Predicting Covid-19 Infections With a Multi-Agent Organizational Approach and Machine Learning Techniques

Samir Safir and Abderrahim Siam
ICOSI Laboratory, Computer Science Department, Abbes Laghrour University, Khenchela BP 1252 El Houria, Algeria
E-mail: safir.samir@univ-khenchela.dz, siamabderrahim@gmail.com

Student paper

Keywords: Multi-agent system, machine learning, predictive analytics, healthcare, machine learning

Received: April 1, 2023

Our study presents a strategy for designing and implementing a Multi-Agent System (MAS) using organizational paradigms. The developed system offers a healthcare-oriented approach that utilizes the Internet of Medical Things (IoMT) to assist public health authorities in predicting COVID-19-infected patients. The proposed approach leverages autonomous agents to handle dynamic data from various sources within a structured organization. These agents collaborate to make effective, real-time predictions. As the agents continuously learn from the cases entering the system, the accuracy of predictions improves over time. The system was implemented using the JaCaMo framework, which integrates three key layers of MAS programming: organization, environment, and agent programming. The methodology demonstrated a prediction accuracy of over 90%, outperforming state-of-the-art (SOTA) approaches by enabling faster real-time decision-making. This capability facilitates the efficient processing of real-time big data, making a significant contribution to the advancement of predictive healthcare systems.

Povzetek: Razvit je sistem za napovedovanje okužb s COVID-19 z uporabo večagentnega sistema (MAS) in algoritmov strojnega učenja. Sistem omogoča natančne napovedi z analizo podatkov v realnem času, izboljšanih z učnimi agenti in IoMT.

1 Introduction

Artificial intelligence [1] is becoming increasingly important in healthcare and has the potential to revolutionize the way we diagnose, treat, and prevent diseases. In this paper, our aim is to develop an intelligent system to predict COVID-19 infection cases by involving a multitude of concepts, such as Multi-Agent Organization, IoT devices, Machine Learning algorithms, and Big Data analytics. This paper presents a novel approach that leverages these advancements to address the urgent need for accurate COVID-19 infection predictions.

Agent-Oriented Engineering is a widely recognized method for constructing distributed and complex software systems. This approach focuses on using autonomous, proactive agents as the key elements in the design and development process, making it well-suited for systems that operate in highly dynamic environments. Autonomy is a fundamental characteristic of agents in Multi-Agent systems (MAS). However, while the autonomy of agents can be beneficial in many contexts, it may also lead to challenges, such as dispersed behavior that prevents alignment with global objectives. Consequently, conventional Multi-Agent models frequently adopt an individualistic outlook towards the environment by treating agents as self-governing entities pursuing their objectives based on their perceptions and abilities. In critical applications, such as those found in business or government settings, it is essential to consider the behavior of the overall system.

The aim of Multi-Agent systems research is to understand how autonomous agents can collaborate to solve problems and create collective outcomes that cannot be achieved by each agent working alone [2]. To reduce the dispersing effects of agent autonomy, organizations offer solutions. A Multi-Agent Organization is a social entity composed of multiple agents, structured according to specific topologies and communication relationships. These agents work together to complete multiple tasks to fulfill the overall goal of the organization [3].

To attain global objectives in MAS, the independent behavior of individual agents may need to be regulated. This is where organizational models come into play, serving as a means of controlling agent behavior so that they can collaborate effectively to attain shared objectives [4]. These models can be divided into two perspectives [5]: Agent-Centered MAS, where the designer focuses on the behavior of individual agents and their interactions without considering the system's structure, and Organization-Centered MAS, which emphasizes the behavior of the system as a whole. From this perspective, the designer considers both the overall organizational structure and coordination patterns, as well as the individual actions of each agent.

By creating norms or guidelines, organizational abstraction facilitates the coordination of agents' local behavior and interactions with one another. Furthermore, this model enables agents to reason about the overall organizational structure and the behavior of other agents.

Several applications based on Multi-Agent systems that utilize Organization Theory have demonstrated its usefulness and proven successful in various fields, including simulation, e-commerce, network management, collective robotics, avionic mechanical design, traffic simulation, and more. These diverse applications underscore the versatility and effectiveness of multi-agent systems across various domains. Additionally, the organization in a Multi-Agent system is important to support adaptation to environmental changes. These changes may be addressed by transitioning from one organization to another (reorganization or self-organization), as evidenced by the significant amount of research in this area.

The Internet of Things (IoT) is a network of physical objects, devices, and sensors that are connected to the internet and can communicate and exchange data with other devices or systems. IoT technology enables the collection and analysis of large amounts of data, which can be used to improve operational efficiency, reduce costs, and enhance decision-making. More specifically, we are interested in the Internet of Medical Things (IoMT) [6], which refers to medical devices, sensors, and wearables that are connected to the internet and can exchange health-related data. This interconnectedness plays a pivotal role in enhancing healthcare delivery.

In the proposed approach, we combine the concepts and techniques presented above with Big Data Analytics and machine learning algorithms [7]. Big Data Analytics refers to the process of analyzing large and complex datasets to extract insights, patterns, and trends that can help inform decision-making. It involves using advanced tools and techniques to process and analyze data from various sources [1], including structured and unstructured data, to uncover meaningful insights.

With these foundational concepts established, the subsequent sections of this paper will detail the proposed approach and its implementation. Section 2 presents a motivating healthcare example, while Section 3 reviews related research across various fields. Section 4 compares our approach with existing methods from related work, and Section 5 focuses on the design of the proposed solution. Section 6 addresses the implementation and utilization of the developed approach, and Section 7 provides a comparison and discussion of results from each agent in the system. Finally, the "Conclusions and Future Work" section offers concluding remarks and potential directions for future research.

2 Motivation and overview

To demonstrate the importance of incorporating organizational perspectives into the design of Multi-Agent Systems (MAS), we present a scenario envisioning a solution for combating the Coronavirus pandemic. This study introduces an architecture for a Multi-Agent System that leverages machine learning algorithms to rapidly identify Covid-19-infected patients. Our approach is based on a real-time investigation system that collects physiological data from patients, including body temperature, ECG, heart rate, oxygen levels, blood pressure, glucose levels, and more.

The system relies on accurate information from hospitals connected to it, necessitating an efficient method for storing and processing large volumes of data. Big Data technology is utilized to digitally store comprehensive information on all Covid-19 cases, including those currently infected, recovered, or deceased. The stored data can be continuously analyzed to develop future preventive measures. The system applies a Multi-Agent Organizational model, enhanced with data analytics powered by machine learning (ML), to analyze the collected data and improve prediction models in real-time.

This solution aids local health authorities in monitoring a large number of users, promptly alerting them if symptoms are reported. Health officials can then reach out to the affected users, instructing them to report to the hospital for testing. Patients are admitted for observation until test results are confirmed. Additionally, the system tracks individuals in close contact with the infected patient, including family members, friends, and coworkers, and monitors them for any signs of infection.

By assisting health authorities in controlling the spread of Covid-19, this approach helps alleviate the burden on medical staff. Furthermore, the solution is adaptable for use in other hazardous pandemics or public health crises, offering a versatile and scalable tool for managing health emergencies.

3 Related work

As noted by Ilana et al. (2021) [8], the majority of AI research aimed at combating the coronavirus can be classified into four main categories: diagnosis and prognosis, treatments and vaccines, social control and tracking, and prediction. This paper focuses primarily on the latter category—prediction.

