Micro Agent and Neural Network based Model for Data Error Detection in a Real Time Data Stream

Sidi Mohamed Snineh[1], Mohamed Youssfi[2], Abdelaziz Daaif[3], Omar Bouattane[4]
SSDIA ENSET Mohammedia, Hassan II Universit, Casablanca, Morocco

***Abstract*—In this paper, we present a model for learning and detecting the presence of data type errors in a real time big data stream processing context. The proposed approach is based on a collection of micro-agents. Each micro-agent is trained to detect a specific type of error using an atomic neural network based on a sample multilayer perceptron. The supervised learning process is based on a binary classifier where the training data inputs are represented by data types and data values. The Micro-Agent for Error Detection (MAED) is deployed at several instances depending on the number of error types to be handled. The orchestration mechanism of data streams to be examined is performed by a special Host Micro Agent (HMA). This later receives in real time a data stream, splits the current record into several elementary fields. Each field value is streamed to an instance of MAED Agent which responds with a signal of presence or not of a specific data type error of the corresponding data field. For each detected data type error, the HMA Agent selects and performs the appropriate cleaning algorithm from a repository to correct the present errors of the data stream. To validate this approach, we propose an implementation based on Framework Deep Learning 4j for the Machines Learning part and JADE as a Multi Agent System (MAS) platform. The used dataset is generated by an events generator for smart highways.**

***Keywords*—Micro-agent; machine learning; errors; big data; multilayer perceptron; stream processing**

I. INTRODUCTION

The emergence of new technologies in recent years, such as sensors, smart cities, the Internet of Things, social networks, e-commerce etc., as well as the evolution of some other technologies of information systems of companies especially in the banking sector, medicine, industry have allowed the generation of a large amount of data. This fast evolution forces companies to evolve their information systems to keep up with the fast pace of data flow whether for processing, analysis or storage.

Several technologies and several lines of research have been proposed to find solutions to the management and storage of this massive data. Among these technologies is machine learning, which has been very successful in recent years in several activity areas such as speech recognition, computer vision and natural language processing [1].

Given the abundance of data flows that we have today, this science (machine learning) has changed dimension and to aroused a great interest of the scientific community which has allowed it to be present in almost all fields.

This abundance of data (big data) represented today by ZettaBytes and tomorrow by PetaBytes-for example according to IDC1 by 2025, the world will create 180 zettabytes of data per year against 4.4 zettabytes in 2013 - allows businesses to exploit machine learning technology in other areas of research to bring new solutions.

But even if we have large amounts of data and high computing power, it will not be enough for companies to make correct predictions to anticipate and make the right decisions. Indeed, the exponential flow of data and the speed of its generation give a high probability of having different types of errors according to the different types of data (Structured "S", Semi-structured "SS" or UnStructured "US") (Fig. 1) manipulated by enterprises and can therefore negatively influence decision-making.

It is therefore essential that the data received by the companies must be of high quality.

Our approach proposes a contribution, in this sense, that helps to improve the quality of data from the large data flows which are intended to decision-making. This approach combines two technologies: machine learning technology and multi-agents systems in order to distribute processing. The main goal is to detect frequent errors in real time in large data streams. This approach also complements our contribution "Frequent Big Data Error Handling Repository" [2].

This approach reveals several advantages as:

- Splitting our problem into smaller, more manageable units.
- Having an extensible model: if a new error is detected, by a manager of the company, and if its frequency of appearance is important, the abundance of the data of the company allows the training and the test of a new ANN to identify this new error.
- Exploiting the concept of machine learning because this branch of artificial intelligence (AI) is becoming widespread.
- Using this approach to all companies not only to giant companies such as Amazon, Facebook and Google but also to other companies that now have big data thanks to unstructured data such as text, sensors and images.

[1]- Michael Kanellos, https://www.forbes.com/sites/michaelkanellos/2016/03/03/152000-smart-devices-every-minute-in-2025-idc-outlines-the-future-of-smart-things/#4c3569df4b63., Mar 3, 2016

- Easy portability of this approach to other companies of different activities;
- Using this approach in information systems by the means of data warehouses and data stores.
- Generalizing this approach to other new situations to improve data quality.

