Unsupervised Continual Learning From Synthetic Data Generated with Agent-Based Modeling and Simulation: A preliminary experimentation

Gianfranco Lombardo[1,*,†], Mattia Pellegrino[1,†] and Agostino Poggi[1,†]

[1]*University of Parma, Parma, 43125, Italy*

Abstract
Continual Learning enables to learn a variable number of tasks sequentially without forgetting knowledge obtained from the past. Catastrophic forgetting usually occurs in neural networks for their inability to learn different tasks in sequence since the performance on the previous tasks drops down in a significant way. One way to solve this problem is providing a subset of the previous examples to the model while learning a new task. In this paper we evaluate the continual learning performance of an unsupervised model for anomaly detection by generating synthetic data using an Agent-based modeling and simulation technique. We simulated the movement of different types of individuals in a building and evaluate their trajectories depending on their role. We collected training and test sets based on their trajectories. We included, in the test set, negative examples that contain wrong trajectories. We applied a replay-based continual learning to teach the model how to distinguish anomaly trajectories depending on the users' roles. The results show that using ABMS synthetic data it is enough a small percentage of synthetic data replay to mitigate the Catastrophic Forgetting and to achieve a satisfactory accuracy on the final binary classification (anomalous / non-anomalous).

Keywords
Agent-based modeling, Synthetic data, Data generation, Continual Learning, Replay methods,

1. Introduction

Deep Learning has been growing in scale, breadth of applications, and the amount of required data, the so-called “data-hungry" deep neural networks. This is an important problem to deal with, especially when the task to be learned presents dynamic aspects in terms of environment and classes of tasks where few data are available during the training phase. Moreover, for some context, new tasks become available only during the time and for this reason, the models should be smoothly updated. When learning new tasks from data in sequence, Neural Networks usually face the so-called Catastrophic Forgetting: performance on the previous tasks drops significantly. This is a serious disadvantage that prevents many intelligent applications to real-life problems when not all the context is known beforehand. Continual Learning (CL) aims

WOA 2022: 23rd Workshop From Objects to Agents, September 1–2, Genova, Italy
*Corresponding author.
†These authors contributed equally.
gianfranco.lombardo@unipr.it (G. Lombardo); mattia.pellegrino@unipr.it (M. Pellegrino); agostino.poggi@unipr.it (A. Poggi)
0000-0002-0877-7063 (G. Lombardo); 0000-0001-7116-9338 (M. Pellegrino); 0000-0002-9421-8566 (A. Poggi)

to introduce novel methodologies to enable the update of a model in terms of new tasks and data distributions, unseen in the past, without losing the already acquired knowledge to solve the previous tasks accurately [1].

On the other hand, another possibility to have a higher quantity of data or enough data for new tasks is exploiting synthetic data generation techniques [2]. More recently, it has also been proven that using synthetic data during training can reduce Catastrophic Forgetting when used for a Continual Learning framework [3].

In several contexts, Agent-based Modeling and Simulation (ABMS) techniques can represent an effective way to generate synthetic data. In this paper, we propose the preliminary experimentation performed using Continual Learning with synthetic data generated using ABMS. In particular, we investigated an unsupervised reconstruction task of high-resolution trajectories that aims to recognize the anomaly trajectories followed by the users inside a building depending on their job role. Users are divided in three classes: customers, workers and maintenance operators. Trajectories of each users have been generated exploiting the Boid model [4] and Agent-based Modeling and simulation techniques. An unsupervised Deep Recurrent Neural network is trained to recognize the anomalies in the trajectories of each class of users by exploiting a Replay-based Continual Learning technique.

2. Literature review

To correctly generate synthetic data to exploit the characteristics of continuous (lifelong) learning, we first had to choose a model suitable for our use case. We considered the main modeling and simulation techniques used for crowd and pedestrian modeling within the current literature, at different scales of complexity. A large number of models are available to implement simulations regarding pedestrians inside or outside buildings and their relative behavior. These models can be classified into two main classes:

- macroscopic models: focus on the whole system as a single entity (regression models, route choice models)
- microscopic models: study the behavior, decision, and attributes of the individuals and their interaction among themselves (rule models, cellular automata).

