Introduction to Multi-Agent Systems
Laboratory 1

Victor Gimenez Abalos - victor.gimenez.abalos@est.fib.upc.edu
Daniel Felipe Ordonez Apraez - daniel.ordonez@est.fib.upc.edu
Albert Rial Farràs - albert.rial@est.fib.upc.edu
Jordi Armengol Estapé - jordi.armengol.estape@est.fib.upc.edu
Joan Llop Palao - joan.llop@est.fib.upc.edu

October 22, 2019

Abstract

In this activity, we have to study the theoretical concepts related to the characteristics of the environment, the kind of architecture of the agents and their types and properties in order to design an agent-based decision support system (A-DSS).

Contents

1 Introduction **4**

2 Environment Characteristics **4**

2.1 Accessible vs inaccessible environment 4

2.2 Deterministic vs non-deterministic environment 4

2.3 Episodic vs non-episodic environments 4

2.4 Static vs dynamic environment 5

2.5 Discrete vs continuous environment 5

3 System and Agents Architectures **5**

3.1 User agent 5

3.2 Manager agent 6

3.3 Classifier agents 6

4 Agents Properties **7**

4.1 User agent 8

4.2 Manager agent 8

4.3 Classifier agent 9

5 Agents types **10**

5.1 User agent 10

5.2 Manager agent 10

5.3 Classifier agent 10

References **10**

A Appendices **10**

A.1 Task distribution 10

A.2 Project meetings 11

1 Introduction

The proposed A-DSS system includes three different agents:

- **User agent**
  This agent operates as the main interface with the system user, by providing the capability of executing actions on command, e.g. the system user can request a *training* or *prediction* action that will affect the classifier agents.
- **Manager agent**
  The main task of this agent is to control and coordinate the operation of the classifier agents in order to fulfill the commands of the system user. In the case of a *training action* command the manager should coordinate the training of all classifier agents with its correspondent hyper-parameters and depending on the resultant accuracy of each of them decide on a metric to agglomerate/combine the multiple output, once a *prediction action* command is received.
- **Classifier agents**
  The classifier agents are encapsulations of supervised machine-learning models which have the capability to be trained on an specific data-set and predict with a certain accuracy the nature of unseen data.

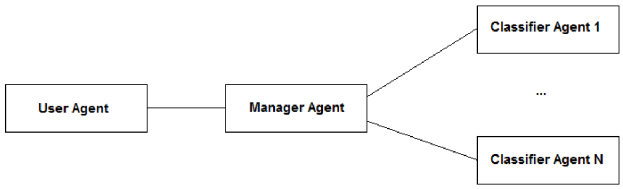

Figure 1: System architecture displaying the interaction between the proposed agents

2 Environment Characteristics

2.1 Accessible vs inaccessible environment

In an accessible environment the agents can obtain complete, accurate and updated information about the environment's state. In our system, as there is no need to avoid giving all the information to each agent, we believe that the environment we will be working in is accessible. Even though we could consider the environment inaccessible for the classifier agents, since they only receive partial information about the data they classify, we will consider that they can access to all the information and the manager tells them what parts they need to use.

2.2 Deterministic vs non-deterministic environment

A deterministic environment is one in which any action has a single guaranteed effect and there is no uncertainty about the resulting state that will result from performing an action. The non-deterministic and deterministic nature of the environment is dependent on the application domain, i.e. which data-set we want to learn and predict, and the models used for this endeavor (since this can be deterministic and non-deterministic models). For now, we will consider that our models will be deterministic, and so is our environment.

2.3 Episodic vs non-episodic environments

Our environment can be regarded as episodic. That means the performance of every agent is independent in each episode. Considering a simple implementation of the prediction task the system should not need to consider previous executions. Having this in mind, a more complex approach, the system could consider its performance on previous episodes to change its behaviour, e.g. diversification of the data, or different aggregation mechanisms, etc. This, however, is not necessary for the simple approach we have chosen.

2.4 Static vs dynamic environment

A dynamic environment is one that has other processes operating on it. For that reason, we consider that the environment of our system is dynamic taking into account that we have human interaction with the system at any moment of time.