This article is organized as follows: We will begin by giving a background of machine learning and multi-agents systems. After we presented some related work and then we describe the dataset used to train and test each micro-agent. We then detail the proposed model and before concluding, we explain the operation of this approach.

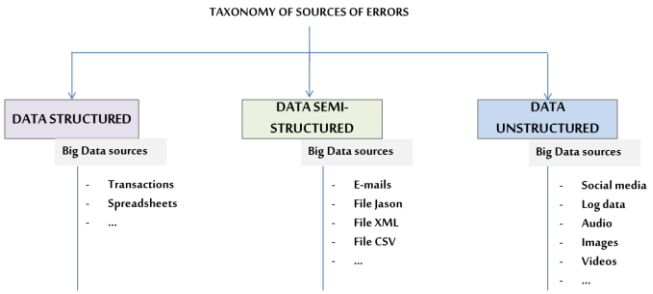

Fig. 1. Taxonomy of different Sources of Big Data Errors [2].

II. Background

A. Machine Learning

Machine learning is a type of artificial intelligence where the machine is able to learn on its own after training based on a large amount of data.

Machine learning uses the network of artificial neurons imitating the human brain. This network is composed of several layers of neurons; its behavior depends on its network architecture. The architecture of a network of neurons can be defined by the following elements [3]:

- Number of neurons
- Number of layers
- Types of connections between layers

Machine learning is used in several areas:

- Automotive,
- manufacturing,
- consumer products,
- finance,
- agriculture,
- energy,
- health car,
- pharmaceuticals,
- public and social sector,
- media,
- telecom,
- transportation, travel and logistics,
- etc.

In this paper we have used the atomic neural network (ANN) for the detection and identification of data type errors in real time big data stream.

We have chosen to use a variant of the perceptron of Fig. 2, the first and the smallest unit of neural networks, as an error detection algorithm to have a minimum level of granularity and to split this problem to small units that allow the extension, integration and to be easy management of our system.

The choice of the initialization of the weights of a neural network is very important because it allows preventing layer activation outputs from exploding or vanishing during training neural network. We chose the method proposed by He et al. [4] which initialization of weights for ReLu is random, but depends on the size of the previous neuron layer. This allows controlling initialization of the weights in order to solve the vanishing/exploding gradient problem.

In fact, we have chosen to use an atomic neural network based on a sample multilayer perceptron. It is composed of three layers of neurons Fig. 3: the input layer, the hidden layer (s) and the output layer.

- The input layer is a set of neurons that carry the data to be processed.
- The hidden layer, very often several hidden layers, is an intermediate part between the input layers and the output layer. This is where the network stores its internal abstract representation of learning data, in the same way that a human brain has an internal representation of the real world[2].
- The output layer represents the end result of the neural network: its prediction.

B. JADE : Java Agent Development Framework

JADE is a multi-agent platform. It is a framework that allows the development of multi-agent systems. It has three modules:

- Directory Facilitator (DF): provides a yellow pages service to the platform.
- Communication Agent Channel (ACC): manages communication between agents.
- Agent Management System (AMS): supervises the registration of agents, their authentication, access and use of the system.

These three modules are activated each time the platform is started.

[2]- Ivan Vasilev, “A Deep Learning Tutorial: From Perceptron’s to Deep Networks”, https://www.toptal.com/machine-learning/an-introduction-to-deep-learning-from-perceptrons-to-deep-networks

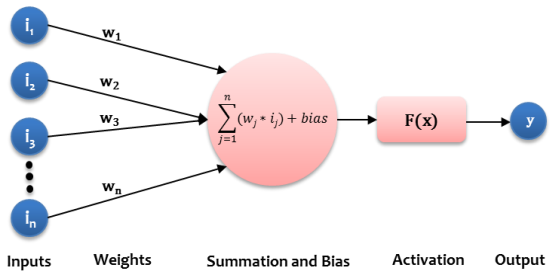

Fig. 2. The First Artificial Neural Networks.