Otoom et al. (2020) [9] proposed a system for detecting and monitoring Covid-19 cases in real-time. During quarantine, IoT devices were deployed to gather real-time physiological data, and machine learning algorithms were used to enhance predictive accuracy. Their study compared seven machine learning algorithms, with five showing improvements in prediction accuracy. This demonstrates the effectiveness of using IoT in combination with machine learning for real-time pandemic response.

A survey by Thanh (2020) [10] analyzed various AI techniques applied to combat Covid-19, focusing on data analytics, natural language processing, and data mining. The work highlighted the diversity of AI applications in pandemic management, particularly in addressing key challenges such as big data processing and decision-making efficiency.

Carrillo et al.(2020)[11] employed unsupervised machine learning techniques, including k-means clustering, to classify countries based on similar Covid-19 infection patterns. This study emphasized the importance of using machine learning to understand the geographic spread of the virus and revealed significant insights into transmission trends across different regions.

Janko et al.(2021)[7] explored how non-countermeasure factors, such as culture, development, and travel, contributed to the early spread of Covid-19 before strict interventions were implemented. Using machine learning and statistical models, the study achieved approximately 80% prediction accuracy, showing how interconnected societal factors affect viral transmission in diverse contexts.

Recently, agent-based systems have emerged as a promising approach to addressing limitations in the healthcare sector. By integrating Multi-Agent Systems (MAS) into medical applications, healthcare costs can be reduced, and the burden on medical professionals can be alleviated. These systems shift the focus toward preventive, long-term care, which is patient-centered rather than hospital-centered. This transformation includes the use of remote monitoring systems that enable patients to play a more active role in managing their health and treatment, especially during extended care periods in both hospitals and homes.

Isern et al.(2016)[12] developed a MAS-based platform for managing patient care during hospital stays. This platform collects real-time data from various sources, facilitating dynamic bed occupancy allocation, doctor assignments, medical procedure planning, and automated billing by tracking the behavior of hospital actors in real time. The system demonstrates the potential of MAS to optimize healthcare operations and improve hospital management efficiency.

Lanzola et al.(1999)[13] proposed a framework for developing interoperable Multi-Agent Systems for medical applications. Their work highlighted the need for MAS in enhancing collaboration across diverse healthcare systems. Additionally, Juan et al.(2006) [14] in an Ambient Intelligence (AmI) ecosystem for Alzheimer patients, another study by Gonzalez et al.(2002)[15] developed BDI agents that integrate context-aware technologies to gather real-time data from users, further advancing patient-centered care.

4 Discussion

To summarize the key methodologies, results, and contributions from the aforementioned studies, we present a comparative table (Table .1). This table provides a clear overview of the different approaches to Covid-19 prediction, their key features, and their relative performances.

This table highlights how different methods approach COVID-19 prediction, focusing on collaboration efficiency, machine learning techniques, and accuracy. It also demonstrates the strength of our proposed MAS-based system, particularly its adaptability, collaboration-driven improvements, and real-time monitoring capabilities, which are critical for managing future pandemics and healthcare crises.

In summary, previous research demonstrates the effectiveness of AI and MAS in addressing various healthcare challenges, particularly in pandemic response and patient monitoring. Our study builds on these approaches by integrating Multi-Agent Organizational paradigms with IoMT devices and machine learning for real-time Covid-19 prediction, providing a novel solution that enhances predictive accuracy and supports public health efforts.

5 The proposed solution

Several studies conducted in the field of Multi-Agent learning have emphasized the importance of large datasets [16, 17]. Additionally, extensive research has been performed on various models, such as ensembles of classifiers [18, 19]. The goal of this research is to develop a predictive Covid-19 case detection system using a Multi-Agent approach based on an organizational model. The system operates in a distributed environment and is composed of different sites referred to as ”Assistant Controllers” (ACs). Each AC consists of two agents: a Learner Agent and an Interface Agent. It is crucial to note that different sites may have distinct instances of datasets. The knowledge generated by the independent ACs at each site will be consolidated into a single knowledge repository. The global system comprises of a collection of ACs and a special agent called a Broker agent or Mediator, as illustrated in (Fig. 1).

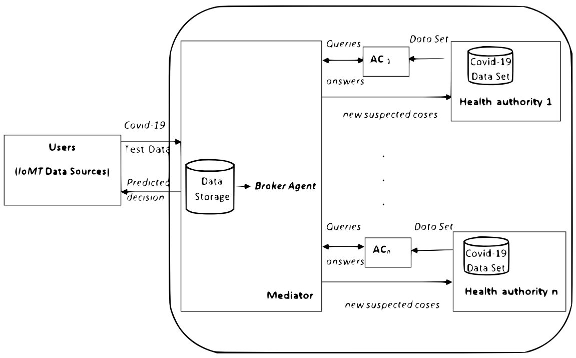

Figure 1: The global architecture of MAS organization Covid-19 approach

The agents communicate and share information about the selected datasets at each node. Subsequently, the Broker Agent selects the dataset that provides the most significant

Table 1: Performance comparison of predictive methods

Article Title	Key Metrics/Variables	Methods	Results/Findings
[9]:An IoT-based Framework for Early Identification and Monitoring of Covid-19 Cases	Accuracy, real-time health data	IoT, Ensemble Learning (RF GBTs), Apache Spark	Achieved over 90% accuracy with various AI algorithms; proposed a real-time framework for monitoring Covid-19 using IoMT and WBANs.
[11]:Using Country-Level Variables to Classify Countries According to Confirmed Covid-19 Cases	Covid-19 case counts, country variables	Unsupervised Machine Learning	Identified distinct clusters of countries based on Covid-19 case counts using unsupervised learning techniques.
[13]:A Framework for Building Cooperative Software Agents in Medical Applications	Collaboration efficiency, application areas	Framework Development, Agent-Based Modeling	Proposed a cooperative agent framework for healthcare applications to enhance collaboration among medical agents.
[14]:Intelligent Environment for Monitoring Alzheimer Patients, Agent Technology for Health Care.	Patient monitoring metrics, AI integration	Agent Technology, Sensor Networks	Developed an intelligent monitoring environment for Alzheimer's patients to enhance care and support for families.
[7]:Machine Learning for Analyzing Non-Countermeasure Factors Affecting Early Spread of Covid-19	Spread factors, machine learning accuracy	Machine Learning, Data Analysis	Analyzed non-countermeasure factors affecting Covid-19 to spread; developed predictive models with promising accuracy 80% in identifying influential factors.
The proposed approche	Collaboration efficiency ,Decision-making efficiency , machine learning accuracy , real-time health data	Multi Agent based on an Organizational Modeling, Incremental Machine & ensemble learning Techniques	Support for Remote Care, Real-time Monitoring and Alerts, Improved Decision-Making, Customizable for Future Pandemics

information. After this, the prediction decisions are sent to health authorities and users for preventive action (see Fig. 2). In our proposed approach, the agents work together within a structured organization to coordinate their behavior and cooperate to achieve the global objective of improving Covid-19 predictions.

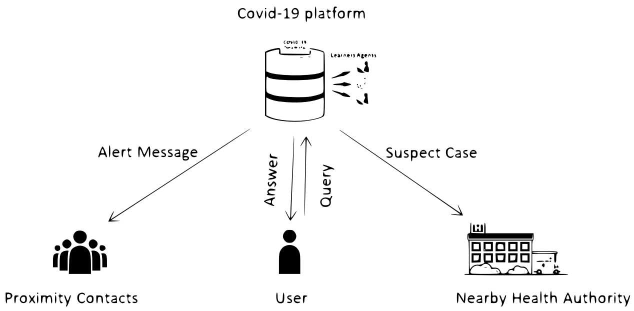

Figure 2: The global architecture of the MAS organizational approach to Covid-19 prediction

5.1 Assistant controller AC

An Assistant Controller (AC) is a Multi-Agent System (MAS) designed to categorize and predict Covid-19 patients using streaming datasets provided by health authorities. The structure of the AC is illustrated in Fig. 3. The Assistant Controller (AC) operates within the framework of the Agent and Artifact (A&A) paradigm, as discussed in Section 5.2.3. This paradigm serves as a conceptual model that outlines the interactions between autonomous agents and their environment, facilitating effective communication and collaboration.