2.5 Discrete vs continuous environment

We consider our environment to be discrete. That means that we have a fixed and finite number of actions in the system. The only action that could be considered 'unlimited' would be the manager giving samples to the classifiers, and even then, given the finite nature of the datasets, there is a finite number of combinations that can be performed. The dynamism of the environment does not affect the number of percepts in this case.

3 System and Agents Architectures

In this section we describe the type of architecture we think it is better for each kind of agent in our system. The main kinds of agent architectures we have taken into account are:

- **Reactive architectures:** These architectures are focused on fast reactions/responses to changes detected in the environment.
- **Deliberative architectures:** They are focused on long-term planning of actions, centred on a set of basic goals.
- **Hybrid architectures:** As the name suggests, these architectures combine a reactive side and a deliberative one.

3.1 User agent

The role of the user agent is to send either training data or inference queries to the manager and present the responses of the manager to the end-user of the system. The user will exhibit a reactive architecture. Reactive architectures focus on fast responses to changes in the environment. Since this agent is essentially in charge of sending and receiving information on demand, without any sophisticated rule, we believe the **reactive** architecture to be the most adequate in this case.

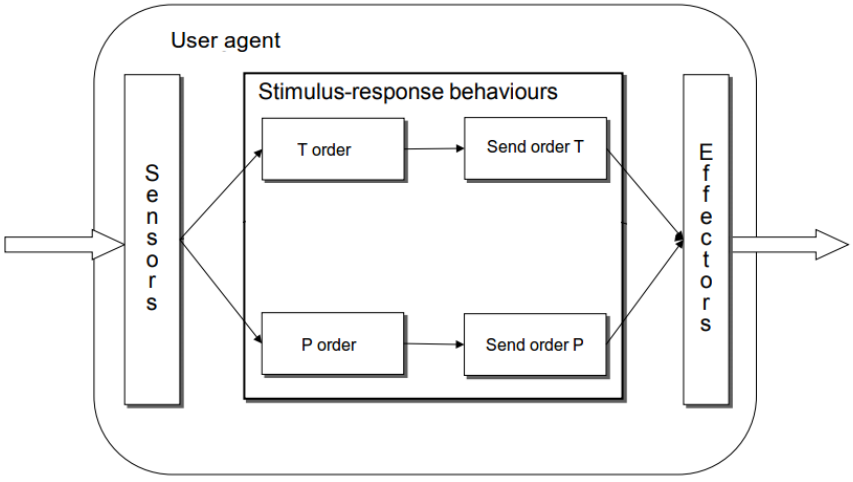

Figure 2: User agent architecture (reactive)

3.2 Manager agent

The manager is the agent responsible for transmitting the user information to each classifier, to receive the results and combine them to return the final answer to the user. The manager will be based on a **hybrid** architecture since we identify both reactive and deliberative functionalities:

- The behavioral layer of the agent will be in charge of transmitting the data to each classifier. It will consist of simple rules and there will not be any representation of the world.
- The cooperative planning layer of the agent will have a representation of the classifier agents. In inference time, this layer will solve conflicts between different classifiers, if they express different beliefs on a particular instance (for example, using a reputation system).