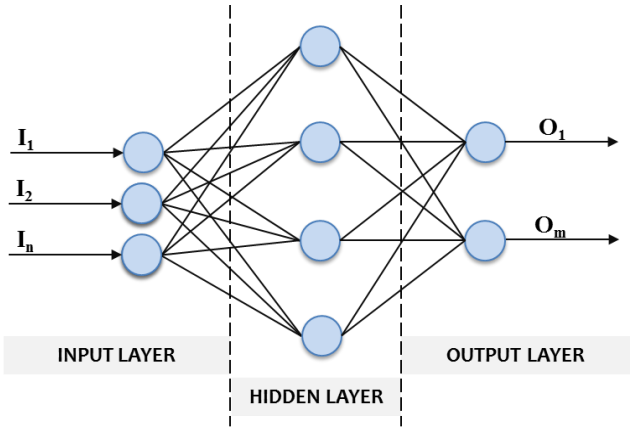

Fig. 3. Architecture of Multilayer Perceptron.

Each JADE instance is called "Container" and can contain multiple agents. The set of containers constitutes a platform. Each platform must contain a special container called "main-container" where all other containers are registered as soon as they are launched. Containers can be run on different machines, creating a distributed platform. Agents in a JADE application perform tasks and interact by exchanging messages (Fig. 10).

The choice of multi-agents in this paper is justified by the following advantages:

- Each agent is responsible for a specific task in the data processing;
- Agents are autonomous and distributed;
- The cooperation between the agents is done by the transmission of messages. They communicate in the same way [5].
- Agents can learn, perceive, reason, judge and make decisions based on their knowledge of themselves and their environment and can cooperate and coordinate through communication [6].

III. Related Work

Since the advent of Data Warehouse and Data mart, which were among the first data sources for decision-making, several research projects were proposed to improve the quality of data and to ensure reliable decision-making, such as In [7], the authors proposed hierarchical Framework compound of dimensions, of characteristics and of quality indications, and on the basis of which they built a process of dynamic evaluation of the quality of the data of flows Big Data.

In [8], authors proposed a solution which allowed the evaluation of the quality of the data of the social media in every phase of processing in the architecture Big Data and the improvement of the decision-making by supplying data validated in real time for the user.

Other authors [9] proposed a novel unsupervised real-time anomaly detection algorithm. The technique is based on an online sequence memory algorithm called Hierarchical Temporal Memory (HTM).

To our knowledge, the relationship between machine learning, multi-agents and the quality of big data flows is very rare. Lately, there are some new researches, as in [10] which show how machine learning can increase the efficiency and cost-effectiveness of measures of error identification and correction using learning algorithms supervised and how machine learning can help overcome the lack of data.

The article [11] uses a method of prediction of traffic flows based on deep learning that is also an artificial intelligence technique that considers the spatial and temporal correlations inherently.

Another article [12] that talks about a point view on the problems of the quality of big data flows and which states that research in this problem will be specific to each sector. This is what we propose in this paper: detection of errors of big data flows for a company of a given sector of activity.

We have therefore thought to create a relationship between these technologies (machine learning, micro-agents) to make a contribution to this problem of data quality omnipresent in companies.

IV. Data

In this paper, the dataset used to train and test each micro-agent are based on the data generated by the spatio-temporal traffic event generator for real road networks [13]. In the real case the considerable number of sensors used in highways generates large, fast and real-time data flows, especially if the density of the sensors is high enough. So, the analysis and exploitation of these data can be used to Anticipate and predict how to control the traffic of each highways axis and consequently help to make the right decisions to save lives.

