Scientific Research Publishing

World Journal of Engineering and Technology, 2023, 11, 682-697
https://www.scirp.org/journal/wjet
ISSN Online: 2331-4249
ISSN Print: 2331-4222

A Generic Intelligent Agent Design Approach Based on Artificial Neural Networks

Thierry Noulamo[1], Alain Djimeli-Tsajio[1], Roger Kameugne[2], Jean-Pierre Lienou[3]

[1]UIT Fotso Victor of Bandjoun, University of Dschang, Dschang, Cameroon
[2]Faculty of Sciences, University of Maroua, Maroua, Cameroon
[3]College of Technology, University of Bamenda, Bamenda, Cameroon
Email: thierry.noulamo@gmail.com, adjimeli@gmail.com, rkameugne@gmail.com, lienou@gmail.com

How to cite this paper: Noulamo, T., Djimeli-Tsajio, A., Kameugne, R. and Lienou, J.-P. (2023) A Generic Intelligent Agent Design Approach Based on Artificial Neural Networks. *World Journal of Engineering and Technology*, **11**, 682-697.
https://doi.org/10.4236/wjet.2023.114046

Received: August 8, 2023
Accepted: October 4, 2023
Published: October 7, 2023

Abstract

Artificial intelligence in general and software agents in particular are recognized as computer science disciplines that aim to model or simulate so-called intelligent human behaviors such as perception, decision-making, understanding, learning, etc. This work presents an approach to designing a generic Intelligent Agent that can be used in a multi-agent system to solve a complex problem. The generic agent that is proposed can be instantiated as a concrete agent, which is enabled with learning and autonomy capabilities by using Artificial Neural Networks. To highlight the generic aspect, the proposition is instantiated to be used in agriculture, health and education. The instantiated software agent applied in agriculture can process images in real time and detect defect on plants' leaf. In the health field, the agent process image to diagnose breast cancer. When applied in Education, the agent can load an image of a student's script and grade it. The performance of the designed agent system has the same accuracy as that of the respective neural networks used to instantiate them. In the educational field, the software agent has an accuracy of 98.9% and in the health field, it has an accuracy of 99.56% while in the agricultural field, it has an accuracy of 97.2%.

Keywords

Artificial intelligence, Abstract Agents design, Formal Neurons Interconnection, Multi-Agent System

1. Introduction

The growth in size and heterogeneity of computer systems, observed for more than twenty years, has led computer science research to take an interest in and develop systems composed of several entities among which are distributed com-

plex tasks to be performed. This system design approach, called the distributed approach, has developed in different branches of computer science. Thus, from Artificial Intelligence (AI) was born Distributed Artificial Intelligence (lAD) which itself generated Multi-Agent Systems (MAS). To successfully improve their information system and succeed in their process automation project, companies call on IT engineering service companies (SSII). In order to help their customers automate their process by detecting the anomalies present there and identifying opportunities to improve their productivity, IT services companies need an approach and a tool that makes it possible to represent the different types of requirements required.

Business and production process automation is an alternative to IT engineering service development. Indeed, this method allows business experts to quickly and iteratively define and execute the business logic they need. In this case, the business logic translates into sets of service, rules, or workflows, rather than application functions. These rules and processes are then executed by an engine in response to certain conditions or events [1] [2] [3]. This engine is in most cases specific to a business domain [4] [5] [6]. The question that emerges here is how to produce a generic engine independent of the business domains. One solution is the development of an adaptive intelligent agent, capable of enriching its knowledge following a learning process using Machine Learning (ML).

ML is a sub-field of Artificial Intelligence that focuses on the development of models capable of representing certain characteristics of the world around us, of learning certain statistical properties of the distributions of the data they process, in order to accomplish various tasks. The relationship to intelligence comes from the ability of these models to generalize, *i.e.* to extract relevant information from data studied over the course of an updating process called training, and to know how to reuse it effectively on new data never encountered before.

A Multi-Agent System (MAS) is composed of several autonomous subsystems called agents, each of which has its own activity and information. The general behavior of the MAS is then linked to the combined activity of all its agents and the performance of a task may involve several officers. This distribution requires that each agent can locally perform the task (or sub-task) assigned to it, but also that it can coordinate with other agents if it is necessary to manage dependencies between the sub-tasks or if it needs functions provided by other agents. An agent is a software component, a computer module or an autonomous virtual entity which is able to act in an environment, to communicate directly with other agents, to make decisions and has a behavior which tends to satisfy its objectives, taking into account the resources and skills at its disposal, and according to his perception, his representations and the communications it receives. There are various classifications for agents, we have chosen to present the typology established in [7] which relates to the properties of agents (see **Figure 1**). In this work, the smart agents area of [7] is used. For the support of IT engineering service companies, intelligent agents seem more appropriate to us, given their strong adaptability. A decade ago, efforts were made to improve Multi Agent