At the heart of the AC's operation are two primary agents:

1. *Interface Agent:* This agent is responsible for capturing every incoming dataset at the Assistant Controller (AC) and processing bid queries from the Broker Agent, ensuring that relevant data is available for decision-making. Upon receiving new data or queries, the Interface Agent immediately notifies the Learner Agent by updating its environment through the Agent

and Artifact (A&A) mechanism, where artifacts act as intermediaries to facilitate communication, coordination, and collaboration among agents. This process initiates the prediction process, ensuring continuous and adaptive updates to the predictive model.

2. *Learner Agent:* This agent plays a crucial role in updating the predictive model. Upon receiving notifications from the Interface Agent, the Learner Agent incrementally updates the model based on the incoming data. It can also predict the queried bids as the expected results. This capability allows the Learner Agent to refine its predictions continuously, ensuring that the model remains current and relevant.

The real-time coordination between the Interface Agent and the Learner Agent enables the AC to continuously rebuild and dynamically update the predictive model. As a result, our system effectively manages highly dynamic healthcare datasets, adapting seamlessly to new information as it becomes available. This flexibility is essential in a rapidly changing context like Covid-19, where timely and accurate predictions can significantly impact public health responses.

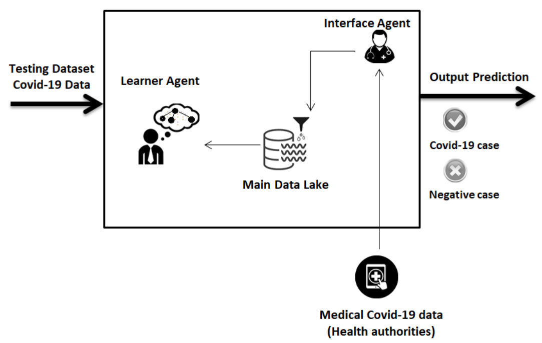

Figure 3: Assistant Controller architecture

5.1.1 Interface agent

The Interface Agent plays a crucial role in cleaning, standardizing, and transforming data to produce high-quality datasets. Preparing these training datasets is essential for machine learning and statistical analysis, as it allows a computer to acquire knowledge and effectively learn problem-solving techniques. Both machine learning models and statistical analyses rely heavily on these well-prepared datasets.

Since datasets may contain erroneous or missing information, data cleaning is necessary to remove or correct these issues. The data is partitioned into separate sets for testing and training purposes, with the machine learning (ML) model being trained on 75% of the dataset. The efficiency of the proposed solution is then evaluated using the remaining 25%. Additionally, the Interface Agent serves as a connection point between the Learner Agent and the Broker Agent, facilitating seamless communication and coordination within the system.

Medical data streams: Whenever a new Covid-19 case is discovered, it must be reported promptly to the relevant Assistant Controller system. This allows the intelligent learning agents to update the model using only a single iteration through the data, leading to better and more accurate real-time predictions.

5.1.2 Learner agent

The Learner Agent has two vital roles to fulfill. The first function is responsible for creating the training model, referred to as the Classify function. The second function is tasked with making predictions, known as the Predict function. This function utilizes the model to predict unknown outcomes. Notably, the goals of these two missions can be achieved in parallel, leveraging the strengths of a Multi-Agent System (MAS) and incremental machine learning techniques.

Supervised machine learning algorithms include a classification type, where a model is developed from a set of labeled training data. As a result, the model can predict the class or label for newly discovered data, known as testing data. Metrics such as accuracy, which are calculated based on the testing results, can be used to evaluate classification performance . Numerous classification algorithms exist, including decision trees, random forests, neural networks, rule-based algorithms (such as conjunctive rules and PRISM), logistic regression, naive Bayes, and others. Each algorithm has its distinct advantages and limitations, and the selection of an appropriate algorithm depends on various factors, including the data's attributes and the desired objectives [20].

Decision trees are highly regarded classification techniques in the field of data mining and have applications across various domains, including business intelligence, biomedicine, and healthcare. The conventional method of creating a decision tree is known as the Greedy Search approach. This process involves loading the complete dataset into memory and organizing it into a series of nodes and leaves that form a hierarchical structure. However, a significant drawback of this method is that once the decision tree has been constructed, it cannot be easily modified or updated, even in the presence of new data. Incorporating newly acquired information necessitates the complete reconstruction of the tree by loading both historical and recent data, a process that may be time-consuming and computationally intensive. Additionally, this approach risks losing critical information or patterns that were present in the original tree. To address these limitations, various alternative methods for decision-tree learning have been developed to enhance the scalability and adaptability of decision trees to changing data [21].

The conventional approach is inadequate for handling limitless data inputs, such as data streams, where information arrives in real-time. To tackle this challenge, an incremental approach has been introduced, enabling dynamic model construction so that the tree expands as new data is inputted. The Very Fast Decision-Tree (VFDT) algorithm, proposed by [22], employs the Hoeffding Bound [23] for incremental node splitting. This approach constructs a decision tree by continuously observing the features of incoming data and maintaining a record of their statistics. Once sufficient statistics are collected from every leaf, a node-splitting algorithm is applied to determine whether there is adequate statistical evidence to justify a node split. If a node split is warranted, the tree is expanded, and a new decision node is added in place of the leaf. This technique allows the decision tree to learn continuously as the data stream is processed in real-time [24]. The primary advantage of this approach is its ability for real-time data mining, eliminating the need to store all data in advance since data streams are potentially unlimited. Unlike traditional decision trees that require retraining with every new data arrival, the incremental approach allows for dynamic updates to the tree, enabling more efficient and effective data analysis. The process of building a VFDT is demonstrated in Algorithm 1 [25]. Notably, node divisions across multiple leaves can occur concurrently and autonomously [26].

While several libraries in the literature support incremental learning—particularly for the Very Fast Decision Tree (VFDT), such as MOA, Apache Spark MLlib, Scikit-Multiflow, and River—we have chosen WEKA for the following reasons:

1. *Compatibility with Java Eclipse:* WEKA is fully compatible with Java, allowing for seamless integration with the Java Eclipse IDE. Its rich package and API facilitate the easy incorporation of WEKA's machine learning capabilities into larger Java applications, making it a versatile tool for implementing machine learning within the JaCaMo framework.
2. *User-Friendly Interface:* WEKA provides an intuitive interface for data preprocessing, model evaluation, and visualization. This streamlines the machine learning workflow, making it accessible for researchers who are new to the field.
3. *Community Support and Documentation:* WEKA boasts a strong community and extensive documentation, which facilitate troubleshooting and ensure the availability of resources for further experimentation.
4. *Suitability for Multi-Agent Systems:* WEKA's capabilities align well with the requirements of our multi-agent system, particularly in handling large datasets, performing real-time predictions, and integrating seamlessly with the JaCaMo framework.
5. *Runtime Metrics:* We will present detailed runtime metrics for WEKA's machine learning tools to demonstrate their efficiency, including average training and testing times for various algorithms implemented within WEKA, such as Random Forest, VFDT, and others utilized in our study.