To train our micros-agents, we took in our case a single XML data file that represents the vertices (vertex.xml) of the event generator [13]. These data are characterized by the following attributes:

- name: Sensor identifier (ID)
- type: Element type (Enumeration : I (Entrance), IO (Entrance/Exit), X (Exchange), R (Service Area), T (Toll), S (Sensor), O (Exit)
- label: Name of the highway (string)
- locality: The locality name of the sensor position (string)

- long: Longitude (double)
- lat: Latitude (double)
- factor: Attendance factor.

V. Proposed Model

Data errors are often related to fields and records Fig. 4. We have proposed two approaches Fig. 5 in order to detect errors. In this paper, we have chosen to describe and to develop the first approach. It allows detecting the presence of data type errors in a real time big data stream.

The second approach will be the subject of a future paper. It consists to detect errors in the data in the records. These errors are often related to referential integrity, duplicates and other errors.

The approach proposed in this article, therefore, aims to detect frequent errors in real time in each elementary field. It is based on machine learning technologies and micro-agents (the choice of the term "micro" refers to the level of granularity of the tasks of each agent). The relation these two technologies will allow us to use the same concept for all error detection algorithms. The idea is to associate each micro-agent with an atomic neural network trained, tested and serialized in a file in order to detect a specific error for a single type of data. We will have in the repository of the company a collection of "intelligent" micro-agents and will be distributed for the detection of errors specific to each attribute.

The multilayer perceptron used in our approach, represented by Fig. 6, is composed of two inputs, hidden layers and an output layer.

A. The Input Layer of Each ANN

Since data from big data streams are heterogeneous and of different types (Structured "S", Semi-structured "SS" or UnStructured "US"), we have assigned to each data type a value that will be used in our approach to differentiate between data types and their errors. Table I shows an example of the data types and their values.

We need to use for this layer, two inputs for each multilayer perceptron:

- The data of a single attribute of the data flow.
- The data type of this attribute.

The idea of using a single multilayer perceptron for a single type of error of a data type is the decomposition, distribution, and standardization of error detection tasks. Indeed, once the flow of the data arrives, the Host Micro Agent (HMA), which is at the reception of these data, splits the current record into several elementary fields. Each elementary field will be the first input of the corresponding atomic neural network.

The second input is none other than the data type of each attribute chosen by the same micro-agent (HMA) from the types of data stored in the company repository (Fig. 7).

It is these two data, plus the types of data errors, stored in the enterprise repository (Fig. 7) that represents the dataset used to pass each multilayer perceptron through three stages before they are used:

- The training of each ANN;
- The test of each ANN;
- Serializing each ANN so that it will be ready for use.

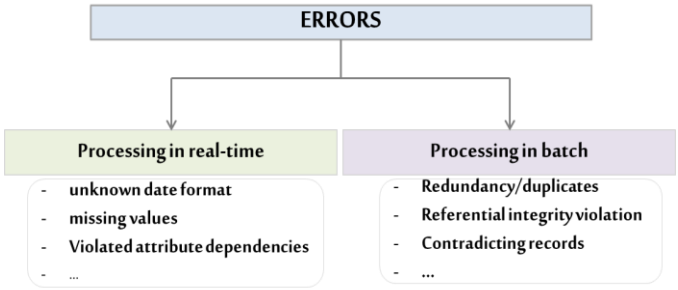

Fig. 4. Types of Errors.[2].

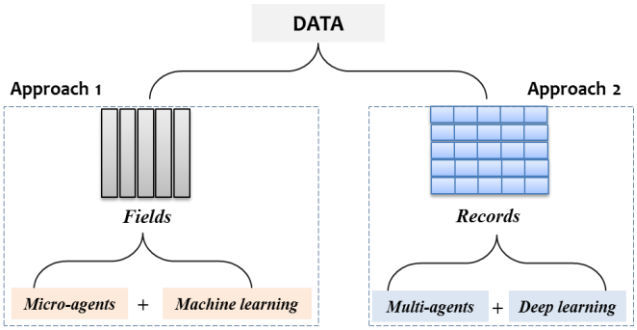

Fig. 5. Proposed Approaches.