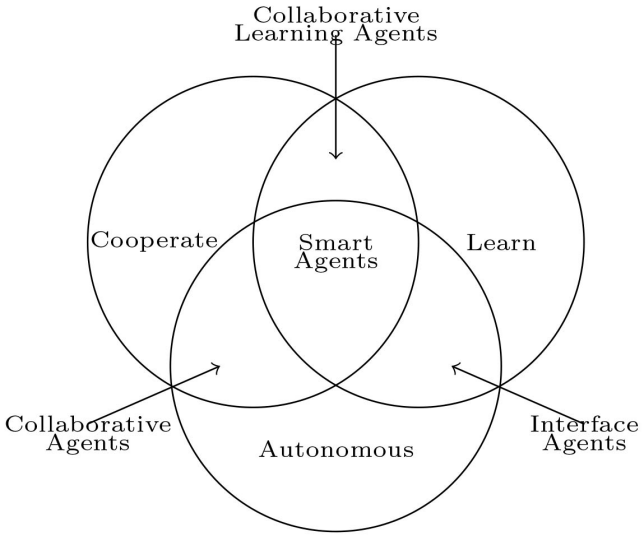

Figure 1. Typology according to the properties of agents.

Systems methodologies by tailoring to fit specific purpose and adapt in the new environment of Internet of Things. In [8], the authors have proposed a generic methodology to design MAS for IoT. The benefit of tailoring methodology is to avoid spend time and resources on activities that may not be essentials to the project under development. In [9], the authors have proposed simple agents having the functions of observation, decision and action, and possess their own knowledge. The obtained agents are modelled as bio-dynamic objects that enjoy the properties of fusion, division and multiplication. Being aware of the context, the proposed agents interact to form potential regional transitory communities, called regions. Being aware of their belonging in a region, agents interact by generating virtual links (virtual extensions). These links produce: fusion, division, multiplication of agents. Claudio Urrea *et al.* [10] have designed Intelligent agents to drive the virtual automobile in the Unity platform, and they are trained using imitation or reinforcement. In learning by reinforcement, a reward function that stimulates the intelligent agent to exert a soft control over the virtual automobile is designed. The learning mechanism is divided into many steps. In the training step, the agents are in a scenario that simulates a four-lane highway while in the test step, they are located in unknown roads created based on random spline curves. Samuel H. Christie *et al.* have proposed in [11] a MAS called Mandrake for programming fault tolerant decentralized applications. An unreusable agent architecture is present and each agent is manually built according to this architecture. Buse DP Sun *et al.* [12], propose architecture based on mobile agents for the integration of data in a system structured in substations. Ingrand *et al.* [13], have proposed an agent architecture for reasoning and control in a real-time system. Noulamo *et al.* [14], have proposed a software agent for monitoring plant diseases in precision agriculture.

These propositions can only be used in a single context. For example, it is not possible to use the same agent from [14] for the diagnosis of plants in phytopa-

thology and for the assessment grading process in the educational field. This paper addresses an agent able to mutate in several domains, with a simple configuration and adaptation.

The generic architecture of an intelligent agent is capable of being made concrete in specific domains, by a simple configuration and adaptation. The mutation is a non-automatic operation, which consists of passing from an abstract agent to a concrete agent. The learning and autonomy capacities of our agent are given by the use of Artificial Neural Networks. The abstract modules of the agent architecture are made concrete through the Multi-Layer Perceptron (MLP) parameters used (eg. Base of weights "W", activation function "f()", pre-processing function "g()"). The concrete agents resulting from this approach are used to implement the services belonging to a (SSII), and integrate them into a multi-agent system to solve more complex problems. This work is organized into 5 sections. After this introduction, Section 2 covers the presentation of the architecture of the automation agent. In Section 3, the results are applied to a few areas of industry: the automation of diagnosis in plant pathology and the automation of grading process in the education system. Section 4 presents some of the results obtained and the resulting discussions. Section 5 concludes the work, followed by some perspectives.

2. Generic Agent Architecture

An agent architecture is the software (or hardware) structure which, from a set of inputs, produces a set of actions on the environment or on other agents. In other words, an agent architecture is a particular methodology for the construction of this agent.

The design methodology used is almost following for the case of agent architecture, the Prometheus model which is in fact a variant of the waterfall model readjusted for agent engineering. In the System Specification phase, the real-world problem is tackled. The system goals, the functional descriptors and the possible actions and percepts of the future agents are developed. The generic model is obtained from the specification phase and the agent designed is of BDI (Beliefs-Desire-Intention) architecture (**Figure 2**).

The knowledge from the domain is put in place in the local knowledge database and later coin with the architecture of the learning process. Both parts produce the structure of the generic neural agent. The detailed phase is intended to produce the agent overview with the processes having their plans and capability descriptors. The neural part is decided with the type of neural to be built (MLP). The generic agent with the specific architecture and knowledge will produce at this level the specific domain agent. The architecture of our mutant agent named MUTAG is presented below. The architecture of our mutant agent is composed of five modules that interact to provide the expected service.