We selected the Very Fast Decision Tree (VFDT) algorithm for several compelling reasons relevant to healthcare applications. First, VFDT is designed for high scalability [27], enabling it to efficiently manage large-scale data streams typical in healthcare settings. Unlike ensemble-based approaches, which often require full retraining, VFDT allows for incremental updates, making it well-suited for the rapid influx of patient data. Additionally, VFDT's real-time adaptation capabilities [28] ensure that the model can quickly adjust to dynamic changes in healthcare datasets. Finally, its memory efficiency [29] enables effective resource management, which is crucial in healthcare environments where computational power may be limited. Thus, the combination of efficiency, adaptability, and memory management makes VFDT a more suitable choice for our predictive modeling needs compared to ensemble-based incremental algorithms.

Algorithm 1: Very Fast Decision Tree Induction

Input: S: A sequence of examples; X: The set of attributes; γ: One minus the desired probability; ε: The Hoeffding bound
Output: τ : very fast decision tree learned from S

Initialize τ with a single root node.
Initialize the statistics for tree growth.
foreach $s \in S$ **do**
  Sort s into leaf node l using τ.
  Update the statistics at l for tree growth.
  if *Examples at l are not from the same class* **then**
    foreach *attribute* $X_a \in X$ **do**
      Calculate the Hoeffding bound using the formula: $\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2|S|}}$ *(where R is the range of the attribute values)*;
      Select the attribute with the highest Hoeffding bound and split l on that attribute.
      if *multiple attributes have the same highest Hoeffding bound* **then**
        select the one with the highest information gain.

5.2 Broker agent

The system continuously receives incoming queries related to Covid-19 testing datasets for prediction. To facilitate communication between the collection of Assistant Controller (AC) agents and external entities, an intermediary agent known as the Broker Agent is employed. The Broker Agent is responsible for mediating interactions between the AC agents and the external environment.

Once it receives the queries, the Broker Agent utilizes Random Forest techniques to select the most appropriate predictions based on the aggregated data from the AC agents. This approach ensures that the predictions are robust and take advantage of the diverse insights provided by the various AC agents.

5.2.1 Random forest

The Random Forest algorithm is an ensemble learning method that involves building multiple decision trees and combining their predictions to enhance accuracy. Each decision tree is constructed using a randomly selected subset of the training data and a random subset of features. The final prediction is determined by averaging the predictions from all the trees in the forest. This approach helps reduce overfitting and improves the overall accuracy of the model. Based on the principles of Random Forest classification [30], the proposed approach involves selecting the most relevant majority vote decision from various Assistant Controller (AC) groups. This method operates under the assumption that different AC groups may have access to different sources of information and may employ distinct decision-making criteria. By combining the decisions from multiple AC groups, we can leverage their diverse perspectives and expertise to arrive at a more informed and accurate conclusion.

We chose Random Forest as the primary decision-making technique at the broker agent level due to its strong performance in managing complex, high-dimensional datasets typical of healthcare applications. Although Random Forest presents challenges in interpretability—especially when dealing with distinct datasets from each AC—its robustness and ability to handle missing data (particularly when one AC does not respond), along with its capacity to provide actionable insights, make it a strong candidate for our predictive modeling efforts in the healthcare domain.

To implement this approach, we first need to identify the relevant AC groups that can contribute input to the decision-making process. Various criteria can be employed to select these groups, including their area of expertise, past performance, or level of authority.

Once the relevant AC groups are identified, we can gather their decisions and use a voting mechanism to determine the most pertinent majority decision. This can involve assigning weights to the decisions based on the expertise or performance of each group or utilizing a more complex algorithm to combine decisions in a meaningful manner.

Overall, this approach has the potential to improve the accuracy and robustness of the decision-making process by leveraging the collective intelligence of multiple AC groups. However, it also requires careful coordination and communication between the different groups to ensure that their decisions are aligned and consistent with the overall goals and objectives of the organization.

5.2.2 Contract Net

To prevent potential bottlenecks associated with the Broker Agent, the system employs the Contract Net Protocol, originally proposed by [31] and later refined by [32]. This protocol is designed to facilitate the announcement of transferable tasks and to solicit bids from Interface Agents representing each Assistant Controller (AC) group capable of executing these tasks.

When a task is announced, the Interface Agents respond with bids that indicate their perceived capability to fulfill the task. The Broker Agent collects these bids and allocates the task to the bidder with the highest offer. Although the Contract Net Protocol is often viewed as a negotiation technique, its primary function is to act as a coordination mechanism for task allocation.

Task Announcement and Bid Solicitation: When a specific task arises—such as processing a new Covid-19 dataset for prediction—the Broker Agent announces this task to the relevant AC groups within the system. Each Interface Agent receives the announcement and evaluates its own capabilities, resources, and current workload before deciding whether to respond with a bid. The bid submitted by each Interface Agent includes not only an estimate of the time and resources required to complete the task but also their confidence level in executing it successfully.

Bid Evaluation and Task Allocation: Once the Broker Agent has received all the bids from the Interface Agents, it conducts a thorough evaluation to determine which bid offers the best combination of capability and cost. The Broker Agent allocates the task to the bidder with the highest offer, taking into account the quality of the bid and the capabilities of the Interface Agent. This allocation process ensures that tasks are assigned to agents that are most likely to perform them effectively and efficiently.

Coordination and Flexibility: Although the Contract Net Protocol is often viewed primarily as a negotiation technique, its primary function is to serve as a coordination mechanism for task allocation. By allowing dynamic task allocation and enabling agents to bid for multiple tasks simultaneously, the protocol enhances the system's flexibility and adaptability. This is particularly important in a healthcare setting where the demands on the system can vary significantly and unpredictably.

Workload Balancing The protocol also promotes workload balancing within the system. Since agents that are already busy with other tasks are less likely to place bids, the distribution of tasks tends to be more even across the available agents. This self-regulating mechanism helps to prevent any single agent from becoming overwhelmed with too many tasks, thereby maintaining overall system performance.

Limitations of the Contract Net Protocol: However, the Contract Net Protocol does have its limitations:

1. *Conflict Detection and Resolution:* One notable drawback is that the protocol lacks built-in mechanisms for conflict detection and resolution. If two agents bid for the same task or if their bids conflict in terms of resource availability, the protocol does not provide a framework for resolving these conflicts, which may lead to inefficiencies or delays.
2. *Heavy Reliance on Communication:* The protocol heavily relies on communication between agents. This reliance can introduce delays or inefficiencies in task allocation, especially in scenarios where network la-

tency or communication failures occur. If the communication channels are congested or disrupted, it could hinder the timely allocation of tasks.

3. *Scalability Concerns:* In larger systems with numerous agents and tasks, the process of soliciting bids and evaluating them can become computationally expensive and time-consuming. This could potentially lead to performance bottlenecks as the system scales.
4. *Limited Bidder Pool:* The effectiveness of the protocol is contingent upon having a sufficient number of agents willing and able to submit bids. If too few agents respond to a task announcement, it could result in suboptimal task allocation.

Overall, while the Contract Net Protocol offers a structured and efficient means of task allocation within the Broker Agent framework, it is important to be aware of these limitations and to consider supplementary mechanisms or strategies that could mitigate potential issues and enhance the overall robustness of the system.

