applied sciences

MDPI

Communication

Adaptive-Modeling Multi-Agent Learning System for Video Behavioral Clustering Recognition

Xingyu Qian [1], Aximu Yuemaier [2], Wenchi Yang [3], Xiaogang Chen [1,*], Shunfen Li [1], Weibang Dai [1] and Zhitang Song [1]

[1] State Key Laboratory of Functional Materials for Informatics, Shanghai Institute of Micro-System and Information Technology, Chinese Academy of Sciences, Shanghai 200050, China; qianxy@mail.sim.ac.cn (X.Q.)
[2] School of Physical Science and Technology, Shanghaitech University, Shanghai 201210, China
[3] NeuHelium Co., Ltd., Shanghai 200050, China
* Correspondence: chenxg@mail.sim.ac.cn

Featured Application: This work combines a decentralized multi-agent system working form, using a similar approach to associative memory, focusing on the dynamic association of motion processes, to achieve clustered recognition of motion behavior by designing a multilayer structured multi-agent system.

Abstract: Multi-agent systems are suitable for handling complex problems due to their high parallelism and autonomous evolution ability. In this paper, we propose an adaptive clustering multi-agent learning system for intelligent applications with continuously changing requirements. Each agent model changes temporal sequences using the longest common subsequence (LCS) algorithm. Multiple agents collaborate in a multilayer decentralized approach to enhance learning adaptability and achieve self-supervised behavioral clustering. The system is constructed using a "memory-like" method and operates primarily on memory access and comparison, avoiding extensive matrix operations of artificial neural networks while achieving learning and prediction functions. We chose an unsupervised vehicle behavioral clustering scenario for feasibility validation in which the system's cognitive objective is to cluster and recognize vehicle behaviors. In a low computational environment, the system can complete clustering functions and exhibit continuous learning capabilities when new behavioral changes occur. The proposed approach achieves an accuracy of 97.4% while processing at a speed 1–5 times faster than similar clustering algorithms. The verification results indicate that this system has excellent potential to enhance intelligent sensing front ends.

Keywords: continuous learning; multi-agent system; prediction; adaptability; clustering

check for updates

Citation: Qian, X.; Yuemaier, A.; Yang, W.; Chen, X.; Li, S.; Dai, W.; Song, Z. Adaptive-Modeling Multi-Agent Learning System for Video Behavioral Clustering Recognition. *Appl. Sci.* **2023**, *13*, 7486. https://doi.org/10.3390/app13137486

Academic Editors: Vincent A. Cicirello and Byung-Gyu Kim

Received: 23 May 2023
Revised: 12 June 2023
Accepted: 22 June 2023
Published: 25 June 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Multi-agent systems are extremely adaptable in dealing with complex problems. These systems have since been combined with reinforcement learning algorithms, enabling even more research progress [1,2]. Independent training approaches, such as independent Q-learning (IQL) [3], suffer from difficulty in achieving learning convergence due to the shared dynamic instability environment factors, while centralized execution, such as value-decomposition networks (VDNs) [4], integrates the value function of each agent, improving performance by ignoring individual characteristics to a greater extent. Most multi-agent systems use the centralized training and decentralized execution (CTED) approach, such as MADDPG [5] and QMIX [6]. However, centralized training usually evaluates the agents and computational tasks based on the system goal in a multi-agent system. In some cases, learning takes place in a simulator or laboratory [7], where agents have access to additional state information and can communicate freely through global data sharing.

In contrast, the decentralized execution approach uses a strategy of decentralizing the computational tasks, where each agent performs its learning process with local data and the environment. The decentralized approach also reduces the complexity of interaction with the increased number of agents and avoids partial observability challenges. Therefore, finding an effective decentralized execution strategy is important for both training and execution.

In addition, the process of changing the visual information of the video is always continuous and connected to the temporal task. If the frame is divided into several regions, the changes in different regions over time will show a clear regularity. The regularities of these changes can be summarized in higher dimensions. Therefore, the frame's change can be summarized in higher dimensions by focusing on the dynamic situations of many small regions. The law enables encoding of changes in different regional pictures to obtain temporally related but different temporal encoding sequences. Real-time clustering recognition requires neighboring regions at the same level to form a new sequence based on discrete temporal sequences (DTSs). A multilayer structure for clustering recognition can cover the global video frame and learn from the summary of small change patterns to achieve global behavioral recognition.

A multi-agent system is a computing system comprising multiple autonomous entities that interact with each other to achieve certain goals. Each agent possesses its own objectives, knowledge, and decision-making capability and can effectively handle parallel tasks. More importantly, clustering of video behavior would be a very complex task if only regular modeling was used, whereas a multi-agent approach can segment the frame to effectively reduce the computational complexity of each agent, and using a multilayer agent system can cover the entire frame, achieving learning effects from the recognition of small regions to global behavioral clustering. Thus, based on the decentralization strategy and the continuity of the encoding of movement information, by using traditional methods rather than the neural network or reinforcement learning, this paper proposes a memory-like adaptive-modeling multi-agent learning system, and it makes the main contributions that follow:

1. The design system handles video-level behavioral recognition, encoding video information as a discrete temporal sequence as a means of processing data. The design uses a multilayer structure, and the clustering results of agents form a new DTS at higher-layer agents, forming a self-organizing structural pattern in the bottom-up direction;
2. The design agents all use prediction as the evaluation criterion, so they are only concerned with local performance. The top-layer agent only cares about its own execution, but it also represents the performance of the system, and its clustering result is expressed as the recognition result of the video behavior;
3. The dynamic model-building method helps in agent comparison calculations to achieve pattern recognition or behavioral clustering, and the ability to maintain the matched models based on matching degree fusion sequence characteristics helps to achieve continuous learning with online updates.