2.1. Pre-Processing Module

The Pre-processing module: It implements a denoising auto-encoder, whose role

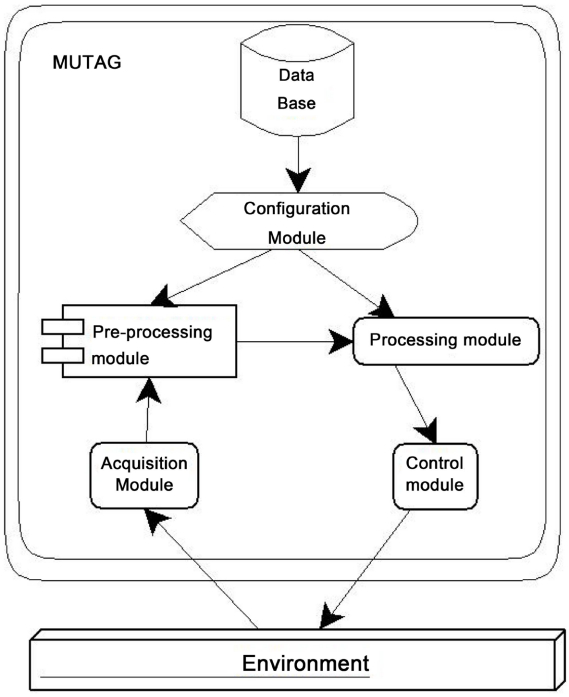

Figure 2. Generic agent architecture.

is to reconstruct the data from the acquisition module in order to extract the characteristics deemed relevant.

The pseudo code that this module implements is given in Algorithm 1.

Algorithm 1: $Pre - Treatmen(X, n, k, step)$

Input: X the image to be pre-processed, of size n; k the size of the sub-windows of the main window to process; "step" is the slip step for the construction of the next sub-window; $g(X, k, i, j)$ is the function for extracting the features deemed relevant;

Output: $Y_{i,j}^{k}$ is the sub-window of size k going from row i to column j, result of processing;

```
1 i ← 1
2 while i + k ≤ n do
3 |   j ← 1
4 |   while j + k ≤ n do
5 |   |   Y^k_{i,j} ← g(X, k, i, j)
6 |   |   j ← j + step
7 |   i ← i + step
8 return Y
```

This algorithm extracts from the image "*X*" provided as input, the characteristics deemed relevant to the domain. Its principle consists in extracting sub-matrices from the initial matrix "*X*" by performing a sliding of step value "step" from left to right and from top to bottom. The function "*g*()" of line 5, is a generic function that aggregates the data base to produce the desired characteristics. The developer adapts the agent to the target domain by modifying the implementation of this function.

2.2. Processing Module

The processing module implements a multi-layer perceptron, which is trained to classify the input signal according to the characteristic coming from the pre-processing module. Figure 3 below defines its structure. In this figure, $W_{i,j}^{k}$ are the weights of the layer k of our MLP. These weights are presented in a matrix of size $M \times N$ where M is the number of output and N-1 is the number of inputs of layer k.

Its specification is given by Algorithm 2.

Algorithm 2: $Classifier(W, K, X)$

Input: X the image to be pre-processed, of size n; k is the number of lattice layers; step the slip step for the construction of the next sub-window; f_i the activation function associated with layer i;

Output: $h(k)$ is the information from the pre-processing module to be classified, result of processing;

```
1 h[0] ← X
2 for i ← 1 to K do
3 |   a(i) ← W^i h(i − 1)
4 |   h(i) ← f_k(a[i])
5 return h(i)
```

This algorithm takes as input a neural network materialized by the weights "W" of the neurons, a source image "X" and performs forward propagation of the weighted inputs to produce the outputs. The function "$f()$" of line 4, used as an activation function, extracts the correct output from the result vector. The developer adapts the processing module to the target domain by modifying the implementation of this function.

2.3. Control Module

A multi-agent system (MAS) is distinguished from a collection of independent agents by the fact that the agents interact in order to jointly perform a task or jointly achieve a particular goal. Agents can interact by communicating directly with each other or, through another agent or even by acting on their environment. The control module is responsible for ensuring communication between agents or with the environment. It is responsible for structuring the information from the processing module and transferring it to other agents in the system or transmitting it to the end user.

2.4. Configuration Module

The configuration module is responsible for adapting the behavior of the processing module to the new domain. Indeed, the weights that accompany our PMC are stored in a database. The configuration module extracts from this database the parameters to configure the PMC and make it transfer to the new domain. Figure 4 is an extract from the Mutant database structure.

The Algorithm 3 below is used to extract the project's weights in our database.

Algorithm 3: *Filed_Weight(IdPro)*

Input: *IdPro* the Identifier of the new domain project;
Output: *W* the matrix weights of the new domain project;
1 Select count(*) Into N From DLM Where Idprojet=IdPro;
2 **for** $i \leftarrow 1$ *to* N **do**
3 Select Count(*) Into M From Node Where NumNode=i;
4 **for** $j \leftarrow 1$ *to* M **do**
5 Select Weight Into W_{ij} From Is_Connected_To, Node where Node.Num_Node=Is_Connected_To.Numnode and Node.Source=i And Node.Target=j;
6 **return** *W*

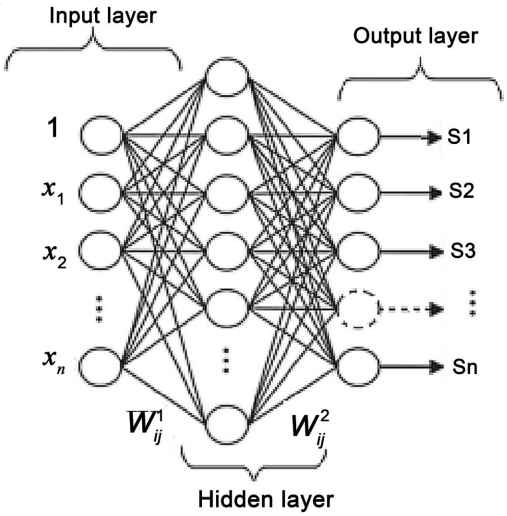

Figure 3. Structure of the MLP used.