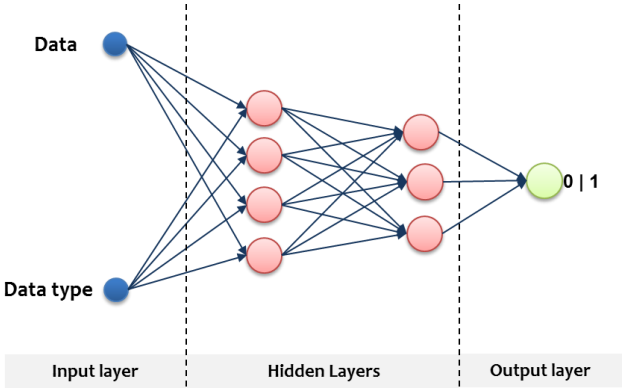

Fig. 6. Multilayer Perceptron Architecture used in this Approach.

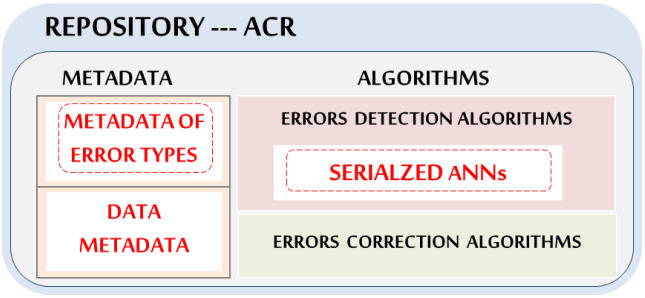

Fig. 7. Big Data Frequent Error Management Repository [2].

TABLE I. DATA TYPES

Data Type	value
Integer	1
double	2
String	3
Date	4
Boolean	5
Image	10

B. Training and Testing of ANNs

The training and testing steps were limited earlier by the lack of data and the slowness of the processors. Now with the abundance of data (Big Data), and the technological evolution such as processors speed and distributed and parallel processing, the companies can use them in order to have "intelligent" ANNs trained in a reasonable time to improve the quality of the data.

The steps mentioned above are essential in our approach, because we will only serialize neural networks which have a high accuracy rate, after variation of the number of hidden layers, during the training and test operation.

The datasets in Tables IV and VI present examples for training and testing some micro-agents.

Table II provides an example of a dataset for the "long" attribute of double type. The corresponding ANN will be trained to detect the missing value error of this attribute. The choice of the type of error is explained in the section: Instances of micro-agents.

Table III gives an example of a dataset for the "type" attribute of type String. The corresponding ANN will be trained to detect the “check constraint” error of this attribute containing enumerated data.

TABLE II. DATASET FOR THE "LONG" ATTRIBUTE

Inputs		**Outputs**
long : Longitude	***Data type***	
-6.83255	2	0
-7.6113801	2	0
5.599	2	1
	2	1
-7.9999399	2	0

TABLE III. DATASET FOR THE "TYPE" ATTRIBUTE

Inputs		**Outputs**
type	***Data type***	
I	3	0
R	3	0
IIO	3	1
r	3	0
B	3	1
H	3	1

C. Instances of Micro-Agents

Once the micro-agent HMA separates the current record into fields, this micro-agent looks for the types of errors of each type of data in the metadata stored in the repository of company.

Table IV shows an example of metadata of the types of errors related to each type of data. This metadata stored in the repository, will be used in our approach so that the micro-agent HMA deploys several instances of micro-agents according to the number of errors of the same data type. For example, if we have in a data stream, N attributes of type "String" with M errors stored in the repository, the micro-agent HMA will deploy M*N instances of the micro-agents for detection possible errors of the type "String". It will do likewise for the other data types of this same flow of data by running the other micro-agents instances.

D. Serialization of ANNs

Serializing neural networks of machine learning in files is the last step in our approach before they are used. It makes it possible to associate each ANN with a micro-agent. This step depends on scores obtained after the training of each multilayer perceptron and its test.

Tables V and VI, there are examples that explain the link between errors of each data type, serialized ANNs, and instances of micro-agents.