The choice of a traditional algorithmic solution to design a decentralized multi-agent system rather than using artificial neural networks means that there is no unified standard for evaluating the execution state of the agents for centralized management. The real-time changing nature of the temporal sequence represents a continuous process of movement state occurrence, and the processing logic of individual agents is reduced to the act of clustering the temporal sequence for identification and change prediction. This clustering can be passed on as output in the connected structure of the system based on the clustering results of the temporal sequence. The theoretical level of a system formed by the combined connection of agents allows for a process of clustered cognition of complex computational behavior through the predictive drive mechanism of each agent.

In this study, vehicle footage captured from webcams was used as a validation dataset for unsupervised video behavioral recognition scenarios. Without any editing or tagging, the system aims to recognize vehicle types and driving behavior throughout the entire frame

by utilizing group recognition and collaborative behavior of the system. The validation results show that the system could cluster the different behaviors reasonably well in scenarios of the video behavior recognition and that it could dynamically adapt the agent network data to newly emerging situations. The paper includes an introduction to related research work and the inspiration for this study (Section 2); DTS coding, the modeling method, and how the multi-intelligent system operates and the collaboration between agents (Section 3); experimental operation (Section 4); results and analysis (Section 5); discussion (Section 6); and a conclusion (Section 7).

2. Related Work

Mainstream multi-agent systems use reinforcement learning for policy optimization. Moreover, most use centralized training and decentralized execution (CTED) approaches, such as MADDPG and QMIX, where the learning process is centralized and relies on labeled datasets to proceed with offline learning. For an agent evaluation problem such as a credit assignment, Google's Sunehag et al. [4] proposed a novel value-decomposition network architecture to solve the "lazy agent", making each agent a learning goal of its own, and the overall contribution of the team is the sum of individual contributions. Iqbal and Sha [8] introduced the attention mechanism on the basis that agents need to pay more attention to those agents that require attention while learning, allowing the agents' attention to be differentiated. Qian et al. [9] propose an evolutionary population curriculum to solve the problem of excessive learning complexity as the number of agents increases. However, the centralized training approach still limits the learning freedom of individuals, and it also requires the manual setting of global evaluation criteria for all agent behavior, limiting the dynamics of the system.

In video processing, the optical flow method is a commonly used computer vision technique for estimating pixel motion from image sequences. For example, FlowNet [10,11] uses a convolutional neural network to solve the optical flow estimation problem. Moreover, Two-Stream [12–14] uses two classifiers, one trained for RGB graphs and one trained for optical flow graphs, and then the results of both are subjected to a fusion process. However, the optical flow method is computationally demanding and slow owing to its high density computation. In unsupervised video learning, DPC [15] slices video into nonrepeating blocks by intensive computation and thus predicts future frames. MoAp [16] is similar to our design concept, which is based on the idea that the human visual system is based on a series of learned patterns for computing the exact statistics of the data. Additionally, deep neural embeddings [17] improve the video instance embedding framework by capturing frames from videos and then composing video embedding from picture embedding to achieve clustering. However, the unsupervised learning approach requires multiple frames of interest to be found autonomously, thus making the processing task computationally intensive. Inspired by traditional optical flow [18,19] and MP3 [20], we chose to use a background-based modeling approach to convert video frames using filtering to discrete sequence coding in a binary encoding, which leads to a loss of accuracy and detail but enables a focus on the changing patterns of the motion process. In the application of recognizing video-level motion processes, binary encoding methods can satisfy clustering operations by focusing on the change process without any impact from the loss of precise RGB information and object details. Furthermore, in the context of video behavioral clustering recognition scenarios, density-based spatial clustering of applications with noise (DBCSCAN) [21] and hierarchical agglomerative clustering [22] are two equally applicable algorithmic models. DBCSCAN is a well-known density-based clustering algorithm suitable for clustering analysis of lower-dimensional data, while hierarchical performs multilayer clustering in a treelike structure. Both algorithms have high clustering recognition rates, averaging above 90%.

3. Method

3.1. Discrete Temporal Sequence and Empirical Models

Vision behavior occurs with change in RGB information; for example, a hand wave causes a change in the associated screen area within a specified time. If binary data (0 meaning no change, 1 meaning a change) is used to represent the change in a certain area of the screen, as shown in the top half of Figure 1, passing vehicles through the region is represented as "1", but different driving situations cannot be distinguished. By subdividing the screen into four regions as illustrated in the bottom half of Figure 1, data from each of the four regions can be encoded separately with the same screen space, allowing for the differentiation of different vehicle situations in the same scene. The four regions can be coded in chronological order, and the four coding sequences of "0000", "0011", "1100", and "0110" can be obtained to represent changes of the four regions over time. This way, by finely dividing and covering the entire visual image area, visual information can be encoded as a DTS, and feature information extraction can be achieved. For similar intersections or other complex dynamic scenarios, the structure shown in Figure 1 may not be sufficient, so additional refinement is necessary.