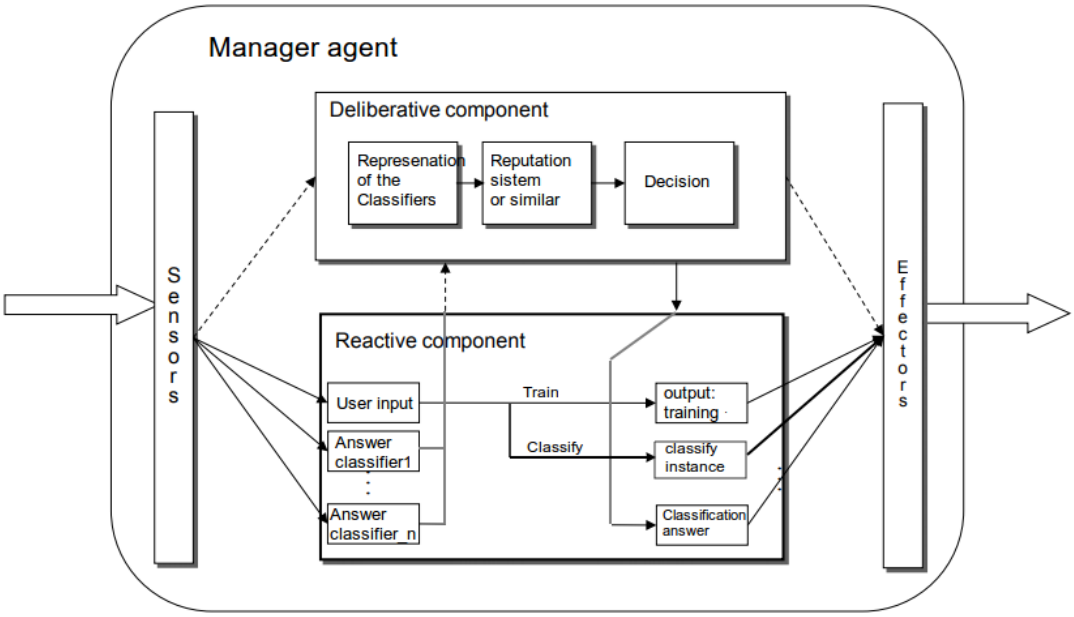

Figure 3: Manager agent architecture (hybrid)

3.3 Classifier agents

Classifier agents are in charge of learning a model to classify the instances of the dataset into different categories. Once the model is learned, it is used to infer the class of new instances. The classifier will be **hybrid** since we identify both reactive and deliberative aspects (figure 4):

- The behavioral layer will act in inference time since there will be a simple action-reaction schema. Each time the agent receives a query, it will reply with the corresponding prediction, without changing the state of the world stored in the agent.
- The planning layer will be activated during training. Learning a pattern recognition model is a non-trivial task that involves developing a certain internal representation of the world, which seems to be deliberative.

Another possibility that we have considered is that the classifier agent could be reactive. The training process could be seen as an action-reaction process, where the input consists of data and the action is to train the model or to predict the class of the input data. However, we have finally decided to consider the classifiers as hybrid agents.

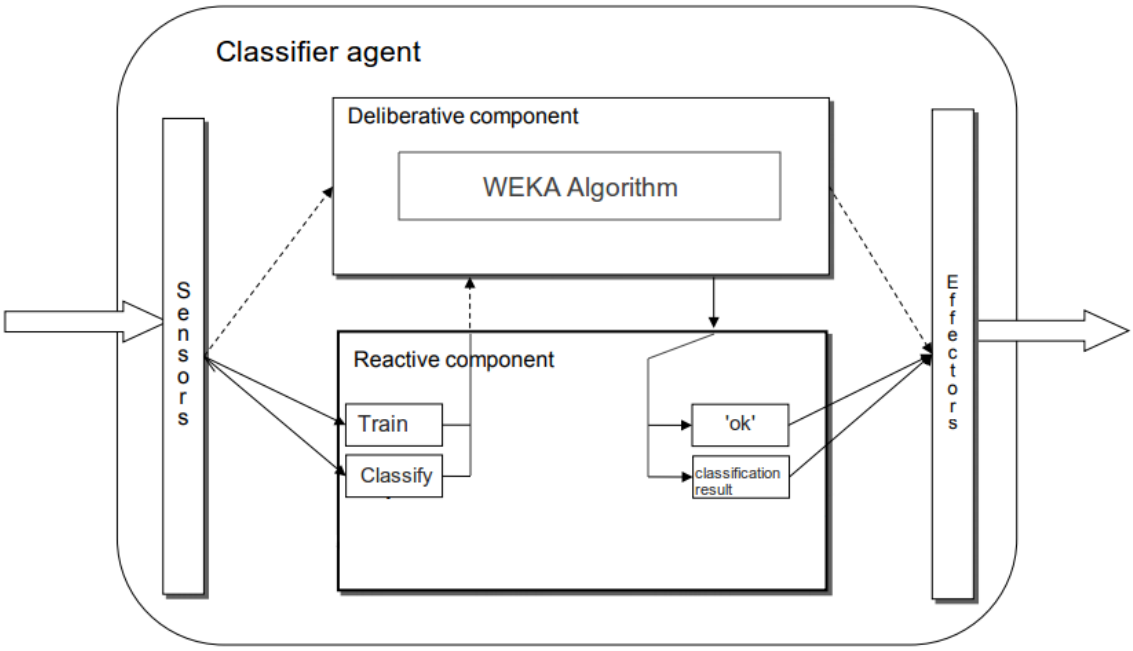

Figure 4: Classifier agent architecture (hybrid)