Regression models analyzed crowd relations in order to predict pedestrian flow behavior under specific circumstances. This method is strong infrastructure bound [5]. In [6] *Nassereddine et al.* used a non-probabilistic regression approach to model the behavior of right-turning drivers: when drivers perceive the possibility of a pedestrian reaching a critical conflict point at the same time as them, they will modify their behavior, even if not coming to a complete stop.

In route choice models pedestrians choose their destinations in order to maximize the utility of their path. The utility defines a set of properties: travel time, points visited, etc. In [7] *Alivand et al.* model a scenic route between two locations identifying several variables of the surrounding environment as significant contributors to route scenery, they considered a scenario regarding tourist and recreation travelers. They used, more specifically, a Path Size Logit (PSL) model, to identify the relevant attributes and their relative importance.

Cellular Automata divide the space into a uniform grid. Each agent owns a particular grid position, a single grid position is called "cell". Each agent can move between cells depending on the modeling system. Cellular Automata use discrete time steps to measure the time. The variables at each cell are updated simultaneously based on the values of the variables in their neighborhood at the previous time step [8]. In [9] *Zhao et al.* used a 2D random cellular automata to model the occupant evacuation considering the influence of human psychology and behavior.

Rule-based models have been used to simulate flocks of animals or crowds of people. A well-known model that is able to simulate life-like complex behavior is the boid model [10]. Each agent is an independent entity that navigates according to its perception and the environment where is placed, the physics, and its behavior. The aggregate motion of flocks is created by such distributed model. To simulate a generic flocking behavior, the basic rules are just three: they describe how an entity has to move, taking into account its position, its velocity, and how near to another entity is:

- separation: steer to avoid crowding local flock mates
- alignment: steer towards the average heading of the local flock mates
- cohesion: steer towards the average position of the local flock mates

In order to use these three rules, it's important to take into consideration not only the position of an entity but even its neighborhood. So it must be considered that managing the neighborhood can also be very complex. Each entity moves and therefore must be considered along with all the other entities within a certain radius. The computation complexity is $O(n^2)$, where n is the number of entities in the neighborhood. To simplify and speed up this operation, a neighborhood can be modeled in a simpler way. Each agent can move in a limited range, so they can move a lot over time. The basic idea is to divide the space into a set of bins. The steps are:

- Create a circle or sphere (2D or 3D)
- See if the bin falls into the sphere
- Follow the rules for cohesion, separation, and alignment

In [11] *Shinoda et al.* designed and developed a pedestrian dynamics simulator to perform navigation system studies. Moreover, they performed simulation experiments with a novel wearable navigation device that can guide a pedestrian through an evacuation process.

In the light of these analyses, we decided to use the boid approach to model our use case. Moreover, we are interested to simulate a complex scenario at a microscopic scale. The agents, involved in the simulation process, have to reach predetermined points and reach gradually the end of their path. For these reasons, we decided to adopt a rule-based model (boid) because it is well-suit for our needs.

After choosing the most suitable simulation method, our attention shifted to the trajectory data generation, produced by the system, and the exploitation of these data for anomaly detection tasks and continual (lifelong) learning.

Detecting an anomaly behavior is an important problem in various situations. Finding an anomalous behavior may indicate important objects and events in a wide variety of domains. In literature, there is a huge variety of algorithms that can detect an anomalous behavior during a predetermined trajectory [12]. In [13] *Laxhammar et al.* proposed a Sequential Hausdorff Nearest-Neighbor Conformal Anomaly Detector (SHNN-CAD) for online learning and sequential anomaly detection in trajectories. However, in this work, we adopted an encoder-decoder architecture based on Long-Short Term Memory (LSTM) to perform the anomaly detection task. We chose LSTM among other models for its ability to deal with temporal sequence and maintain the temporal dependencies [14].

Continual learning provides the ability to learn continuously and enrich its ability when data and tasks become available over time [1]. Continual Learning updates its intelligent model taking into account data and attributes that it didn't see in the past training steps, without losing any information about the past (Catastrophic Forgetting) [15].