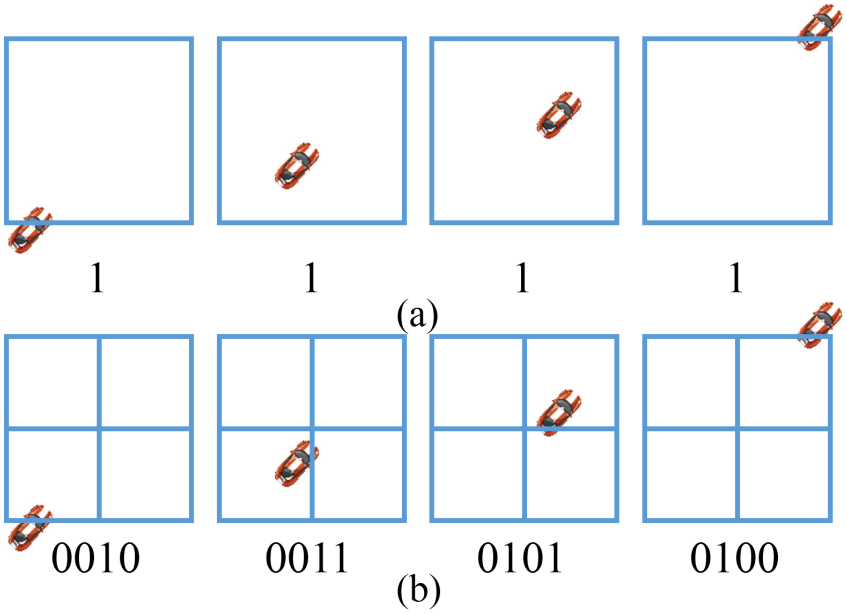


Figure 1. Encoding the DTS.

Following the principle of associative memory, our designed agents serially model the input discrete temporal sequence. In addition, the model is a discrete sequence. The model is used to match the new input DTS using the model to achieve the clustering capability of the discrete temporal sequence. IDs of selected models from multiple agents are merged at a higher-layer agent to form a new DTS. As shown in Figure 2, data such as "a", "b", "c", and "d", which are input from different frames of an agent, are gradually combined into the DTS "abcd…" The agent selects the most suitable model by matching it with the input DTS and outputs the model number as the clustering result using multiple models that are saved locally. When "1", "2", "2", and "1" are output at different frames, they are combined into a new DTS "1221…" at the next agent, which represents the activity of the agent. As for the model built and used by the agents using the memory-like method, we call it the empirical model (EM). EM is the result of clustering the input DTS by an agent. By matching the input DTS with the EM, the merging operation can execute the features of the DTS and EM that matched, which updates the learning model and enables online learning.

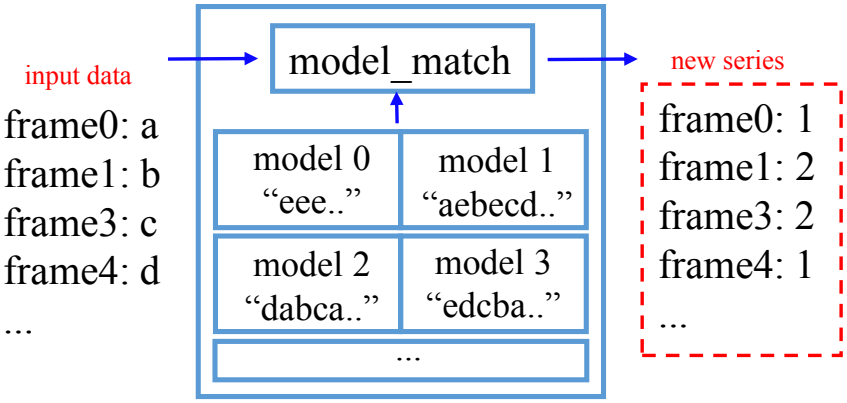


Figure 2. Application of DTS and EMs.

3.2. Multi-Agent Clustering System

The computational process of discrete temporal sequence matching and the fusion update and maintenance of the model is performed in each agent, employing a multilayer hierarchical structure to combine multiple agents. It is worth noting that this system is designed using traditional methods rather than a fully connected neural network. A pyramidal structure is used between layers in the multilayer system, where correlated regions correspond between different layers. Agents only communicate with two types of data transfer: clustering results, passed from bottom-up, and prediction feedback, passed in the opposite direction. The communication between agents is performed with only two kinds of data transfer. The LCS similarity comparison algorithm calculates the matching degree between any-length DTS and EM in terms of sequence similarity, applied in the matching computation process, model feature fusion, and prediction. The model-matching degree is evaluated using the model weights and the LCS which is obtained by dynamic programming. For the model upgrading operation, according to the model matching degree, it was decided to update the selected EM in which the LCS is used to modify the weights of EM units to set an effect of strong and weak memory, or create a new one.

Each agent implements the computational process of matching discrete temporal sequences while also performing model operations such as merging and modeling. Agents employ uniform functional modules within a homogeneous design, and the multi-agent system is combined using a multilayer, hierarchical structure. Figure 3 shows the structure of three agents, wherein agent 1 and agent 3 only represent the interface part. Interfaces S_in and D_out serve for data input and clustering results, respectively. Moreover, interfaces Fb_in and P_out receive feedback information and output prediction results in the opposite direction of data transfer. The History Encoder module encodes input data over time into DTSs, the Model Library module saves local EMs, and the Compare Core module computes matching scores between DTS data and EMs using feedback information and history selections from the History State. The Model_op module performs model feature merging or generates a new model based on the matching performance to achieve self-supervised learning effects.