4 Agents Properties

Finally, in this last section we detail the properties that should be exhibited by each type of agents. The properties that we consider are:

- **Reactivity:** A reactive system is one that maintains a continuous interaction with the environment and responds to changes that occur in it.
- **Proactiveness:** Being a proactive agent means being able to take the initiative when appropriate and behave not driven solely by events.
- **Communication and social ability:** Ability to interact with other agents or humans, and cooperate with them.
- **Flexibility:** To be flexible, an agent must be reactive, proactive and social.
- **Rationality:** An agent with this property will act in order to achieve its goals.
- **Reasoning:** Is the capacity of an agent to make plans and to infer and extrapolate based on its knowledge and experience.
- **Learning:** If an agent has this property means that is able to improve its performance over time.
- **Autonomy:** It is the ability to achieve goals in an autonomous way, without interacting continuously with the user.
- **Mobility:** Having this property means to be able to move physically through a network and execute in different computers.
- **Temporal continuity:** An agent with temporal continuity is one that is continuously running processes.
- **Veracity:** The information communicated by the agent will be truthful, or at least the agent does not know that is false.

4.1 User agent

Considering the reactive architecture of this agent its properties are:

- **Reactivity:**
  This agent is in fact reactive since it responds to user action commands, by initiating the communication process with the manager agent.
- **Communication and social ability:**
  This agent not only interacts with the human user but also with the manager agent, requiring the capability to broadcast and receive commands and data.
- **Rationality:**
  Given the simple nature of this system as a rule-based interface between the human user and the rest of the system, rationality is guaranteed.
- **Temporal Continuity:**
  The agent is required to stay on a passive/sleeping state until a human interaction is started, and then it has to perform the necessary action to broadcast this command to the manager agent.

On the contrary the agent does not present the behaviour of:

- **Proactiveness:**
  The agent reacts only to stimulus from human agent request.
- **Flexibility:**
  Since there is no proactiveness in the agent operation.
- **Reasoning and Learning:**
  This agent only responds to user action commands and communicate with it and other agents. It does not have any logic, so it is not able to make plans nor improve over time.
- **Autonomy:**
  Since the agent is entirely reactive, and follows the commands of the human user.
- **Mobility:**
  Since the entire system will run locally.
- **Veracity:**
  By definition the agent will communicate hypothesis based on the response from non-perfect learning models that try to approximate the behaviour of real-world data, nonetheless, there will be room for error and the agent cannot guarantee the complete veracity of the information.

4.2 Manager agent

Considering the hybrid architecture of this agent its properties are:

- **Reactivity:**
  This agent is reactive since it responds to commands/percepts provided by the user agent, in order to initiate the training and prediction actions.
- **Proactiveness and Autonomy:**
  This agent is proactive and autonomous since it decides the ensemble methodology used to agglomerate the models responses in order to achieve the optimum prediction accuracy.
- **Communication and social ability:**
  This agent interacts with the user agent by receiving and answering to training and prediction queries, and also with the classifiers agents by coordinating their functionality and agglomeration of their responses.
- **Flexibility:**
  Flexibility is guaranteed by the social ability of this agent, its reactivity and its pro-activity.

- **Rationality:**
  This agent is rational since it will coordinate the training and prediction queries for each of the classifier agents in order to complete the requests/actions provided by the user agent.
- **Reasoning and Learning:**
  If the agent uses some methodology of ensemble learning this properties would be present in the agent.
- **Temporal Continuity:**
  The agent remains passive until a request for a training or prediction action is received and then maintains active until these are fulfilled by combining the responses from all classifier agents.