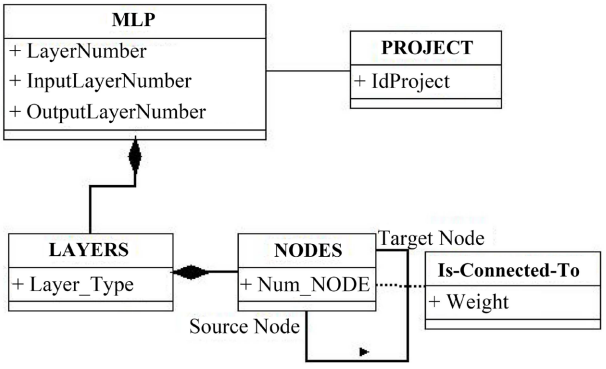

Figure 4. Extract of the database.

This algorithm extracts the parameters from the database modeled in Figure 4, in order to make the agent usable in the new domain. It takes as parameter the project identifier. Line 1 calculates the number "*N*" of network layers. Line 3 calculates for each layer the number "*M*" of neurons of the layer. Line 5 extracts from the "Is-Connected-To" table the weight between $node_i$ and $node_j$.

3. Some Applications of the Proposed Procedure

Our approach is applied on three areas, the automation of grading process in the

field of education, the diagnosis in plant pathology in the agricultural field and in the health domain.

3.1. Agriculture Domain Apply

Improving agricultural productivity is an effective factor in economic growth and poverty reduction, both inside and outside the agricultural sector. Tomato production is a high impact production branch in this sector. It's one of the most cultivated and consumed vegetables in the world because of its short production cycle with high yield and its nutritional and therapeutic properties [15]. However, this crop faces phytosanitary problems (viruses, bacteria, and fungi), which considerably reduce its productivity [16] [17]. The proposed model performs the DDI of tomato from the images of tomato leaves or the videos from cameras installed in the farm.

3.2. Health Domain Apply

Breast cancer is really a public health reality for woman and earlier detection increases the chances of it being cure. Even for specialists, diagnosing breast cancer at an early stage can be challenging because the symptoms are subtle, variable and Computer-Aided Diagnosis systems can be extremely useful. Machine Learning techniques are available for decision support system. However, improving the performance of such system is a challenging task. In this study, a machine learning technique for breast cancer classification is implemented [18]. The proposed approach uses noisy training data augmentation through nearest neighbour interpolation with noisy class label [19]. This model was tested on the Wisconsin Breast Cancer (original) dataset and Multi-layer Perceptron (MLP).

3.3. Application in the Educational Field

Multiple Choice Questions (MCQ) or Unique Choice Questions (UCQ) has proven to be an objective, fair and efficient assessment process in an environment where one wants to select a few candidates out of thousands of applicants, such as entrance examinations. Thanks to information and communication technology tools that reduce the number of participants in the marking process, this form of assessment is now automated and accessible online. However, due to the lack of infrastructure and maintenance, online assessment under such conditions cannot yet be applied in all countries of the world and traditional paper-based assessment still has a bright future ahead. Multiple forms for automatic grading of MCQ are available [20] [21]. In the exam database, each candidate name is associated with a special code called an anonymity number which is pending the corresponding grade. The proposed model makes possible the automatic correction of MCQ paper form and reporting of the grade.

4. Results and Discussions

4.1. Significance of the Study

In order to automate many tasks in Business Processes, an architecture suitable

for a generic agent which is then mutated in a specific agent depending on the domain specification is designed. The intelligence of the agent comes from the embedded ANN. After a training of the ANN of a specific domain, the weights are collected and saved in a file and transferred in a database in the configuration step. The weights of the ANN are loaded from a database which structure is presented in **Figure 4**. Another result is the instantiation of the generic agent in the agriculture and education field.

4.2. Pre-Processing Results

In the educational field, our pre-Processing module implement an algorithm that use the 2D Fast Fourier Transform (2DFFT) and the reduction of the window size tools to perform the extraction of the receiver fields from the entity map lowed to obtain the corresponding final local characteristic in the representation. **Figure 5** shows the local Fourier coefficients obtained for the 7×7 image with the stride of 2 and a window of size 4. In this example, the square window of size 4 is superimposed top left on the image before the computation of the 2DFFT implements in this case by our function "$g()$". For the preservation of the image energy, only the magnitude of the 2DFFT is reported. The window is then shifted successively by two pixels to the right and successively by two pixels (stride) downwards to obtain the required matrix which will undergo normalization before being used.