For example, the "Check constraint" error of the "double" type serialized under the name "ANN_21" associated with the micro-agent MAED_21 to detect error of this data type. The instance "ANN_32" associated to the micro-agent MAED_32 serves to detect "check constraint" error of data type "String".

TABLE IV. METADATA OF ERRORS OF EACH TYPE OF DATA

Data types	**Errors**
Integer	Missing value
	Error format
	…
Double	Missing value
	Error format
	…

TABLE V. INSTANCES OF MAED THE DOUBLE TYPE

Errors	**ANN serialized**	**Instances MAED**
Check constraint	ANN_21	MAED_21
Error Type	ANN_22	MAED_22
Null	ANN_23	MAED_23

TABLE VI. INSTANCES OF MAED THE TEXT TYPE

Errors	**NN**	**MAED**
Null	ANN_31	MAED_31
Check constraint	ANN_32	MAED_32
Error Type	ANN_33	MAED_33

E. The Hidden Layers of ANN

The number of neurons in the layer is the strength of machine learning because during the training and testing stages of each ANN the number of these layers is varied to have very high levels. Indeed, some results highlight the interest of considering two or more hidden layers to obtain more parsimonious and more efficient networks, by composing several levels of non-linearity [14].

F. The Output of ANN

The output layer of our multilayer perceptron is binary. It corresponds to two states. If the data in the field contains an error, they will be represented in the data set as "1". Otherwise, if this data does not contain errors it will be represented by the value "0". See the example in Tables II and III. We have chosen supervised learning that allows the ANN to learn from each example with the aim of being able to generalize its learning to new cases.

VI. Operation of this Approach.

The operation of our approach is detailed in the following two sections:

A. Using the Host Micro Agent (HMA)

The use of multilayer perceptron (ANN) is related to micro-agents guarantee distributed and parallel execution. Each micro-agent associated with an ANN constitutes a single unit to perform a specific task. Among these micro-agents we have the Host Micro Agent which receives the data flows of the sensors in real time. Each sensor of our event generator [13] sends a data intercepted by the framework Kafka Streams Fig. 8. At the output, after transformation and aggregation of the data, the HMA will support the data flows and perform the following tasks:

- After the reception of the data flows HMA splits the current recording into fields (attribute).
- Searches the data type of each attribute in the repository metadata (Fig. 7).
- Adds each attribute to its data type so as to form the inputs of the corresponding neural network.
- Searches in the repository of the company the types of errors of each type of data in order to choose the appropriate micro-agents.
- Depending on the number of error types of each data type, HMA deploys the necessary number of instances of the micro-agents.
- HMA distributes the instances of micro-agents in multiple machines to ensure task distribution.
- For each elementary data, it requests parallel execution of all micro-agents to detect any errors.
- On reception of the result, if an error is detected, that is to say that agent HMA receives a positive answer "1" it searches for in the repository of the company the appropriate correction algorithm. In this case, the agent HMA is informed that the sensor that sent the data is damaged. Automatically, it will run a virtual sensor [15] that will provide data from historical data until the physical sensor is repaired.

Fig. 9 shows the arrival of a data flow, the HMA splits it into several attributes and deploys, depending on the number of errors of each attribute, several instances of micro-agents.

For example, the instances of micro-agents linked to the serialized multilayers perceptron's (MAED_31, MAED_32 etc., MAED_3n) are deployed to detect any errors of the attribute "type: String". It's the same for the other attributes "long", "locality", etc.

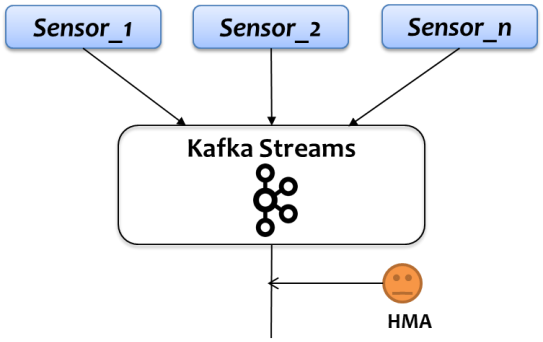

Fig. 8. Receipt of the Data Flow by HMA.