To minimize the forgetting phenomenon, in a real use case, it might be useful to use synthetic data along with the real ones. In [16] *Ma et al.* trained a multi-agent future trajectories predictor given a stream of datasets collected at different scenarios. In particular, they use a graph-neural-network-based conditional generative memory system to mitigate catastrophic forgetting. However, in scientific literature there is currently a lack of unsupervised models trained with a CL paradigm. Indeed, most of the state of the art techniques are related to supervised tasks.It is possible to distinguish three categories of CL techniques:

- Replay-based methods: When learning a new task, the model is exposed with a variable percentage to examples from the previous learned tasks [17, 18]
- Regularization-based methods: Continual learning is performed by acting on the parameters of the model by discouraging the updating of neural network's layers that are deemed relevant for the single tasks. Unfortunately, these methods does not scale well when increasing the number of tasks [19].
- Architectural methods: Developing ad-hoc solutions where different parts of the model take care of each task exclusively [20, 21]

3. Use case

As use-case to investigate continual learning from ABMS we choose an anomaly detection task related to people's trajectories inside a building as a service offered by an intelligent Location-based service. We hypothesized an indoor localization infrastructure that locates individuals using different tracking technologies with different sample and errors rate.

We focused on the development of an intelligent service which exploits Location-based service data. Indeed, Locate and analyze indoor users' paths may introduce novel intelligent services that can help management to make predictions, measure performance indicators and ensure safety conditions. Examples of services could be: Measuring the inefficiency of spatial organization, occupation of spaces prediction, users waiting times estimation and finally anomaly detection.

We focus on indoor users' paths to classify if they are allowed to visit specific environments depending on their role or to understand if any change in the building organization induces anomaly trajectories for a set of users that have the same role in the organization. Different classes of users can emerge over time and are required to be learned within their normal behaviors. Depending on the target organization the roles can be different: for example, in a hospital scenario could be medical staff, patients, and maintenance staff; In a retail environment could be customers, salesmen, and warehouse workers. Modeling and simulating the users' behaviors offer mainly two advantages:

- It is possible to generate synthetic trajectories when a new category of users should be tracked and recognized and there is not real-data available to train the anomaly detection model on this new one
- It is possible to evaluate how the performance of the intelligent service changes depending on the accuracy and sample rate that is used to locate and track the class of users.

This anomaly detection task can be performed using both supervised or unsupervised ML techniques. However, when users' tracking is performed with a high-resolution as a sequence of visited locations in the environment $[(x_0, y_0)...(x_T, y_T)]$ several benefits come up with the adoption of an unsupervised paradigm like auto encoder-based trajectory reconstruction:

- No need of a priori knowledge of the environment topology nor supervision, since allowed trajectories can be learnt by observing a set of trusted real users or with agents that perform trustable trajectories;
- Easy adoption in a new organization;
- Low cost to update the model if something changes in the environment topology

Figure 1 shows a map segmentation of a real building we used for the experiments. The environment is divided according to the main operations that are daily performed. In our use-case the building is a real-existing hospital ward where can interact three different categories of users: patients that need for treatments; medical staff and finally the maintenance operators. Patients enter the ward (yellow area) by the waiting room and proceed through the back-office and reception area (green area). Patients can also visit the clinics areas (orange) and the areas where treatments are provided (pink areas). They cannot access the maintenance area (violet zone) and the private spaces of clinicians and the archive (Red and purple areas).

On the other hand, the medical staff can access all the environments except the maintenance area and can enter the ward from multiple gates. Finally, the maintenance staff can enter from multiple gates, they can access the private area for maintenance but we supposed they cannot access the archive and the private medical offices.

4. ABMS

To realize our ABMS model we exploited the properties of ActoDeS. ActoDeS is a software framework that allows the development of distributed and concurrent systems [22]. ActoDeS is Java-based and inherits some functionalities and some implementation solutions used in JADE [23, 24, 25]. Multiple and various application has been developed with the aid of ActoDeS functionalities, especially for agent-based modeling and simulation [26, 27, 28], data analysis [29, 30, 31, 32, 33], and more recently for epidemiological simulation, [34], [35]. This framework is based on the use of concurrent objects (from here named actors) whose main characteristics derive from the actor model [36]. In ActoDeS, an actor is created by another actor; after its creation, the actor can interact with other actors through the exchange of asynchronous messages and, as a consequence, can change its behavior several times; once its work is finished, the actor kills itself. ActoDeS can also support millions of actors and simulate their behavior. Indeed, it has a special structure in which agents can be divided into different pools and managed using multiple threads and resources.