The local Fourier features representation is a pre-processing technique of our function "$g()$", which aim to reduce inter-class correlation in the dataset. The training loss for the current practice and the local Fourier features as a function of the number of iterations weas plotted. The graph in **Figure 6** shows the training error curve of our approach, tends to zero compared to the traditional approach. This reflects the effectiveness of our agent.

In the Health field, the key results for classification tasks are feature selection, which yields smaller discriminant features from the original data. Three approaches to feature selection were found in literature, including sequential algorithms, randomized algorithms, and exponential algorithms (Uzer *et al.* 2013). The pre-processing module employed Sequential backward selection (SBS) in this study, which sequentially removes input features to improve final accuracy, which implements our function "$g()$". The equation below gives the normalized form of features.

$$F_{nor} = \frac{F - F_{min}}{F_{max} - F_{min}}$$

where F_{nor} is the normalized version of feature F and, F_{min} and F_{max} are respectively the minimum, and the maximum of the feature F. This computation considers each feature column as an array of N samples. The Fourier transform of the original feature is the second feature addressed in this approach. In (linear) signal processing and control theory, the Fourier transform is one of the most often utilized techniques. It converts signals from a time-domain representation

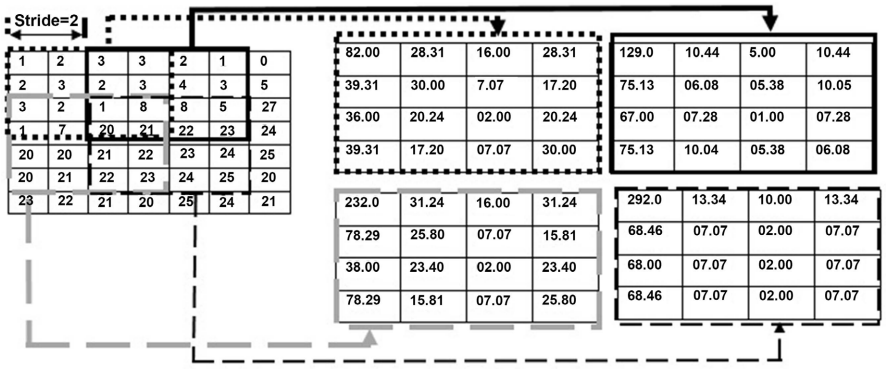

Figure 5. An Example of image representation in the frequency domain.

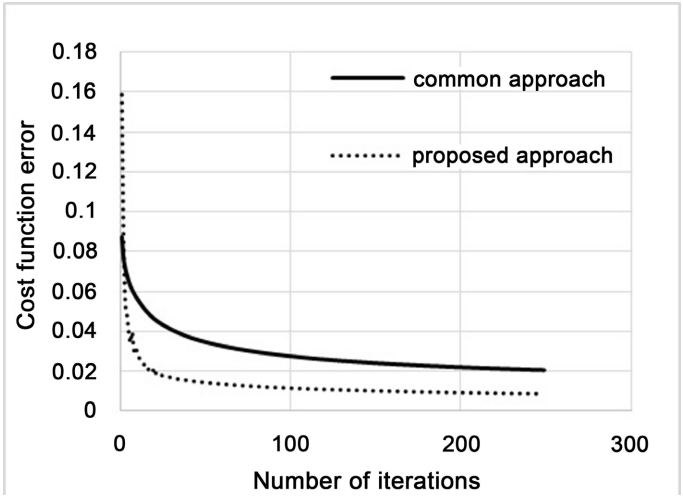

Figure 6. The mean square error evolution for handwriting.

$x(t)$ to a frequency domain representation $X(v)$, with the opposite transformation being feasible. The frequency domain is used to study mathematical functions or signals in terms of frequency rather than time. Transforms are a pair of mathematical operators that are commonly used. For periodic signals, 1D Discete Fourier Transform (1DFT) is appropriate. In the case of a finite-duration or discrete-time signal, the 1D Discete Fourier Transform (1DFT) is applied. The 1DFT of the signal $x(0)...x(N-1)$ is expressed as:

$$X(v)=\sum_{n=0}^{N-1} X[n]\exp\left(-2\Pi j\frac{\mu n}{N}\right)$$

where $\mu=0,1,\cdots,N-1$.

In the agricultural field, the pre-processing is done using the pre-trained network ResNet101 which implements our function "$g()$". Features were extracted from the images using transfer learning based on the pre-trained ResNet101 models implemented by our pre-processing module. Table 1 gives the structure of features extracted from our function "$g()$" and its learning performance.

Figure 7 shows the mean square error curve during training of our agent Pre-Processing with features obtained from ResNet101. The graph shows that

Table 1. Structure of features extracted from ResNet101 and learning outcomes.