On the contrary the agent does not present the behaviour of:

- **Mobility:**
  Since the entire system will run locally.
- **Veracity:**
  By definition the agent will work based on non-perfect models that try to approximate the behaviour of real-world data, nonetheless, there will be room for error and the agent cannot warranty the complete veracity of the information.

4.3 Classifier agent

- **Reactivity:**
  This agent is reactive since it answers to the commands/percepts from the manager agent, which determine the nature of the classifier model the agent will use, the data used for training, and the queries used for prediction.
- **Communication and social ability:**
  This agent is social since it communicates and interacts with the manager agent.
- **Rationality:**
  The agent is rational since given the model type, training data-set and training hyper-parameters the classifier agent always tries to answer the prediction queries.
- **Reasoning and Learning:**
  The agent goal is to learn from the environment through a given model, and use it to reason to predict the environment behavior.
- **Temporal Continuity:**
  The agent remains passive until a request for a training or prediction action is received and then maintains active until these are fulfilled.

On the contrary the agent does not present the behaviour of:

- **Proactiveness nor autonomy:**
  The agent reacts only to control percepts from the manager agent.
- **Flexibility:**
  Since there is no proactiveness in the agent operation.
- **Autonomy:**
  Since the agent is entirely reactive, and follow the commands of manager agent.
- **Mobility:**
  Since the entire system will run locally.
- **Veracity:**
  By definition the agent will work based on non-perfect models that try to approximate the behaviour of real-world data, nonetheless, there will be room for error and the agent cannot warranty the complete veracity of the information.

5 Agents types

5.1 User agent

This agent will be an **Interface agent**.

The user agent will be an agent capable of helping the human user with the interaction with the manager. It should avoid repetitive tasks to the human by minimizing the human input to the system. Since its functionality is simple, the cooperation with other agents will be limited (simple orders and simple responses). For now, there is no need for the user agent to learn to improve the user experience (for example, by auto-completing orders), but at any point we could implement this kind of functionality if we observe that is necessary.

5.2 Manager agent

The manager fits perfectly under the type of **Collaborative agent**:

- The manager is the agent in charge of **communication** to both agents (user agent and classifiers): it communicates with the user agent to get the orders and to send the answers to the classification problem. It also talks to the classifiers to the instances to be classified and to get the answers.
- The manager is the agent that has to resolve the conflict between classifiers, to do so, it will need some kind of **negotiation** process.
- In case that we implement a reputation system, the manager will need to **learn** to update the reputation of each classifier (we are not sure that this will be the best way to proceed). In any case the learning process will be **limited**.

5.3 Classifier agent

This types of agents will be standard machine learning applications translated to an agent. Therefore, they are examples of **agentification**. Since we will be working with weka algorithms, our best option is to make our agent a **translator** as far as calls to weka are concerned. This will simplify the code since we will not need to modify the algorithm only work with the inputs or outputs of the application.

References

[1] Russell, S. J., & Norvig, P. (2016). Artificial intelligence: a modern approach. Malaysia; Pearson Education Limited,.

[2] Wikibooks contributors. (2019, 11 January). Artificial Intelligence: Agents and their Environments. In Wikibooks. Retrieved 22:35, October 16, 2019, from `https://en.wikibooks.org/wiki/Artificial_Intelligence/AI_Agents_and_their_Environments`

[3] Nowrouzi, Ehsan (2012). Artificial Intelligence Chapter Two: Agents Slides. In Slideshare. Retrieved 16:54, October 16, 2019, from `https://es.slideshare.net/EhsanNowrouzi/artificial-intelligence-chapter-two-agents`

A Appendices

A.1 Task distribution

The task distribution can be visualized on the repository project created for this purpose, which can be found here: https://github.com/jordiae/IMAS-MAI/projects/3

The initial draft of this report was developed on the repository wiki page and a version controlled report of its changes can be found here: https://github.com/jordiae/IMAS-MAI/wiki/environ/_history

A.2 Project meetings

A record of the team meetings can be found in the repository wiki page:
https://github.com/jordiae/IMAS-MAI/wiki