Assuming the existence of empirical model $M = \{m_1, \ldots, m_n\}$ and the DTS $\mathbf{\Delta} = \{\delta_1, \ldots, \delta_n\}$, the specific algorithm formula of the dynamic planning table using dynamic programming to find the longest subsequence of the EM with DTS is as follows:

$$T_{ij} = \begin{cases} T_{i(j-1)} + 1, & \delta_i = m_j \\ T_{(i-1)j}, & \delta_i \neq m_j \end{cases} \quad (1)$$

where T is the planning table generated by dynamic programming, δ_i is a certain sequence member, m_j: a certain model member, and i and j denote pointers to $\mathbf{\Delta}$ and M, respectively.

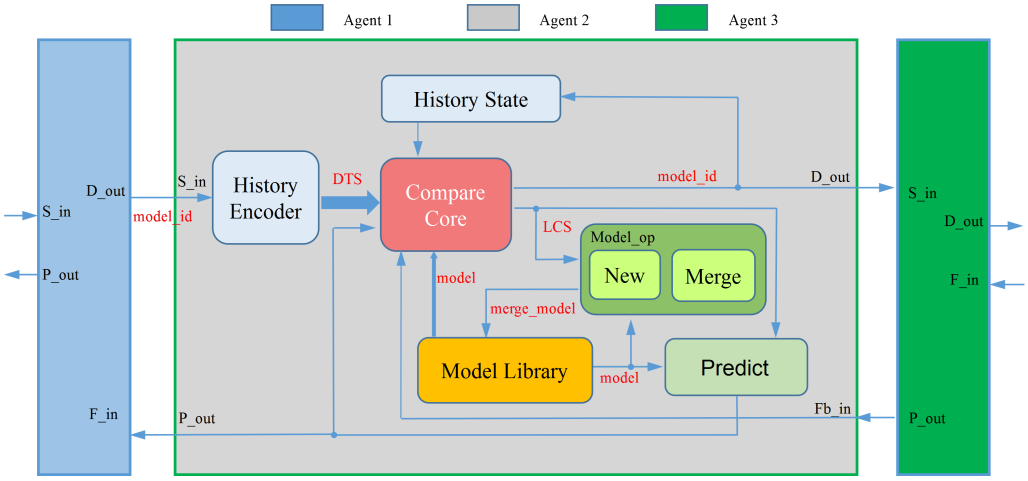


Figure 3. Agent structure diagram.

Moreover, in a multi-agent system, if each agent is designed only to prioritize its own performance, it is similar to implementing an independent learning approach for a single agent, ignoring any underlying systemic relationship among the agents. To address this challenge, the system uses the DTS and bottom-up multilayer approach, in which higher-layer agents can learn the model-matching pattern from the lower-layer agents' behavior. The improved performance is transmitted as feedback to the lower-layer agent, thereby enhancing its capability to provide priority suggestions during subsequent model-matching operations. During the model-matching process, if the entity maintains models that exhibit significant differences, the entity must select the most accurate model at the outset of the DTS step. However, in cases where similar fragments of several models overlap, targeted instructions through DTSs are needed to make informed selections before prediction. Feedback from the higher-layer agent simplifies making faster selections among several similar models and consequently improves data stability, heightening the learning convergence of the whole system. Should the feedback be inaccurate, the agent has the liberty to disregard it depending on the immediate performance. By employing a multilayer structure and a decentralized self-supervised execution approach, the system exhibits strong robustness when dealing with dynamic situations in the fields of computer vision and multi-agent systems.

4. Experiment

In the validation phase, vehicle behavioral recognition in image processing was chosen as the validation scenario, using a dataset from the Rome Street View traffic intersection [23], where different types of vehicle driving styles exist. Because the model learning process can be updated online, there is no dedicated training and testing phase but rather a one-off process. The experiments were conducted using raw video data for random input and statistics on the correct rate of behavioral clustering and the effect of self-supervised online learning.

As shown in Figure 4, the raw frame is divided into 256 sub-regions of size 16×16, forming a total of 256 DTSs. Then, a four-layer proxy structure is used for hierarchical covering, where each grid in the figure represents an agent. The correspondence between grids of the same color indicates that the agent handles data of corresponding regions of a lower layer. Thus, the first layer uses 64 agents for receiving the 256 DTSs that divide the image, and each agent receives the data corresponding to four regions. The second, third, and fourth layers use 16 agents, 4 agents, and 1 agent, respectively, where the fourth layer agent's clustering effect represents the result of the whole system. In the experiment, the test was performed a total of 500 times for three driving behaviors, such as straight ahead and right turn, for the vehicle videos.