N	Level ResNet101	Size Output Vect.	Nbr. Conv. Layer	Learning Perf.
1	50	2048	16	69.2
2	100	2048	31	87.5
3	150	2048	46	87
4	200	2048	61	87.9
5	250	2048	76	87.9
6	300	2048	91	89.4
7	345	2048	105	97.5
8	346	1000	105	70.9

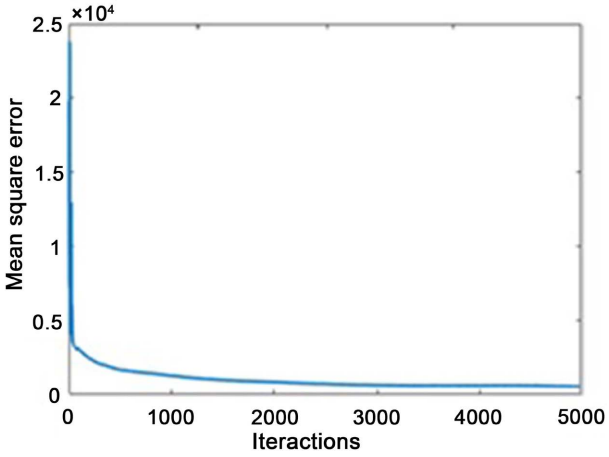

Figure 7. The mean square error for Phyto-Pathology.

the training error curve of our approach tends to zero compared to the traditional approach. This reflects the effectiveness of our agent in term of learning.

4.3. Configuration Results

The configuration of our agent according to a new domain requires the creation of a new project. Project parameters such as weight are then saved. **Figure 8** and **Figure 9** are the project and weight settings registration forms.

4.4. Classifier Results

In the educational field the classification algorithm is done by the multi-layer perceptron (MLP), associated with "*Sigmoid function*" as activation function that present our function "*f*()". MLP represents the fully connected layer to perform the classification.

Figure 10 shows the behavior of the test accuracy with the evolution of the number of iterations. On the test set, the precision curve of the proposed approach is above the precision curve of the commonly used method. This shows

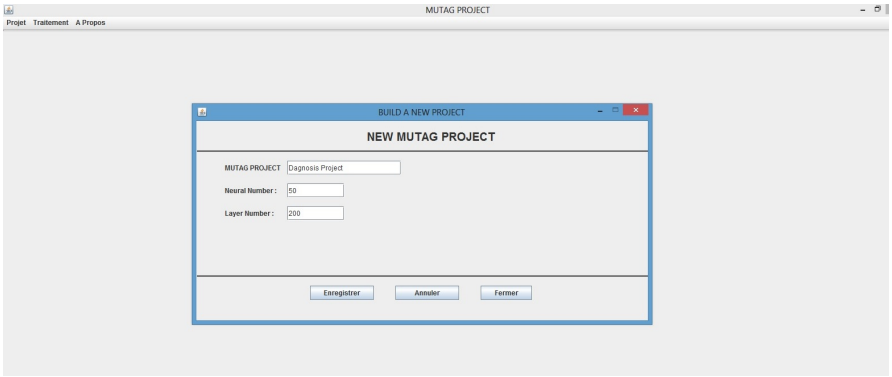

Figure 8. Project form and project weight form.

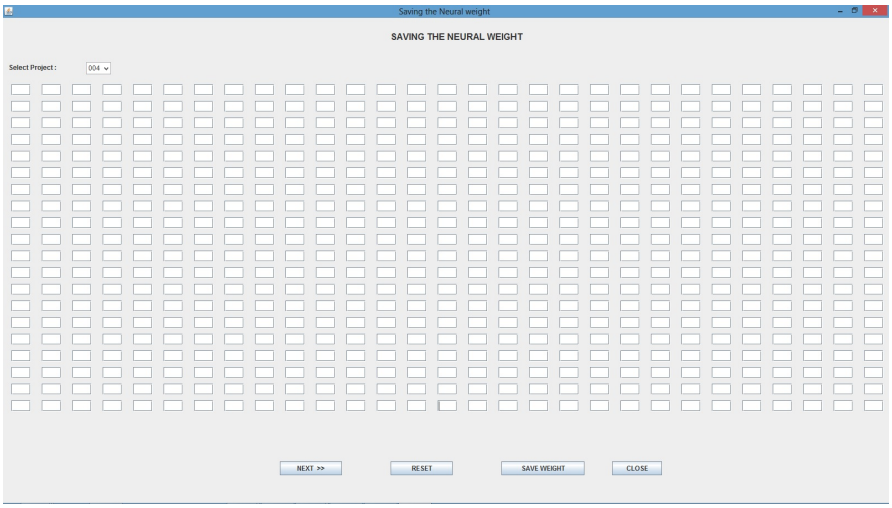

Figure 9. Project form and project weight form.

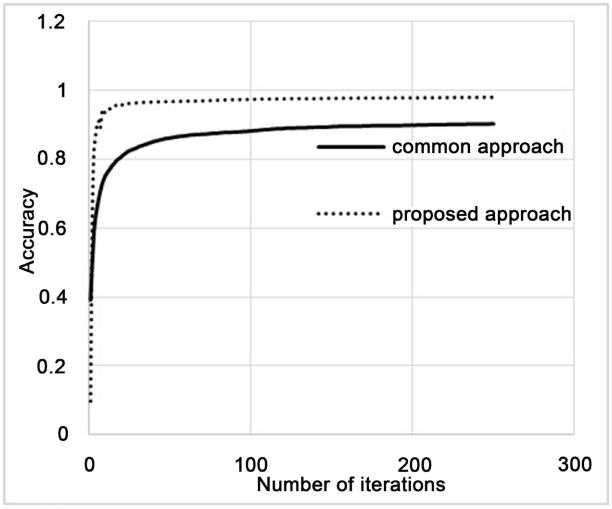

Figure 10. Handwriting behavior of the test accuracy.

that our proposed agent is more effective than the common approach. The accuracy obtained is 98.9%.

