, S., Raje, R., Palakal, M. and Mostafa, J. (2001). *A Comparison Between Single-agent and Multi-agent Classification of Documents.* Proc. 15th International Parallel and Distributed Processing Symposium, pp935-944.
5. Stolfo, S., Prodromidis, A. L., Tselepis, S. and Lee, W. (1997). *JAM: Java Agents for Meta-Learning over Distributed Databases.* Proceedings of the International Conference on Knowledge Discovery and Data Mining, pp. 74-81.
6. Kargupta, H., Byung-Hoon, et al. (1999). *Collective Data Mining: A New Perspective Toward Distributed Data Mining.* Advances in Distributed and Parallel Knowledge Discovery, MIT/AAAI Press.
7. Bailey, S., Grossman, R., Sivakumar, H. and Turinsky, A. (1999). *Papyrus: a system for data mining over local and wide area clusters and super-clusters.* In Proc. Conference on Supercomputing, page 63. ACM Press.
8. Albashiri, K.A., Coenen, F.P., Sanderson, R. and Leng. P. (2007). *Frequent Set Meta Mining: Towards Multi-Agent Data Mining.* In Bramer, M., Coenen, F.P. and Petridis, M. (Eds.), Research and Development in Intelligent Systems XXIII., Springer, London, (proc. AI'2007), pp139-151.
9. Bellifemine, F. Poggi, A. and Rimassi, G. (1999).*JADE: A FIPA-Compliant agent framework.* Proc. Practical Applications of Intelligent Agents and Multi-Agents, pg 97-108 (See http://sharon.cselt.it/projects/jade for latest information).
10. Foundation for Intelligent Physical Agents, FIPA 2002 Specification. Geneva, Switzerland.(See http://www.fipa.org/specifications/index.html).
11. Quinlan, J. R. and Cameron-Jones, R. M. (1993). *FOIL: A Midterm Report.* Proc. ECML, Vienna, Austria, pp3-20.
12. Coenen, F., Leng, P. and Zhang, L. (2005). *Threshold Tuning for Improved Classification Association Rule Mining.* Proceeding PAKDD 2005, LNAI3158, Springer, pp216-225.
13. Schollmeier, R. (2001). *A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications.* First International Conference on Peer-to-Peer Computing (P2P01) IEEE.
14. Prodromides, A., Chan, P. and Stolfo, S. (2000). *Meta-Learning in Distributed Data Mining Systems: Issues and Approaches.* In Kargupta, H. and Chan, P. (Eds), Advances in Distributed and Parallel Knowledge Discovery. AAAI Press/The MIT Press, pp81-114.
15. Yin, X. and Han, J. (2003). CPAR: Classification based on Predictive Association Rules. Proc. SIAM Int. Conf. on Data Mining (SDM'03), San Fransisco, CA, pp. 331-335.
16. Li W., Han, J. and Pei, J. (2001). CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. Proc ICDM 2001, pp369-376.
17. Liu, B. Hsu, W. and Ma, Y (1998). Integrating Classification and Assocoiation Rule Mining. Proceedings KDD-98, New York, 27-31 August. AAAI. pp80-86.
18. Blake, C.L. and Merz, C.J. (1998). *UCI Repository of machine learning databases* $http://www.ics.uci.edu/\tilde{m}learn/MLRepository.html$, Irvine, CA: University of California, Department of Information and Computer Science.