5.2.3 Agents and artifacts (A&A) environment

Our proposal involves utilizing the Agents and Artifacts meta-model to create a coordination mechanism for developing a shared environment, as described by [33]. This model addresses some of the shortcomings of the Contract Net Protocol by representing the environment as workspaces, where Agents and Artifacts coexist to provide various services. In this environment, agents interact with artifacts through a mechanism known as the focus action. When an agent focuses on an artifact, it gains access to the observable properties of that artifact as perceptions. Additionally, artifacts can offer a range of operations or actions that agents can perform, enhancing collaboration and interaction within the system. *Bidding Process for Task Allocation:* The Broker Agent plays a crucial role in locating Interface Agent contractors for each Assistant Controller (AC) group (Fig. 4). This is achieved through a structured bidding process, which follows these steps:

1. The Broker Agent announces the task (the task is considered an artifact in the system).
2. Interface Agents evaluate the task based on their capabilities and commitments.
3. Interface Agents submit bids to the Broker Agent.
4. The Broker Agent evaluates the received bids and selects the majority vote from the competing predictions.

Decentralized and Collaborative Approach: This bidding scheme is completely distributed, meaning that each agent operates independently without a centralized controller. The Broker Agent facilitates the announcement of tasks and the collection of bids, but the decision-making process relies on the collective actions of multiple agents. Each Interface Agent autonomously assesses the task based on its own capabilities and commitments, leading to a diverse range of bids that reflect varying perspectives. The final decision regarding task allocation is made based on the majority of predictions from the Interface Agents. This approach not only ensures a decentralized and collaborative handling of tasks but also leverages the diverse expertise of multiple agents to enhance the accuracy and reliability of the outcomes.

By employing the Agents and Artifacts framework, we create a flexible and adaptive environment that can effectively manage the dynamic and complex nature of healthcare data processing tasks. The interaction between agents and artifacts allows for improved coordination, resource utilization, and task execution in the context of predicting Covid-19 patient outcomes.

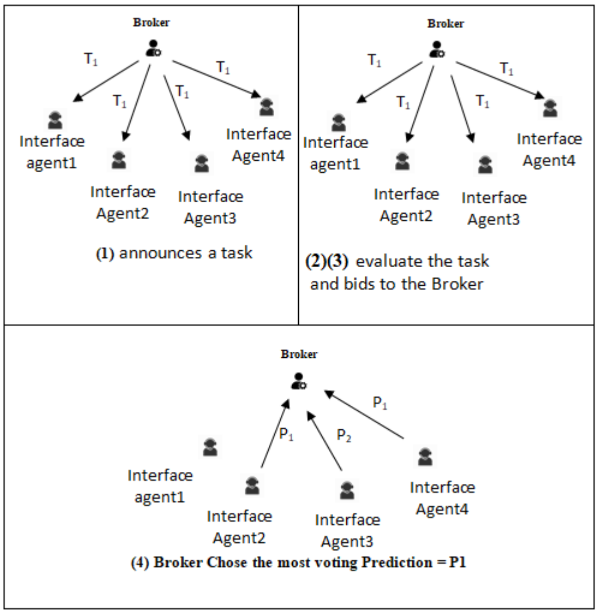

Figure 4: Broker agent

6 Some implementation aspects

The integration process described in the previous sections was implemented using the Eclipse Modeling and JaCaMo framework. The JaCaMo[1] framework is a multi-agent-oriented programming platform (MAOP) designed to facilitate the development of complex multi-agent systems [34]. This framework utilizes autonomous agents programmed in Jason[2] , which operate within environments defined as artifacts in CArtAgO [3]. Additionally, these agents are organized by the Moise framework [4], and they interact with each other using a specified interaction language.

All experiments were conducted on a system equipped with an Intel Core i5-1035 G1 processor running at 1.00 GHz and 16 GB of RAM. The software application Alert-Covid employs the proposed model, consisting of three Assistant Controllers named UK, Biskra, and ESI. The Broker

[1] http://jacamo.sourceforge.net
[2] http://jason.sourceforge.net/wp
[3] http://cartago.sourceforge.net
[4] http://moise.sourceforge.net/

Agent serves as the intermediary, facilitating communication and coordination among the different Assistant Controllers.

6.1 UK assistant controller

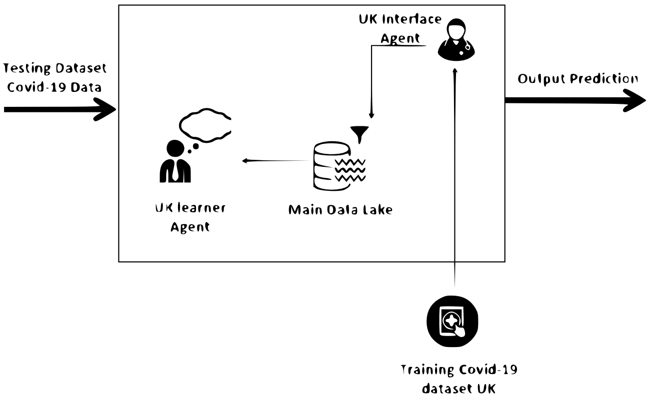

Figure 5: UK architecture

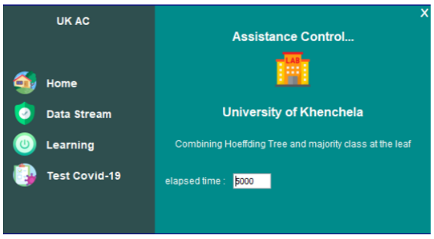

Figure 6: UK interface

6.1.1 UK interface agent

For the training data at the UK Assistant Controller site (Fig 7), we utilize the dataset titled Covid-19 Symptoms and Presence, which is publicly available on Kaggle [35]. This dataset comprises 20 attributes that represent potential factors associated with contracting Covid-19, including symptoms such as cough and fever, along with other relevant indicators. Additionally, it contains a class attribute indicating the presence of the virus.

The dataset is sourced from reputable organizations, including the World Health Organization and the All India Institute of Medical Sciences, with the aim of facilitating research into the prevalence of symptoms and their correlation with Covid-19 diagnoses. Its comprehensive nature makes it a valuable resource for predictive modeling in the healthcare domain.

6.1.2 UK learner agent

The UK Assistant Controller's learning agent implement the Hoeffding tree algorithm, which is integrated into the open-source software WEKA, to obtain knowledge from

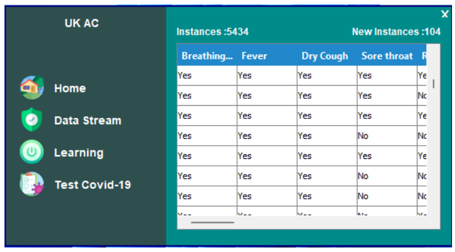

Figure 7: UK interface agent

data streams. WEKA is a suite of machine learning algorithms developed at the University of Waikato in New Zealand [36]. It facilitates data mining by providing a wide range of tools for clustering, pre-processing, visualization, association, regression, and classification.

The VFDT implementation in the UK learner agent uses the following specific parameters Fig. 8:

1. *Splitting Criterion:* Gini Index, which is a measure of the impurity or variance of a node. The algorithm selects the attribute with the lowest Gini Index to split the node, leading to purer child nodes.
2. *Allowable Error δ:* $1 \cdot 10^{-7}$.This parameter controls the confidence level for deciding whether to split a node. A lower error threshold ensures higher confidence in the decision to split, minimizing the likelihood of unnecessary splits, which helps in maintaining the efficiency of the model.
3. *Grace Period:* 100 instances. This setting ensures that the algorithm waits for 100 instances before evaluating a potential split at any node. This helps to accumulate sufficient data to make statistically sound splitting decisions, avoiding premature splits based on insufficient data.

These parameters help the UK learner agent achieve a balance between accuracy and efficiency, making it capable of handling continuous data streams while ensuring the decision tree adapts quickly to new data without overfitting.