In the health field, Artificial Neural Networks (ANN) is used to implement

our function "*f*()". The sum of the inputs x_i is applied to the activation function, which produces a result based on the sum. Synaptic connections between biological neurons are simulated by the w_i.

$$y = f\left(w_0 + \sum_{j=1}^{n} x_j w_j\right)$$

The perceptron is defined by: n inputs noted $(x_1, \cdots, x_n)$ representing the components of the data acquired as input; n weight $(w_1, \cdots, w_n)$ reflecting the synaptic connections between neurons; and n outputs noted $(x_1, \cdots, x_n)$. A bias w_0 is the value from which the neuron is active, as well as an activation function and an exit y. The MLP architecture used includes a hidden layer of 10 neurons. For the best results, the maximum amount of noise applied is 0.112 and the maximum precision achieved is 99.56 percent.

In the agricultural domain, the classification is done by the multi-layer perceptron which implements our function "*f*()". The MLP architectures used here have 400 neurons in the hidden layer and 10 neurons for the output layer representing each class of tomato leaf disease. The results in Table 2 show the mean values of the True-Positive (TP), True-Negative (TN), False-Positive (FP), and False-Negative (FN) values for each class and the performance parameters.

Figure 11 illustrates the accuracy of the test set during the learning of the network with features obtained from ResNet101 at level 345. Analysis of these results shows that after MLP training, the features extracted at level 345 from the pre-trained model ResNet101 provided a better learning performance with an accuracy of 97.2%.

5. Conclusion

This paper proposes the architecture of a generic agent, for process automation. This architecture is based on several modules; therefore the most relevant is the configuration module which allows the use of the agent to be transferred from

Table 2. TP, TN, FP, FN, and performance parameters of our system for each class.

N	Name	M. TP	M. TN	M. FP	M. FN	Accur.	Prec.	Recall	F1-score
1	YLCV	840	2058	6	1	99.76	99.29	99.88	99.52
2	TMV	61	2842	2	0	99.93	96.83	100.0	98.35
3	Target spot	212	2674	5	14	99.35	97.70	93.81	98.51
4	Spider mite	267	2628	5	5	99.66	98.16	98.16	98.90
5	Mold	296	2593	10	6	99.45	96.73	98.01	98.07
6	S. leaf spot	146	2755	0	4	99.86	100.0	97.33	99.93
7	Late blight	296	2597	6	6	99.59	98.01	98.01	98.79
8	No disease	253	2650	0	2	99.93	100.0	99.22	99.97
9	Early Blight	142	2747	11	5	99.45	92.81	96.60	96.02
10	Bacterial spot	342	2551	5	7	99.59	98.56	97.99	99.07

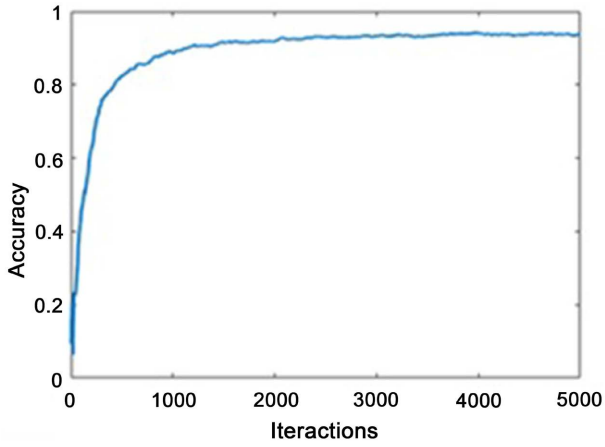

Figure 11. Phyto-Pathology behavior of the test accuracy.

one domain to another, the pre-processing module which implements a generic extraction of relevant parameters from source data in order to facilitate classification and the classification module which also implements a generic output recognition algorithm. The proposal is applied to diagnosis in plant pathology, breast cancer and to the automation of the grading process in education in order to better highlight its genericity character. An extract of the results from the implementation in each area is presented. The obtained agents offer performances that follow those of the operations implemented by our pre-processing and classification modules. In the field of education, the agent has achieved accuracy of 98.9% and in the field of breast cancer diagnosis and agriculture; the accuracy is 99.56% and 97.2% respectively.

However, the major limitation of this approach is the fact that our agent only accepts images as input to the process. It would have been interesting to use information other than the images as input. The process of switching from a generic agent to a specific domain is currently done manually. Our next challenge will be to automate this process and Model Driven Engineering seems to us to be a potential path.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Noulamo, T., Tanyi, E., Nkenlifack, M., Lienou, J.-P. and Djimeli, A. (2018) Formalization Method of the UML Statechart by Transformation toward Petri Nets. *IAENG International Journal of Computer Science*, **45**, 505-513.

[2] Noulamo, T., Talla, B.F. and Lienou, J.-P. (2020) Automatic Generation of Web Users Interfaces Using a Model-Driven Approach. *International Journal of Scientific & Engineering Research*, **11**, 1439-1448.