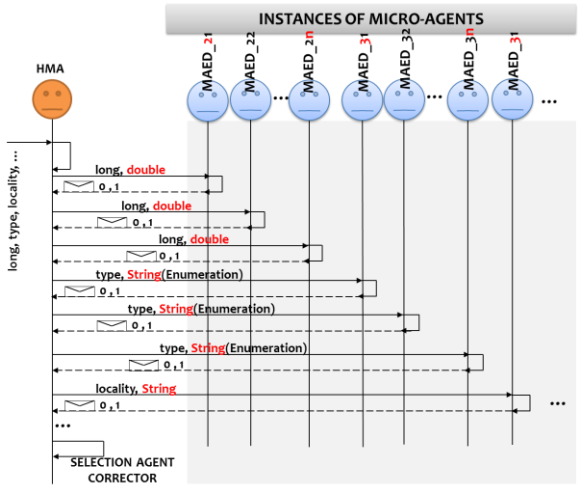

Fig. 9. Using the Host Micro Agent.

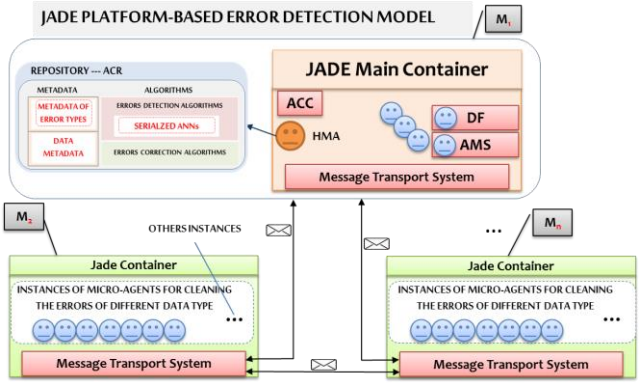

Fig. 10. JADE Platform-based Error Detection Model.

B. Multi-Agent System for Error Detection

The JADE platform chosen in this paper is used to distribute errors detection tasks. The Host Micro Agent distributes instances of micro-agents in multiple machines.

This platform allows having identical instances of micro-agents. Fig. 9 shows an example of these identical instances such as " MAED_31" deployed by the HMA because we have two attributes of the same type "type: String" and "locality: String".

Fig. 10 shows the content of the platform of our approach and the operation of the instances of micro-agents on several machines. In the machine "M1", for example, one finds "JADE Main Container" that must be started to be able to deploy the host micro agent HMA. We also find on this machine the enterprise repository containing in particular the metadata of the types of error, the data metadata, the serialized multilayers perceptron's and the algorithms of correction.

In the other machines there are still other instances of the micro-agents of detecting errors. For instance, the machine "M2" contains instances of detecting errors of some data types such as "Integer", "String", etc. Other detecting errors instances will be distributed to other machines to speed up processing.

VII. Conclusion

In this paper, we have proposed a model for learning and real-time automatic detection of errors existing in a big data stream. The used supervised learning process is based on a binary classifier where the training data inputs are represented by data types and data values. This approach relies on a collection of micro-agents that guarantee distributed and parallel execution. Each micro-agent is linked with a sample multilayer perceptron which is trained, tested and serialized (ANN) in order to perform a single task: detect a single error of a single type of data.

The Host Micro Agent (HMA), in this approach, receives the flow of data and separates the current record into several elementary data (field). Then, it searches in the enterprise repository for the data type of each attribute to compose the multilayer perceptron entries. This data will be supplemented by the types of errors sought by the agent "HMA" starting from the metadata. In the end, it will deploy the necessary number of instances of the micro-agents and distribute them on the machines of the system.

When the agent HMA receives a message containing an error, it selects and runs the appropriate cleaning algorithm from an enterprise repository in order to correct the existing error.