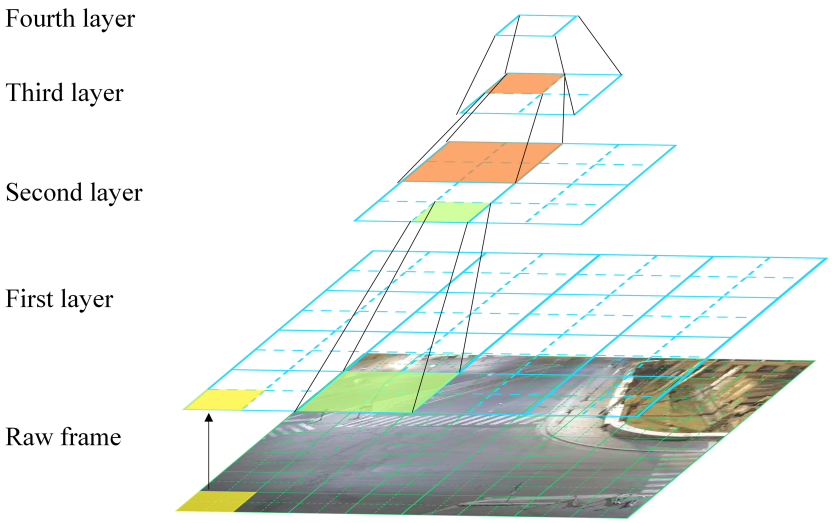


Figure 4. Grid division diagram. One agent at each layer processes the data from the four agents below.

5. Results

5.1. Clustering Performance

Table 1 compares the performance of our system with that of the DBCSCAN and hierarchical algorithms, which were implemented and tested on the same dataset. The evaluation metrics used for performance comparison include the clustering correct rate (CCR), which represents the percentage of clusters identified correctly for all data, the algorithm space complexity (memory), and the processing time (PT), which only counts the processing time for clustering calculations of each algorithm for ease of comparison without considering the encoded DTS of video data and the input time of videos. Our system's computational tasks are decentralized through many agents, translating into collaborative agent behavior with less complex computational tasks per agent. Therefore, the PT per frame for our system was 0.31 s lower than that of other algorithms. Moreover, because the agents do not perform direct floating point computation operations internally but rather query the EM and use the LCS comparison algorithm, the space complexity was O (N^2). Through the multilayer clustering synergy, our system effectively collected common features of different regions well to achieve clustering of motion behavior, exhibiting a CCR of 97.4%.

Table 1. Clustering algorithm comparison.

Algorithm	CCR (%)	PT (S)	Memory
DBCSCAN	81.2	1.54	O (N)
Hierarchical	98.5	0.44	O (N^2)
Ours	97.4	0.31	O (N^2)

5.2. Adaptive Learning Process

Figure 5 depicts the dynamics of the cluster identification experiment performed on our system. Figure 5a shows the correct rate of clustering for all of the agents (85 agents) in the test. Figure 5b,c show the interaction of several agents in two layers during the two inputs. Figure 5d,g show the correct rate of clustering of agents in the first to fourth layers in the test, respectively. The horizontal coordinate indicates the number of tests, while the vertical coordinate indicates the proportion of agents clustered correctly. The first 32 times (the orange) represent the system start-up phase, where two types of behavioral sequential input were performed to help initial model building, followed by (the green) two types of behavioral random tests. Starting from the 200th time (the blue), the third type was gradually added during the random test until the end of the test.