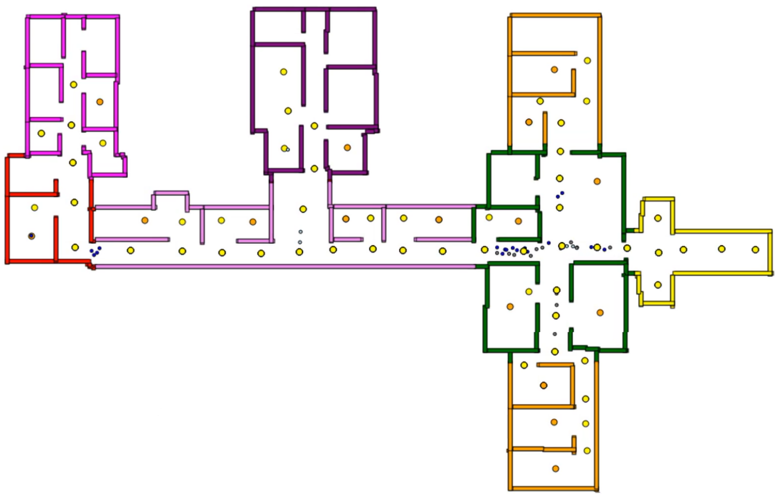

Figure 1: The real-hospital ward map used for the experimental part. The map is color-based segmented according to the daily activities. The colored dots are the gravitational attraction point we used for the Boid simulation using ActoDeS.

In order to develop our use case, we create a virtual environment, where the agents can move and interact among themselves. Each agent represents an individual (actor), that can have a specific role and a specific path. Moreover, it could be subject to limitations and can't access in some specific areas of the virtual environment. We simulate the whole trajectory set by exploiting the Boid model. We can have different types of boid depending on how many agents' categories we want to implement. To create the virtual environment and limit the 2D space in which placing boids we defined as "wall" type actors. These particular types of actors have their own dimension in space and have the task of limiting the movements of the other "pedestrian" actors in a well-defined space.

The agents can reach their location randomly follow other agents or choose between some alternative paths. Each agent is assigned a starting point, a goal point, and a list of waypoints to cross, always according to their category. To generate the most random simulations possible, we planned to decentralize the starting point of every single agent by a certain offset randomly generated. Each trajectory has different lengths.

With this particular type of approach, we can generate different types of boid and different types of professional figures who, for example, move independently in the workplace. Thanks to the ABMS architecture, is possible to generate synthetic data for the training and test set of the application we want to create. Moreover, when we create the test set, we also include negative examples which can contain trajectories for paths not allowed for certain types of boid. Finally, by setting the sampling rate it is possible to simulate a data gathering using different indoor localization techniques.

5. Continual Learning framework

The machine learning model has no prior knowledge about the environment and the allowed trajectories for each category but it learns the allowed ones in the environment for each category as an anomaly detection task over spatial location sequences. The only prior knowledge exploited by the model is the user's role that in a real context would be retrieved according to the technology used to track the user.

The machine learning model learns to reconstruct the high-resolution trajectory of each user using and encoder-decoder architecture aiming to reduce the Root Mean Squared Error (RMSE) among the input sequence and the reconstructed one. This is a common methodology used to detect anomalies in sequences and time-series because the model fails to reconstruct sequence that rely on distributions that have not be seen during the training phase with a higher RMSE. We also provided the user role as input of the model together with its trajectory. We exploited an encoder-decoder architecture based on Long-Short Term Memory (LSTM) deep neural networks. LSTM network have been selected because of their ability of learning patterns on sequence data by considering and preserving temporal dependencies [37]. The encoder network process the trajectory as a sequence of 150 visited locations (x,y) along with a categorical variable that identifies the user category and learns an internal compressed representation that is then used by the decoder network to reconstruct the trajectory. Figure 2 shows the main points of the architecture.

A trajectory is considered anomaly when the Root Mean Squared Error (RMSE) of the two sequences is above a certain threshold that is estimated using a validation set. According to the RMSE, the model classifies a trajectory for a user as not allowed when it results to be anomaly and allowed otherwise. One challenge when dealing with such task with unsupervised techniques is teaching the model that the same trajectory can be for example normal and allowed if the user belongs to the medical staff but not if he belongs to another category without supervising the training. That is the reason why we provide the categorical variable that indicates the user's category in input to the encoder along with its trajectory.

With this hypothesis we experimented this use-case adopting a Continual Lifelong learning paradigm. The model is firstly trained to only reconstruct normal trajectories of doctors which represents the first task. After that the neural network is trained to deal with a second task: reconstruct normal trajectories of the maintenance staff and finally the patients' trajectories. The model is fine-tuned using the validation-set but results have been computed using a balanced test set with the same numbers of normal and anomaly trajectories.