Our approach is extensible because we can train other micro-agents for possible new errors. It is also easy to integrate into other areas of activity since the micros agents are very small units that are easy to set up and manage.

The first results after load they micro-agents are satisfactory especially for the errors cited in this paper.

In perspective, we will detail "approach 2" figure by applying the concepts: multi-agents and deep learning to detect errors related to records such as duplicates, errors related to referential integrity, etc.

References

[1] Xue-Wen Chen and Xiaotong Lin, "Big Data Deep Learning: Challenges and Perspectives", IEEE Access, vol 2, May 28, 2014.

[2] Sidi Mohamed Snineh, Mohamed Youssfi, Omar Bouattane, Abdelaziz Daaif, Oum El Kheir ABRA, "Real-time management model for frequent Big Data errors : Automatic Clean Repository For Big Data (ACR), IEEE Xplor 08 November 2018.

[3] Josh Patterson and Adam Gibson, "Deep Learning","Chapter 1. A Review of Machine Learning" Publisher: O'Reilly Media, Inc. Release Date: August 2017 ISBN: 9781491924570.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification", IEEE International Conference on Computer Vision (ICCV), IEEE Xplore: 18 February 2016.

[5] Bartłomiej Twardowski, Dominik Ryzko, "Multi-agent architecture for real-time Big Data processing", IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 20 October 2014.

[6] Huaglory Tianfield, Jiang Tian and Xin Yao, "On the Architectures of Complex Multi-Agent Systems", Proceedings of the Workshop on "Knowledge Grid and Grid Intelligence" (ISBN 0-9734039-0-X), held at the 2003 IEEE/WIC International Conference on Web Intelligence / Intelligent Agent Technology, October 13-16, 2003, Halifax, Canada, pp. 195–206.

[7] Li Cai and Yangyong Zhu, "The Challenges of Data Quality and Data Quality Assessment in the Big Data Era". Data Science Journal, 14: 2, 2015 pp.1-10.

[8] Anne Immonen, Pekka Pääkkönen, and Eila Ovaska, "Evaluating the Quality of Social Media Data in Big Data Architecture". DOI 10.1109/ACCESS.2015.2490723, IEEE Access.

[9] Subutai Ahmad, Alexander Lavin, Scott Purdy, Zuha Agha, "Unsupervised real-time anomaly detection for streaming data". Neurocomputing Volume 262, 1 November 2017, Pages 134-147. https://doi.org/10.1016/j.neucom.2017.04.070.

[10] Tobias Cagala, "Improving Data Quality and Closing Data Gaps with Machine Learning", A chapter in "Data needs and statistics compilation for macroprudential analysis" vol. 46 from Bank for International Settlements, May 5, 2017.

[11] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang, Fellow, IEEE, "Traffic Flow Prediction With Big Data: A Deep Learning Approach", IEEE Transactions on Intelligent Transportation Systems (Volume: 16 , Issue: 2 , April 2015) Page(s): 865–873.

[12] Dhana Rao, Venkat N Gudivada and Vijay V. Raghavan "Data Quality Issues in Big Data", 2015 IEEE International Conference on Big Data (Big Data), IEEE Xplore: 28 December 2015.

[13] Abdelaziz Daaif, Mohamed Youssfi, Omar Bouattane and Oum El Kheir Abra, "An Efficient Distributed Traffic Events Generator for Smart Highways". (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 8, No. 7, 2017.

[14] Médéric Morel , Pirmin Lemberger , Marc Batty and Jean-Luc Raffaëlli, "Big Data and Machine Learning", Second Edition, DUNOD, ISBN:9782100756667, 05/20/2016.

[15] Abdelaziz Daaif, Omar Bouattane, Mohamed Youssfi and Sidi Mohamed Snineh, "Smart highways sensor network modeling: Real-time sensor fault detection", IJCSNS International Journal of Computer Science and Network Security, VOL.17 No.12, December 2017.