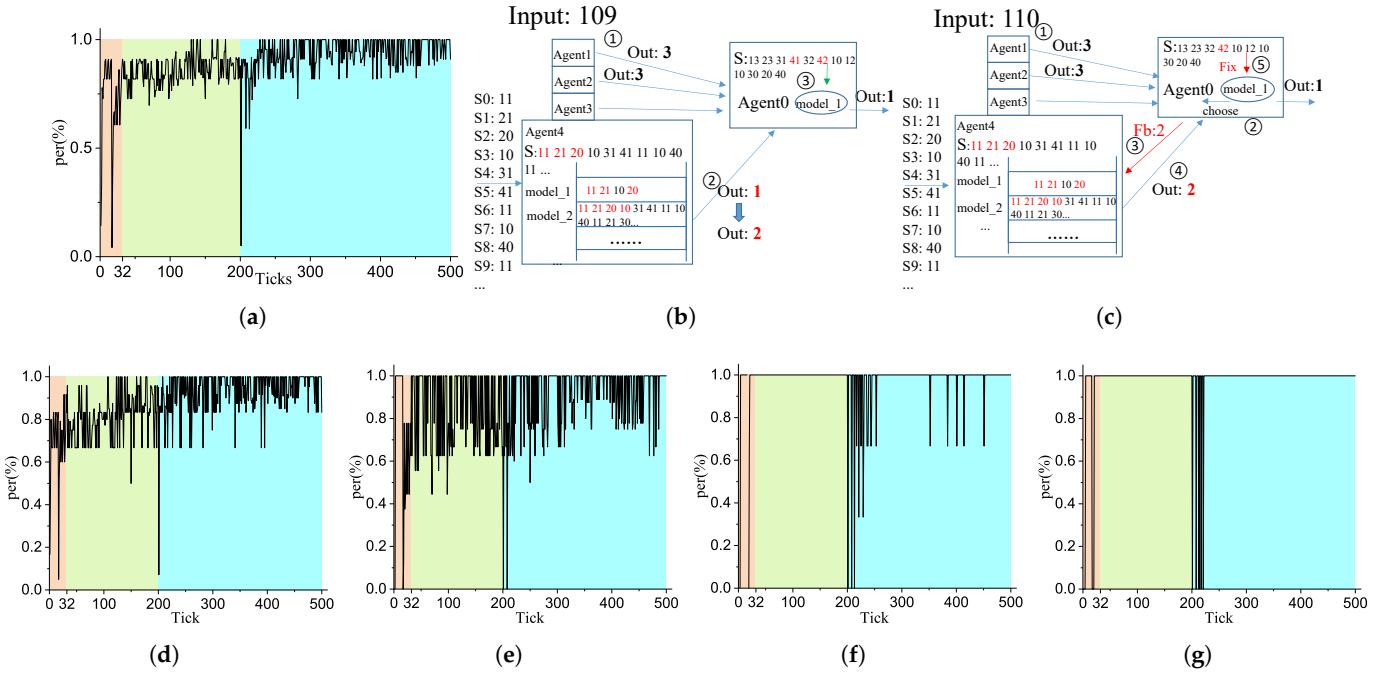


Figure 5. Clustering results. (**a**) The total correct recognition rate for all agents; (**b**,**c**) specific cases of synergy between agents; (**d**–**g**) the correct recognition rate of agents at each layer.

As shown in the figure, in the initial stage and for some time after the 200th test, the absence of the correlation model led to poor clustering results. Subsequently, the clustering effect gradually improved with the tests. This effect is seen in Figure 5a,d–g, corroborating the hypothesis of continuous learning, which is the ability to perform online modeling learning and continuously update it with new changes.

As shown in Figure 5a, the prediction correctness showed a gradual and stable trend with the increasing number of inputs. In contrast, at around the 200th test, there was also a sharp decrease owing to the addition of the new type of video input. The prediction success rate of all of the agents recovered quickly and continued to rise under the conditions of the all-random input method, and the amplitude of the oscillations continued to converge. In summary, in the context of continuous data input and the addition of new types, the system can perform learning and prediction promptly and adapt to new types.

Figure 5b,c illustrate the experiment's ability to produce synergy between agents through predictive behavior. (The figure simplifies the process of receiving feedback from agent 0, as it is in the first layer and did not produce any feedback output.) As shown, agent 4 kept at least two empirical models, and there were partially identical segments or constituent elements in both models, but there were also significant differences. During input 109, agent 4 performed the process of selecting model "1" first and changing to model "2" in the middle, and both models' numbers were output. For agent 0, the data received from agents 1–4 gradually formed a complete discrete sequence, which can be represented as a "memory" of the changing process of agents 1–4. By comparing the difference between the execution of inputs 109 and 110, we found that agent 0 used the new model updated after input 109, as well as feedback to agent 4 with different information, thus helping agent 4 select a more matching model in advance. Agent 0 used the model changes to assist agent 4 in performing predictive actions and to help agent 0 with model updates. In addition, the execution process of agent 0 completed the act of information abstraction between agents to receive multiple agent messages, match, and send the corresponding models.

The figure presented in Figure 5d–g shows the clustering profiles across multiple agent layers, as described below. Agents 1–64 represent the agents in the first layer (Figure 5d); similarly, agents 65–80 represent the second layer (Figure 5e), and agents 81–84 are situ-

ated within the third layer (Figure 5f), whereas there is only one agent (agent 85) in the fourth layer (Figure 5g). In the area corresponding to each layer, the meaning of the vertical axis indicates the number of agents activated, as the quantity of triggered agents varies based on the type of video input. (This is because the number of activated agents is not consistent in cases of different types of videos. For example, in a certain region of the screen, one type of input is always stationary, while another type of input changes, and in both cases, the relevant agents managing this region are in different execution states.) The agents were divided into three different markers: inactive, correctly predicting, or incorrectly predicting, according to different status and clustering results. From Figure 5d–g, it can be observed that agent predictions remain relatively stable at higher levels of the system, whilst greater fluctuations occur at lower levels. The stratification of agents serves to demonstrate that even in the presence of erroneous predictions from lower-tier agents, the majority consensus among higher-level agents ensures accurate predictions.

5.3. The Robustness of the Adaptive System

In experiments with randomly generated video inputs and newly introduced patterns, the system showed a clear trend of improved clustering performance and strong robustness. This is because not all agents in the system directly receive and process raw video data. In the four-layer structure used in the experiment, only the 64 agents in the first layer received and encoded the raw video information. The 16 agents in the second layer received the output data from the first layer agents as input. Specifically, the first layer of agents encoded the raw video information and used EMs to identify clusters based on the encoded DTSs. After identifying the clusters, the correct model number was selected as output and transmitted across different frames. The agents in the second layer then received the outputs of the first layer to create new DTSs. This process continued with agents in the third layer receiving the outputs of the second layer and agents in the fourth layer receiving those of the third. Although the number of agents decreased at each successive layer, the DTSs received by the agents covered progressively larger areas, with the fourth-layer agents acting on the entire image region. The agents in different layers interact in situations such as those shown in Figure 5b,c and, through feedback and self-supervised learning, improve the model's learning effectiveness, and promote improved clustering. For instance, fully random unsupervised input testing was performed, and a new motion law was added in the 200th experiment designed in the study. To address these dynamic issues, the system can rapidly learn emerging changes in rules and effectively incorporate them.