We analyzed how the performances of classifying anomalies change by incrementally learning all the tasks separately. The main goal of this experiment is avoid and reduce the so-called Catastrophic Forgetting of the model when learning new tasks. In our case, this goal becomes more challenging since all tasks are learnt in an unsupervised way. We exploited a Replay-based method by analyzing the performances when changing the percentage of replay samples ($R_{\%}$). The method is composed by the following steps:

1. We first trained the model using only the 100% of doctors' trajectories available (Traditional training) and performing the fine-tune to select the reconstruction error threshold using the validation set.

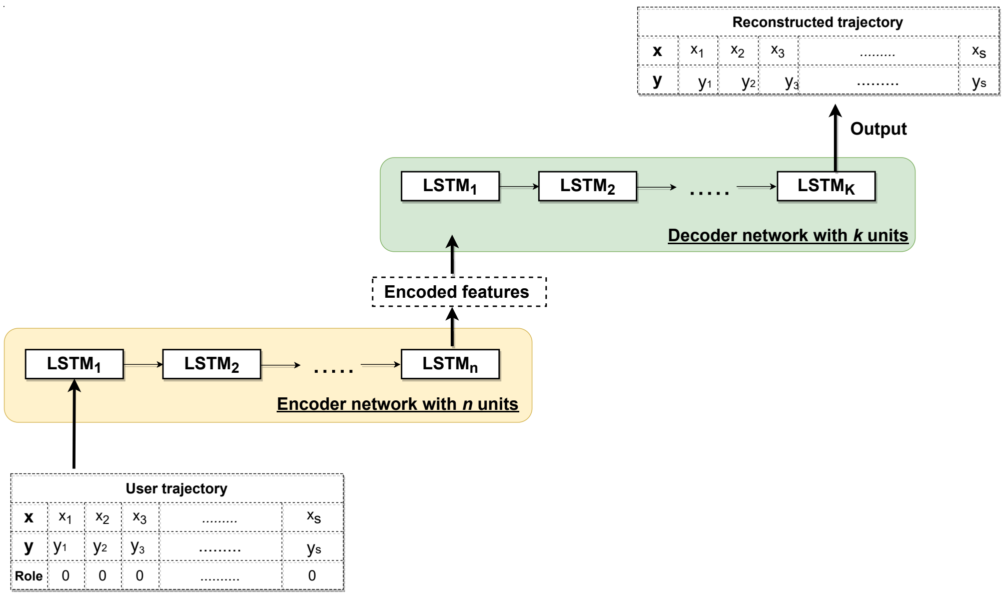

Figure 2: LSTM Encoder-Decoder architecture used to reconstruct each trajectory. The image also shows the structure we adopted for the input and the desired output.

2. Starting from the previous model, the neural network is trained to learn the second task (reconstruct normal trajectories of maintenance staff), by using the 100 % available samples for this new task and an amount of examples of the previous task equal to $R_\%$
3. Finally, the third task (reconstruct normal trajectories of patients) is learnt in the same way as the second one: 100 % available samples for the new task and some additional examples for the previous tasks to avoid the Catastrophic Forgetting of the network. Respectively, the $R_\%$ of the available doctors' normal trajectories and of the available maintenance normal trajectories.

For each configuration we performed 50 runs and report the average performances. Since the model incrementally learns to reconstruct the trajectories of a class of users we analyzed the Precision, Recall and F-1 score achieved by the model when classifying anomalies for each category. Figure 3 shows the result.

6. Results

If trained with a replay-percentage $R_\%$ equal to zero, Catastrophic Forgetting occurs and only the third task is learnt. Introducing a $R_\% = 1\%$, performances improve with a final accuracy in the binary task (normal/anomaly) equal to 62.7%. At the level of the single user category the one that suffers mostly a bad performance is the one related to the maintenance staff.

We then tested with $R_\% = 10, 30, 50\%$. It is interesting that by only replaying the 10% of examples of the first two tasks the final binary accuracy reaches on average the 84.5% and the slope of improvements at the single categories level becomes less pronounced. Finally, we tested using $R_\% = 100$ to compare with a traditional training that achieves a binary accuracy of 91.08 when learning with the a priori knowledge of all the classes of users. The incremental one reaches on average an accuracy equal to 90.8%. It is clear that there is not any important difference by training incrementally over the single categories if the replay involves all the available examples. However, in terms of F-1 score the patient category is slightly learnt better with a traditional learning paradigm.