[3] Noulamo, T., Talla, B.F., Wane, M. and Takou, L.H.N. (2020) A Model-Driven Approach for Developing Web Users Interfaces of Interactive Systems. *International*

Journal of Computer Trends and Technology, **68**, 33-43.
https://doi.org/10.14445/22312803/IJCTT-V68I4P107

[4] Noulamo, T., Tanyi, E., Nkenlifack, M. and Lienou, J.P. (2016) Model-Driven Engineering Applied to the Control and Monitoring of Dynamic Systems. *International Journal of Computer Science and Software Engineering*, **5**, 183-194.

[5] Djontu Tajouo, F.A., Noulamo, T. and Lienou, J.-P. (2021) Procedure for the Contextual, Textual and Ontological Construction of Specialized Knowledge Bases. *European Journal of Electrical Engineering and Computer Science*, **5**, 62-67.
https://doi.org/10.24018/ejece.2021.5.1.282

[6] Noulamo, T., Choppy, C. and André, É. (2015) Filter Pattern for Consistent Use of Data in Real-Time Systems. *Advances in Computer Science and Engineering*, **14**, 73-96.
https://doi.org/10.17654/ACSEMay2015_073_096

[7] Nwana, H.S. (1996) Software Agents: An Overview in Knowledge Engineering Review. *The Knowledge Engineering Review*, **11**, 205-244.
https://doi.org/10.1017/S026988890000789X

[8] Manate, B., Fortiş, F. and Moore, P. (2014) Applying the Prometheus Methodology for an Internet of Things Architecture. 2014 *IEEE/ACM 7th International Conference on Utility and Cloud Computing*, London, 8-11 December 2014, 435-442.
https://doi.org/10.1109/UCC.2014.55

[9] Fougères, A.J. and Ostrosi, E. (2018) Intelligent Agents for Feature Modelling in Computer Aided Design. *Journal of Computational Design and Engineering*, **5**, 19-40. https://doi.org/10.1016/j.jcde.2017.11.001

[10] Urrea, C., Garrido, F. and Kern, J. (2021) Design and Implementation of Intelligent Agent Training Systems for Virtual Vehicles. *Sensors*, **21**, Article 492.
https://doi.org/10.3390/s21020492

[11] Christie, S.H., Chopra, A.K. and Singh, M.P. (2022) Mandrake: Multiagent Systems as a Basis for Programming Fault-Tolerant Decentralized Applications. *Autonomous Agents and Multi-Agent Systems*, **36**, Article No. 16.
https://doi.org/10.1007/s10458-021-09540-8

[12] Buse, D.P., Sun, P., Wu, Q.H. and Baker, B. (2001) An Agent Based Architecture for Substation Data Integration. *Proceedings of CIGRE International Conference on Power Systems*, Wuhan, September 2001, 551-554.

[13] Ingrand, F.F., Georgeff, M.P. and Rao, A.S. (1992) An Architecture for Real-Time Reasoning and System Control. *IEEE Expert*, **7**, 34-44.
https://doi.org/10.1109/64.180407

[14] Noulamo, T., Djimeli-Tsajio, A., Lienou, J.P. and Fotsing-Talla, B. (2022) Agent Platform for the Remote Monitoring and Diagnostic in Precision Agriculture. *Engineering Letter*, **30**, 972-980.

[15] Costa, J.M. and Heuvelink, E.P. (2018) The Global Tomato Industry. In: Heuvelink, E., Ed., *Tomatoes*, CABI, Boston, 1-26. https://doi.org/10.1079/9781780641935.0001

[16] Djimeli-Tsajio, A.B., Thierry, N., Jean-Pierre, L.T., Kapche, T.F. and Nagabhushan, P. (2022) Improved Detection and Identification Approach in Tomato Leaf Disease Using Transformation and Combination of Transfer Learning Features. *Journal of Plant Diseases and Protection*, **129**, 665-674.
https://doi.org/10.1007/s41348-022-00608-5

[17] Jones, J.B., Zitter, T.A., Momol, T.M. and Miller, S.A. (2014) Compendium of Tomato Diseases and Pests. APS Press, St. Paul, MN.

[18] Korjus, K., Hebart, M.N. and Vicente, R. (2016) An Efficient Data Partitioning to

Improve Classification Performance While Keeping Parameters Interpretable. *PLOS ONE*, **11**, e0161788. https://doi.org/10.1371/journal.pone.0161788

[19] Sun, R., Li, D., Liang, S., Ding, T. and Srikant, R. (2020) The Global Landscape of Neural Networks: An Overview. *IEEE Signal Processing Magazine*, **37**, 95-108. https://doi.org/10.1109/MSP.2020.3004124

[20] Al-Sadi, J., Al-Halabi, D. and Al-Halabi, H. (2014) MCQ Exams Correction in an Offline Network Using XML. *GSTF Journal on Computing* (*JoC*), **1**, 176-182.

[21] Habeek, M., Dridi, C.E. and Badeche, M. (2020) Automatic Correction of Free Format MCQ Tests. *International Journal of Software Innovation* (*IJSI*), **8**, 50-64. https://doi.org/10.4018/IJSI.2020010103