6. Discussion

6.1. Decentralized and Multilayer Structure

The reason for the sustained improvement in recognition performance of unsupervised video and adaptive learning of new patterns in the experiment is due to the use of decentralized execution. Although using completely decentralized execution cannot directly optimize the learning effects of top-level tasks like centralized evaluation, it can result in individual agents focusing only on their own learning effects. However, the design of a multilayer structure combined with decentralized execution allows some agents to perform poorly on tasks and even large numbers of agents may alternately have poor effects, as long as most agents perform well in the overall execution of the multi-agent system, good clustering results can be achieved. As shown in Figure 5d–g, the clustering accuracy of the layered agents is difficult to maintain for all agents in the first three layers (especially the first and two layers), but the fourth layer (system task hierarchy) can achieve good clustering results. This feature with layer-by-layer feature retention and commonality transfer enables the global goal obtained from the individual goal to gradually reach convergence, thus achieving the unification of the individual and global prediction goals. Even if the environment changes or new data arrive, the system also converges quickly based on such a prediction feedback synergy and error processing mechanism.

6.2. Collaboration Triggered by Feedback Behaviors

Controlled experiments were conducted on cooperative effects under the same experimental conditions and datasets by removing inter-agent feedback behavior. The overall clustering identification results without inter-agent feedback under these circumstances are shown in Figure 6. Compared with Figure 5a, the system's clustering effectiveness did not see a noticeable improvement as the testing experiments progressed, losing its ability to improve the model. Especially after adding new rules for the 200th experiment, which resulted in a decrease in identification performance, the system could no longer enhance recognition performance, leading to poor identification efficiency. This is because the clustering identification of the system is achieved through the joint clustering of behaviors among numerous layered agents, where the clustering identification performance of agents directly influences the identification efficiency of the system. Inter-agent feedback behavior refers to the suggestion from a higher-layer agent to a lower-layer agent, based on the higher-layer agent's summary of the model matching process of the lower-layer agent. Once the feedback is removed, agents can only rely on self-supervision to judge whether the previously selected model was correct based on the existing input data, resulting in repeated switching of the model according to the current data. For the DTSs that are aggregated based on such model matching processes, the higher-layer agent finds it more challenging to cluster and recognize them. The addition of feedback can help lower-level agents choose suitable models in advance. The agent that selects the correct model in advance not only provides similar feedback behaviors to corresponding lower-level agents but also produces more stable model number outputs in cases of video frames, which are more conducive to behavioral clustering at higher levels and improve the system's behavioral recognition efficiency. Comparative experiments in Figures 5a and 6 demonstrate that inter-agent feedback behavior can effectively enhance the continuous learning capability of the system.

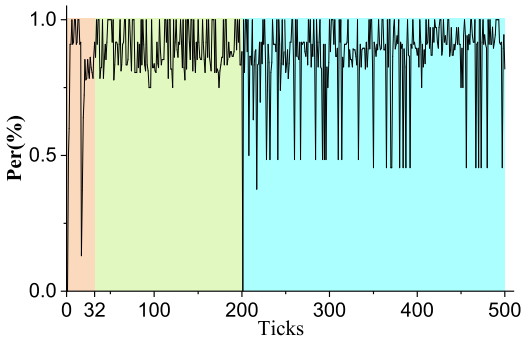


Figure 6. Clustering without synergy.

6.3. Limitations and Future Directions

However, regarding the proposed cluster recognition system, there are still many factors that can be explored. For example, the choice of the number of layers in the system is not limited to a four-layer structure, nor is it limited to an exponentially increasing number of agents, provided that the number of layers is enhanced. In the experimental validation session, a five- or even six-layered structure was also tested; after all, the more layers, the finer the original information that is segmented, theoretically allowing for more refined clustering results. However, during the validation process, we found that the increase in the number of layers did not lead to much improvement in clustering performance compared to the four-layer structure with the same dataset. Of course, this may be because the dataset is not yet rich enough. In addition, the increase in the number of layers undoubtedly reduced the processing rate of the system at the software level because of the way the processors execute serially. If, however, the system is validated on a parallel hardware platform, the processing speed will not be limited by the size of the agents.

7. Conclusions

This paper proposes an adaptive clustering multi-agent learning system for intelligent applications with continuously changing requirements. At the single agent level, the system uses the memory-like method to construct models and primarily executes memory access and comparison operations to achieve clustering recognition in a low computational environment. Among multiple agents, a multilayer decentralized cooperation approach enhances adaptability in learning, enabling self-supervised clustering. The proposed system's feasibility is demonstrated through video behavioral clustering identification, indicating that it can achieve clustering functionality and effectively learn and adapt to new changes in low-computational power environments. These results suggest that the proposed system holds promise for intelligent sensing front-end applications.