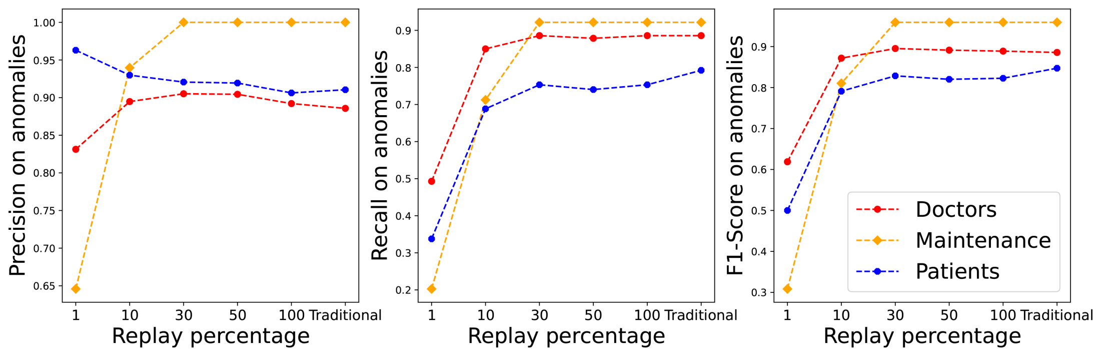

Figure 3: F-measure, Precision and Recall for each task when using a Replay methods for the Continual Learning scenario. Tasks are learnt in the order: Doctors,maintenance and finally patients. Results are reported as an average of different runs with respect of the replay percentage used.

7. Conclusions

In this paper we presented the first experiments we performed with replay-based continual learning using ABMS to generate synthetic data. Moreover, we propose a unsupervised CL approach for anomaly detection of trajectories depending on the users' role. The results are promising and by only using a small percentage of replay it is possible to achieve an overall good accuracy in the final classification task. In our future works we will investigate other CL approaches to understand if they also benefit from ABMS synthetic data. Moreover, we will further analyze a more complex scenario that can involve more than three tasks (users' categories to be recognized as anomaly or not).

References

[1] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: A review, Neural Networks 113 (2019) 54–71.

[2] S. I. Nikolenko, Synthetic data for deep learning, volume 174, Springer, 2021.

[3] W. Masarczyk, I. Tautkute, Reducing catastrophic forgetting with learning on synthetic data, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 252–253.

[4] G. Angiani, P. Fornacciari, G. Lombardo, A. Poggi, M. Tomaiuolo, Actors based agent modelling and simulation, in: International Conference on Practical Applications of Agents and Multi-Agent Systems, Springer, 2018, pp. 443–455.

[5] J. S. Milazzo, II, N. M. Rouphail, J. E. Hummer, D. P. Allen, Effect of pedestrians on capacity of signalized intersections, Transp. Res. Rec. 1646 (1998) 37–46.

[6] H. Nassereddine, K. R. Santiago-Chaparro, D. A. Noyce, Modeling vehicle–pedestrian interactions using a nonprobabilistic regression approach, Transp. Res. Rec. 2675 (2021) 356–364.

[7] M. Alivand, H. Hochmair, S. Srinivasan, Analyzing how travelers choose scenic routes using route choice models, Computers, Environment and Urban Systems 50 (2015) 41–52. URL: https://doi.org/10.1016%2Fj.compenvurbsys.2014.10.004. doi:10.1016/j.compenvurbsys.2014.10.004.

[8] N. Pelechano, A. Malkawi, Evacuation simulation models: Challenges in modeling high rise building evacuation with cellular automata approaches, Automation in Construction 17 (2008) 377–385. URL: https://doi.org/10.1016%2Fj.autcon.2007.06.005. doi:10.1016/j.autcon.2007.06.005.

[9] D. L. Zhao, J. Li, Y. Zhu, L. Zou, The application of a two-dimensional cellular automata random model to the performance-based design of building exit, Build. Environ. 43 (2008) 518–522.