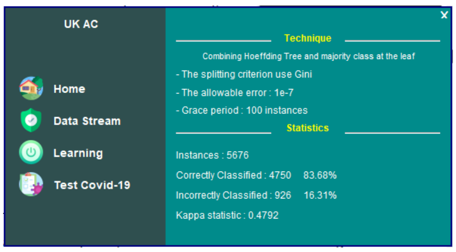

Figure 8: UK learner agent

A learning curve plots accuracy against the number of instances, as illustrated in (Fig. 9), showing how the model's performance improves with more training data.

- The accuracy starts at 81.42% at 5602 instances and fluctuates slightly as more data is processed.
- Over time, there is a general upward trend, with accuracy reaching 93.65% at 1982 instances.

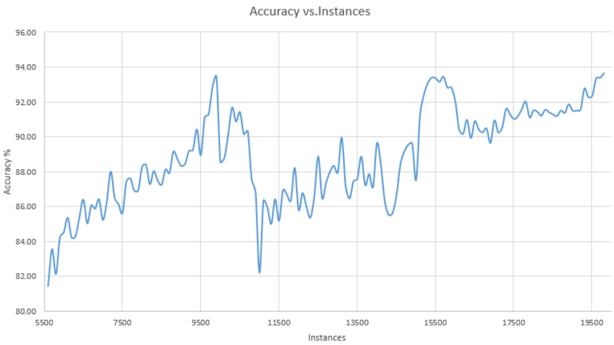

Figure 9: UK learner curve

– The model demonstrates significant improvement, achieving over 90% accuracy as it processes larger datasets.

Analysis:

– The learning curve likely depicts an overall positive trend with some small fluctuations. These fluctuations suggest that the model is learning from new data but may occasionally encounter challenges from certain instances.

– Despite these variations, the general increase in accuracy indicates the model's capacity to learn and improve its predictions as more instances are introduced.

The UK learner model shows a general improvement in accuracy as more data is processed, starting at 81.42% and reaching over 90%. Despite minor fluctuations, the overall trend is upward, indicating the model's ability to adapt and learn effectively from increasing data.

6.1.3 UK test Covid-19 case

With the implementation of our proposed architecture, we developed a GUI module for testing the UK Assistant Controller, enabling functional testing at the local level. Fig.10 presents an example of a positive Covid-19 case.

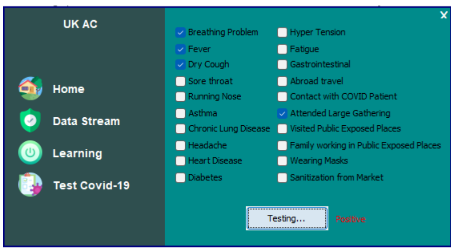

Figure 10: UK test Covid-19 case

6.2 ESI assistant controller

6.2.1 ESI interface agent

The ESI Interface Agent operates as a crucial component within the multi-agent system, specifically tasked with handling and processing COVID-19-related health data from a localized data stream. To enable effective COVID-19 prediction, the ESI Interface Agent utilizes a dataset sourced from a research repository [37]. This combined dataset, titled "Covid Symptoms," features 3,021,444 entries and ten essential characteristics related to COVID-19 symptoms and test outcomes. The dataset includes key information on symptoms such as cough, fever, shortness of breath, sore throat, and headache, which are critical indicators for COVID-19 detection.

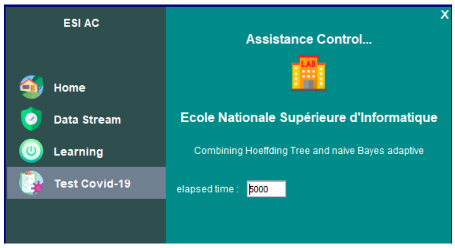

Figure 11: ESI interface

Given the dataset's comprehensive nature, the ESI Interface Agent processes and cleans the data to ensure its usability for predictive modeling. The cleaned and preprocessed data is then forwarded to the Learner Agent, which uses the Very Fast Decision Tree (VFDT) algorithm for incremental learning. This approach allows the agent to continuously update the prediction model in real time as new COVID-19 cases are added to the stream, without the need for retraining from scratch.

The choice of this dataset aligns perfectly with the goals of the ESI Interface Agent for several reasons:

1. *Rich Symptom Features:* The dataset contains detailed records of COVID-19 symptoms, allowing the ESI Interface Agent to make highly accurate predictions. By processing a diverse range of symptoms such as cough, fever, and sore throat, the agent can feed comprehensive input into the learning model.
2. *Large Scale Data:* With over 3 million entries, the dataset provides the necessary volume for evaluating the robustness and scalability of the system. This is especially useful in a real-world scenario where the ESI Interface Agent must process a continuous influx of new COVID-19 cases.
3. *Real-Time Updates:* The incremental learning method employed by the Learner Agent ensures that the ESI Interface Agent can handle real-time data without delays. The large dataset simulates real-world conditions, where updates to the model must happen quickly to reflect new information.
4. *Binary Class Labels:* The presence of a class attribute indicating a positive or negative COVID-19 test result is crucial for the classification tasks performed by the agent. This simplifies the prediction process, allowing for straightforward binary classification and evaluation of prediction accuracy.

In summary, the "Covid Symptoms" dataset not only aligns with the core tasks of the ESI Interface Agent but also ensures the agent can process high-volume data efficiently, update models in real time, and make accurate predictions based on symptom analysis. By incorporating this dataset into the system, the ESI Interface Agent plays a vital role in ensuring the overall success of the multi-agent COVID-19 prediction architecture.

6.3 Biskra assistant controller

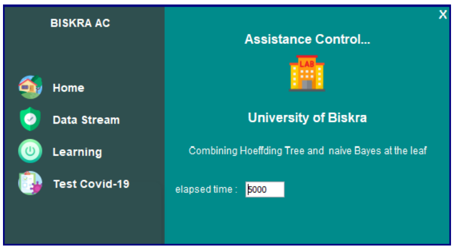

Figure 12: BISKRA interface

6.3.1 Biskra interface agent

The dataset titled 'Covid-19 Symptoms Checker,' sourced from Kaggle[38], contains 27 attributes related to Covid-19 symptoms and demographic information that may affect whether an individual has contracted the Coronavirus. Key attributes include:

– *Symptoms*: Fever, Tiredness, Dry Cough, Difficulty in Breathing, Sore Throat, and more.
– *Severity Levels*: Indicators for Mild, Moderate, Severe, and None.
– *Contact Information*: Details on contact with confirmed Covid-19 cases.
– *Demographics*: Gender (Male, Transgender) and Country.

This dataset is suitable for building a robust model to better predict Covid-19-infected patients for several reasons:

1. *Comprehensive Symptoms*: The dataset includes a wide array of Covid-19 symptoms, making it highly relevant for early detection and diagnosis in real-time applications.
2. *Severity Assessment*: Having labels for different severity levels helps in predicting the potential progression of the infection, allowing healthcare authorities to prioritize responses based on predicted outcomes.
3. *Contact History*: Information on whether individuals were in contact with confirmed cases significantly improves the predictive power of the model, aiding in identifying likely transmissions.
4. *Demographic Factors*: Attributes like gender and country allow for a detailed analysis of symptom presentation and the potential impact of demographics on infection susceptibility and disease outcomes.

6.4 The broker

The role of the Broker is to mediate between different Assistant Controllers (UK, ESI, Biskra) (Fig .13), ademonstrating the benefits of collaboration despite varying interests and goals. Autonomous and collaborative decision-making among the three Assistant Controllers is facilitated through the use of the Contract Net Interaction Protocol (Fig . 14).