Author Contributions: Conceptualization, X.C., X.Q. and S.L.; methodology, X.C., X.Q. and A.Y.; software, X.Q. and A.Y.; validation, X.Q.; formal analysis, X.C.; investigation, W.D. and W.Y.; resources, A.Y. and X.Q.; data curation, A.Y. and X.Q.; writing—original draft preparation, X.Q.; writing—review and editing, X.C. and W.Y.; visualization, X.C. and X.Q.; supervision, W.Y.; project administration, Z.S.; funding acquisition, Z.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Strategic Priority Research Program of the Chinese Academy of Sciences (grant number XDB44010200).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: I want to thank Longfei Liang for guidance in my research and the NeuHelium Co., Ltd. for their great support.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DTS	discrete temporal sequence
IQL	independent Q-learning
VDN	value-decomposition networks
CTED	centralized training and decentralized execution
EM	empirical model
LCS	longest common sequence

References

1. Hernandez-Leal, P.; Kartal, B.; Taylor, M.E. A Survey and Critique of Multiagent Deep Reinforcement Learning. *Auton. Agents Multi-Agent Syst.* **2019**, *33*, 750–797. [CrossRef]
2. Gronauer, S.; Diepold, K. Multi-Agent Deep Reinforcement Learning: A Survey. *Artif. Intell. Rev.* **2022**, *55*, 895–943. [CrossRef]
3. Tampuu, A.; Matiisen, T.; Kodelja, D.; Kuzovkin, I.; Korjus, K.; Aru, J.; Aru, J.; Vicente, R. Multiagent cooperation and competition with deep reinforcement learning. *PLoS ONE* **2017**, *12*, e0172395. [CrossRef] [PubMed]
4. Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W.M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J.Z.; Tuyls, K.; et al. Value-Decomposition Networks for Cooperative Multi-Agent Learning Based on Team Reward. In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '18, Stockholm, Sweden, 10–15 July 2018; pp. 2085–2087.
5. Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; Mordatch, I. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In Proceedings of the Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; pp. 6382–6393.
6. Rashid, T.; Samvelyan, M.; Schroeder, C.; Farquhar, G.; Foerster, J.; Whiteson, S. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 4295–4304.
7. Foerster, J.N.; Farquhar, G.; Afouras, T.; Nardelli, N.; Whiteson, S. Counterfactual Multi-Agent Policy Gradients. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence

Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI'18/IAAI'18/EAAI'18, New Orleans, LA, USA, 2–7 February 2018; AAAI Press: Palo Alto, CA, USA, 2018; pp. 2974–2982.
8. Iqbal, S.; Sha, F. Actor-Attention-Critic for Multi-Agent Reinforcement Learning. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Volume 97, pp. 2961–2970.
9. Long, Q.; Zhou, Z.; Gupta, A.; Fang, F.; Wu, Y.; Wang, X. Evolutionary population curriculum for scaling multi-agent reinforcement learning. *arXiv* **2020**, arXiv:2003.10423.
10. Dosovitskiy, A.; Fischer, P.; Ilg, E.; Häusser, P.; Hazirbas, C.; Golkov, V.; Smagt, P.v.d.; Cremers, D.; Brox, T. FlowNet: Learning Optical Flow with Convolutional Networks. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2758–2766.
11. Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; Brox, T. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1647–1655. [CrossRef]
12. Feichtenhofer, C.; Pinz, A.; Zisserman, A. Convolutional Two-Stream Network Fusion for Video Action Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1933–1941.
13. Tran, A.; Cheong, L.F. Two-Stream Flow-Guided Convolutional Attention Networks for Action Recognition. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017; pp. 3110–3119. [CrossRef]
14. Li, M.; Qi, Y.; Yang, J.; Zhang, Y.; Ren, J.; Du, H. 3D Convolutional Two-Stream Network for Action Recognition in Videos. In Proceedings of the 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 4–6 November 2019; pp. 1697–1701. [CrossRef]
15. Han, T.; Xie, W.; Zisserman, A. Video Representation Learning by Dense Predictive Coding. In Proceedings of the 2019 IEEE/CVF ICCVW, Seoul, Republic of Korea, 27–28 October 2019; pp. 1483–1492. [CrossRef]
16. Wang, J.; Jiao, J.; Bao, L.; He, S.; Liu, Y.; Liu, W. Self-Supervised Spatio-Temporal Representation Learning for Videos by Predicting Motion and Appearance Statistics. In Proceedings of the 2019 IEEE/CVF CVPR, Long Beach, CA, USA, 15–20 June 2019; pp. 4001–4010. [CrossRef]
17. Zhuang, C.; She, T.; Andonian, A.; Mark, M.S.; Yamins, D. Unsupervised Learning from Video with Deep Neural Embeddings. In Proceedings of the 2020 IEEE/CVF CVPR, Seattle, WA, USA, 13–19 June 2020; pp. 9560–9569. [CrossRef]
18. Wang, H.; Klser, A.; Schmid, C.; Liu, C.L. Dense Trajectories and Motion Boundary Descriptors for Action Recognition. *Int. J. Comput. Vis.* **2013**, *103*, 60–79. [CrossRef]
19. Wang, H.; Schmid, C. Action Recognition with Improved Trajectories. In Proceedings of the 2013 IEEE ICCV, Sydney, NSW, Australia, 1–8 December 2013; pp. 3551–3558. [CrossRef]
20. Casas, S.; Sadat, A.; Urtasun, R. MP3: A Unified Model to Map, Perceive, Predict and Plan. In Proceedings of the 2021 IEEE/CVF CVPR, Nashville, TN, USA, 20–25 June 2021; pp. 14398–14407. [CrossRef]
21. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, Portland, OR, USA, 2–4 August 1996; AAAI Press: Portland, OR, USA, 1996; pp. 226–231.
22. Patel, S.; Sihmar, S.; Jatain, A. A study of hierarchical clustering algorithms. In Proceedings of the 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 11–13 March 2015; pp. 537–541.
23. MAMMALS. 2022. Available online: https://github.com/qian-git/MAMMALS (accessed on 28 October 2022).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.