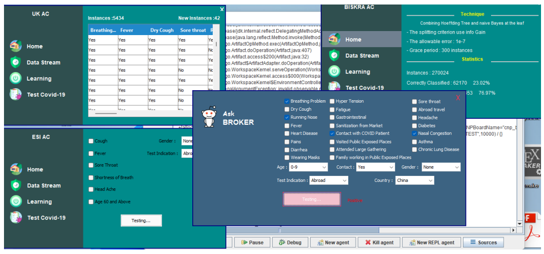

Figure 13: Broker interface

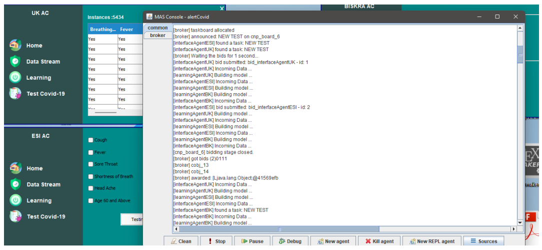

Figure 14: Broker use Contact Net protocol

7 Comparison and discussion

Fig .15 illustrates a comparison of the accuracy results from a portion of the dataset, starting from instance 7500 onward. This timeframe allows the system to build its model effectively. The comparison includes the UK, Biskra, and ESI agents, as well as the Broker Agent.

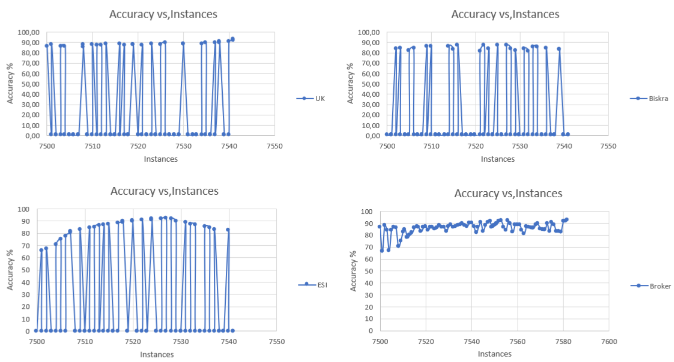

Figure 15: Broker acurracy

The performance of the UK, Biskra, and ESI ACs is summarized as follows:

- *UK AC*: Active in roughly half of the instances, with an accuracy ranging from 85% to 92% when involved.
- *Biskra AC*: Sporadically active, performing well when engaged, with an accuracy between 80% and 90%.
- *ESI AC*: The least active but consistently reliable when used, often achieving 85% or higher accuracy.

7.1 Broker performance highlights

- *Consistent Activity*: Unlike the individual agents, the broker is always active and never shows an accuracy of 0. It ensures that predictions are made even when some agents are inactive.
- *Accuracy Selection:*: The broker typically mirrors the highest-performing agent's accuracy, indicating that it selects or aggregates the best available prediction from the active agents.
- *Aggregation Role*: When multiple agents are active, the broker's performance reflects its ability to select the most accurate prediction or combine insights, ensuring higher accuracy.

7.2 Key observations

- *Broker Stability*: The broker's accuracy remains consistently strong, never dropping to 0, unlike the agents. This reliability makes the broker a crucial decision-maker, especially when certain agents are inactive.
- *Dependence on Agents*: The broker performs better when multiple agents contribute. When only one agent is active, the broker's accuracy mirrors that agent's performance. For example:
  - In instance 7541, only the UK agent contributes, resulting in the broker's accuracy being equal to the UK agent's at 92.86

The response times of the Broker Agent (in nanoseconds) across various instances are illustrated in (Fig .16). The broker's response times range from 5,600 ns to a peak of 113,700 ns in instance 7540, with another notable spike at 50,200 ns in instance 7574. These variations indicate occasional spikes in response times, possibly due to increased processing complexity or load.

The average response time of the Broker Agent is approximately 11,716 ns. Despite occasional delays, which may depend on the materials used, the broker maintains a relatively low average response time, indicating that it can efficiently manage predictions under typical conditions.

The broker plays a critical role in ensuring continuous and accurate predictions, achieving over 90% accuracy consistently over time. Utilizing the Random Forest technique, the broker consistently selects the best-performing agent's result, enhancing the overall system's reliability and performance. To further improve accuracy, increasing the participation of AC, would provide the broker with more data to work with and potentially boost overall performance.

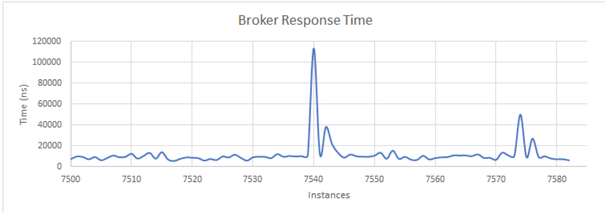

Figure 16: Broker reponse time

8 Conclusion and future work

In conclusion, this study successfully demonstrates the effectiveness of a multi-agent system (MAS) utilizing the Internet of Medical Things (IoMT) in predicting COVID-19-infected patients. The approach presented is grounded in an organizational structure within the MAS, defining the expected behaviors of agents collaborating towards a common healthcare objective. Through the implementation of experimental studies with various datasets, the proposed approach illustrates the contributions and benefits of organizational frameworks in MAS systems, achieving significant improvements in prediction accuracy, reaching over 90%.

Furthermore, we propose a flexible organizational model that allows for dynamic reorganization of the MAS in response to environmental changes. To enhance system performance, future work will focus on increasing the number of broker agents to mitigate potential bottlenecks. Additionally, rather than relying solely on random forests, we plan to explore alternative techniques for decision-making among different Assistant Controllers.

This research contributes to the growing body of knowledge in health informatics, offering viable solutions for public health authorities to monitor and respond effectively to pandemics. Future investigations could extend this model to other infectious diseases and further integrate it with existing healthcare infrastructures. The findings advocate for continued investment in smart health technologies that can transform healthcare delivery and improve patient outcomes.

References

[1] M. Gams and T. Kolenik, "Relations between electronics, artificial intelligence and information society through information society rules," *Electronics*, vol. 10, no. 4, p. 514, Feb. 2021. [Online]. Available: http://dx.doi.org/10.3390/electronics10040514

[2] S. Abderrahim and R. Maamri, "A category-theoretic approach to organization-based modeling of multi agent systems on the basis of collective phenomena and organizations in human societies," *Informatica*, vol. 42, no. 4, Apr. 2018. [Online]. Available: http://dx.doi.org/10.31449/inf.v42i4.1282

[3] J. Ferber and G. Weiss, *Multi-agent systems: an introduction to distributed artificial intelligence*. Addison-wesley Reading, 1999, vol. 1. [Online]. Available: https://dl.acm.org/doi/10.5555/520715

[4] C. M. Toledo, R. H. Bordini, O. Chiotti, and M. R. Galli, *Developing a Knowledge Management Multi-Agent System Using JaCaMo*. Springer Berlin Heidelberg, 2012, p. 41–57. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31915-0_3

[5] H. Ahmed Abbas, "Organization of multi-agent systems: An overview," *International Journal of Intelligent Information Systems*, vol. 4, no. 3, p. 46, 2015. [Online]. Available: http://dx.doi.org/10.11648/j.ijiis.20150403.11

[6] M. H. Al-Zubaidie and R. H. Razzaq, "Maintaining security of patient data by employing private blockchain and fog computing technologies based on internet of medical things," *Informatica*, vol. 48, no. 12, Sep. 2024. [Online]. Available: http://dx.doi.org/10.31449/inf.v48i12.6047

[7] V. Janko, G. Slapničar, E. Dovgan, N. Reščič, T. Kolenik, M. Gjoreski, M. Smerkol, M. Gams, and M. Luštrek, "Machine learning for analyzing non-countermeasure factors affecting early spread of covid-19," *International Journal of Environmental Research and Public Health*, vol. 18, no. 13, p. 6750, Jun. 2021. [Online]. Available: http://dx.doi.org/10.3390/ijerph18136750

[8] J. W. Ilana Harrus, "Artificial intelligence and covid-19: Applications and impact assessment," 2021. [Online]. Available: https://www.aaas.org/ai2

[9] M. Otoom, N. Otoum, M. A. Alzubaidi, Y. Etoom, and R. Banihani, "An iot-based framework for early identification and monitoring of covid-19 cases," *Biomedical Signal Processing and Control*, vol. 62, p. 102149, Sep. 2020. [Online]. Available: http://dx.doi.org/10.1016/j.bspc.2020.102149

[10] T. T. Nguyen, "Artificial intelligence in the battle against coronavirus (covid-19): A survey and future research directions," Aug. 2020. [Online]. Available: http://dx.doi.org/10.36227/techrxiv.12743933.v1

[11] R. M. Carrillo-Larco and M. Castillo-Cara, "Using country-level variables to classify countries according to the number of confirmed covid-19 cases: An unsupervised machine learning approach," *Wellcome Open Research*, vol. 5, p. 56, Jun. 2020. [Online]. Available: http://dx.doi.org/10.12688/wellcomeopenres.15819.3

[12] D. Isern and A. Moreno, "A systematic literature review of agents applied in healthcare," *Journal of Medical Systems*, vol. 40, no. 2, Nov. 2015. [Online]. Available: http://dx.doi.org/10.1007/s10916-015-0376-2

[13] G. Lanzola, L. Gatti, S. Falasconi, and M. Stefanelli, "A framework for building cooperative software agents in medical applications," *Artificial Intelligence in Medicine*, vol. 16, no. 3, p. 223–249, Jul. 1999. [Online]. Available: http://dx.doi.org/10.1016/s0933-3657(99)00008-1

[14] J. M. Corchado, J. Bajo, Y. de Paz, and D. I. Tapia, "Intelligent environment for monitoring alzheimer patients, agent technology for health care," *Decision Support Systems*, vol. 44, no. 2, p. 382–396, Jan. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.dss.2007.04.008

[15] M. González Bedia, J. M. Corchado Rodríguez *et al.*, "A planning strategy based on variational calculus for deliberative agents," 2002.

[16] A. Srivastava, E.-H. S. Han, V. Singh, and V. Kumar, "Parallel formulations of decision-tree classification algorithms," in *Proceedings. 1998 International Conference on Parallel Processing (Cat. No.98EX205)*, ser. ICPP-98. IEEE Comput. Soc, p. 237–244. [Online]. Available: http://dx.doi.org/10.1109/icpp.1998.708491

[17] S. Orlando, P. Palmerini, R. Perego, and F. Silvestri, *Scheduling High Performance Data Mining Tasks on a Data Grid Environment*. Springer Berlin Heidelberg, 2002, p. 375–384. [Online]. Available: http://dx.doi.org/10.1007/3-540-45706-2_49

[18] F. Provost and V. Kolluri, "A survey of methods for scaling up inductive algorithms," *Data Mining and Knowledge Discovery*, vol. 3, no. 2, pp. 131–169, 1999. [Online]. Available: http://dx.doi.org/10.1023/a:1009876119989

[19] C. Mastroianni, D. Talia, and P. Trunfio, "Managing heterogeneous resources in data mining applications on grids using xml-based metadata," in *Proceedings International Parallel and Distributed Processing Symposium*, ser. IPDPS-03. IEEE Comput. Soc, p. 11. [Online]. Available: http://dx.doi.org/10.1109/ipdps.2003.1213204

[20] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning - ICML '06*, ser. ICML '06. ACM Press, 2006, p. 161–168. [Online]. Available: http://dx.doi.org/10.1145/1143844.1143865

[21] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, p. 81–106, Mar. 1986. [Online]. Available: http://dx.doi.org/10.1007/bf00116251

[22] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the sixth ACM*

SIGKDD international conference on Knowledge discovery and data mining, ser. KDD00. ACM, Aug. 2000, p. 71–80. [Online]. Available: http://dx.doi.org/10.1145/347090.347107

[23] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association*, vol. 58, no. 301, p. 13, Mar. 1963. [Online]. Available: http://dx.doi.org/10.2307/2282952

[24] H. Yang and S. Fong, "Incremental optimization mechanism for constructing a decision tree in data stream mining," *Mathematical Problems in Engineering*, vol. 2013, p. 1–14, 2013. [Online]. Available: http://dx.doi.org/10.1155/2013/580397

[25] A. Das, J. Wang, S. M. Gandhi, J. Lee, W. Wang, and C. Zaniolo, "Learn smart with less: Building better online decision trees with fewer training examples," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, ser. IJCAI-2019. International Joint Conferences on Artificial Intelligence Organization, Aug. 2019, p. 2209–2215. [Online]. Available: http://dx.doi.org/10.24963/ijcai.2019/306

[26] R. Jin and G. Agrawal, "Communication and memory efficient parallel decision tree construction," in *Proceedings of the 2003 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, May 2003. [Online]. Available: http://dx.doi.org/10.1137/1.9781611972733.11

[27] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà, "New ensemble methods for evolving data streams," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD09. ACM, Jun. 2009. [Online]. Available: http://dx.doi.org/10.1145/1557019.1557041

[28] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," in *Proceedings of the 2007 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, Apr. 2007. [Online]. Available: http://dx.doi.org/10.1137/1.9781611972771.42

[29] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, p. 1–37, Mar. 2014. [Online]. Available: http://dx.doi.org/10.1145/2523813

[30] A. Liaw and M. Wiener, "Classification and Regression by randomForest," *R news*, vol. 2, no. December, pp. 18–22, 2002. [Online]. Available: https://api.semanticscholar.org/CorpusID:3093707

[31] R. G. SMITH, *The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver*. Elsevier, 1988, p. 357–366. [Online]. Available: http://dx.doi.org/10.1016/b978-0-934613-63-7.50039-5

[32] R. Davis, R. G. Smith, and L. Erman, *Negotiation as a Metaphor for Distributed Problem Solving*. Elsevier, 1988, p. 333–356. [Online]. Available: http://dx.doi.org/10.1016/b978-0-934613-63-7.50038-3

[33] A. Omicini, A. Ricci, and M. Viroli, "Artifacts in the a&a a meta-model for multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 17, no. 3, p. 432–456, May 2008. [Online]. Available: http://dx.doi.org/10.1007/s10458-008-9053-x

[34] O. Boissier, R. H. Bordini, J. F. Hübner, A. Ricci, and A. Santi, "Multi-agent oriented programming with jacamo," *Science of Computer Programming*, vol. 78, no. 6, p. 747–761, Jun. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.scico.2011.10.004

[35] H. Hari, "Symptoms and covid presence (may 2020 data)," Aug 2020. [Online]. Available: https://www.kaggle.com/hemanthhari/symptoms-and-covid-presence

[36] W. The Departments of Computer Science, "Weka 3: Machine learning software in java," 2002. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/

[37] Nshomron, "Nshomron/covidpred: Machine learning-based prediction of covid-19 diagnosis based on symptoms." [Online]. Available: https://github.com/nshomron/covidpred

[38] B. Hungund, "Covid-19 symptoms checker," Mar 2020. [Online]. Available: https://www.kaggle.com/iamhungundji/covid19-symptoms-checker