[10] C. W. Reynolds, Flocks, herds and schools: A distributed behavioral model, in: Proceedings of the 14th annual conference on Computer graphics and interactive techniques - SIGGRAPH '87, ACM Press, 1987. URL: https://doi.org/10.1145%2F37401.37406. doi:10.1145/37401.37406.

[11] K. Shinoda, I. Noda, E. Oyama, A pedestrian dynamics simulator for wearable navigation (????).

[12] S. A. Ahmed, D. P. Dogra, S. Kar, P. P. Roy, Trajectory-based surveillance analysis: A survey, IEEE transactions on circuits and systems for video technology 29 (2018) 1985–1997.

[13] R. Laxhammar, G. Falkman, Online learning and sequential anomaly detection in trajectories, IEEE transactions on pattern analysis and machine intelligence 36 (2013) 1158–1173.

[14] Y. Yu, X. Si, C. Hu, J. Zhang, A review of recurrent neural networks: LSTM cells and network architectures, Neural Comput. 31 (2019) 1235–1270.

[15] R. Kemker, M. McClure, A. Abitino, T. Hayes, C. Kanan, Measuring catastrophic forgetting in neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.

[16] H. Ma, Y. Sun, J. Li, M. Tomizuka, C. Choi, Continual multi-agent interaction behavior prediction with conditional generative memory, IEEE Robotics and Automation Letters 6 (2021) 8410–8417.

[17] P. Buzzega, M. Boschini, A. Porrello, D. Abati, S. Calderara, Dark experience for general continual learning: a strong, simple baseline, Advances in neural information processing systems 33 (2020) 15920–15930.

[18] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, G. Wayne, Experience replay for continual learning, Advances in Neural Information Processing Systems 32 (2019).

[19] F. Zenke, B. Poole, S. Ganguli, Continual learning through synaptic intelligence, in: International Conference on Machine Learning, PMLR, 2017, pp. 3987–3995.

[20] J. Schwarz, W. Czarnecki, J. Luketina, A. Grabska-Barwinska, Y. W. Teh, R. Pascanu, R. Hadsell, Progress & compress: A scalable framework for continual learning, in: International Conference on Machine Learning, PMLR, 2018, pp. 4528–4537.

[21] G. Lombardo, A. Poggi, M. Tomaiuolo, Continual representation learning for node classification in power-law graphs, Future Generation Computer Systems 128 (2022) 420–428.

[22] F. Bergenti, E. Franchi, A. Poggi, Agent-based social networks for enterprise collaboration, in: 2011 IEEE 20th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE, 2011, pp. 25–28.

[23] A. Poggi, M. Tomaiuolo, P. Turci, Extending jade for agent grid applications, in: 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE, 2004, pp. 352–357.

[24] A. Negri, A. Poggi, M. Tomaiuolo, P. Turci, Dynamic grid tasks composition and distribution through agents, Concurrency and Computation: Practice and Experience 18 (2006) 875–885.

[25] F. Bergenti, G. Caire, S. Monica, A. Poggi, The first twenty years of agent-based software development with jade (2020).

[26] A. Poggi, Agent based modeling and simulation with actomos., in: WOA, 2015, pp. 91–96.

[27] F. Bergenti, A. Poggi, M. Tomaiuolo, An actor based software framework for scalable applications, in: International Conference on Internet and Distributed Computing Systems, Springer, 2014, pp. 26–35.

[28] M. Pellegrino, G. Lombardo, M. Mordonini, M. Tomaiuolo, S. Cagnoni, A. Poggi, Actodemic: A distributed framework for fine-grained spreading modeling and simulation in large scale scenarios., in: WOA, 2021, pp. 194–209.

[29] G. Lombardo, P. Fornacciari, M. Mordonini, M. Tomaiuolo, A. Poggi, A multi-agent architecture for data analysis, Future Internet 11 (2019) 49.

[30] E. Franchi, A. Poggi, M. Tomaiuolo, Blogracy: A peer-to-peer social network, in: Censorship, Surveillance, and Privacy: Concepts, Methodologies, Tools, and Applications, IGI global, 2019, pp. 675–696.

[31] E. Franchi, A. Poggi, M. Tomaiuolo, Social media for online collaboration in firms and organizations, in: Information Diffusion Management and Knowledge Sharing: Breakthroughs in Research and Practice, IGI Global, 2020, pp. 473–489.

[32] G. Lombardo, A. Poggi, Actornode2vec: An actor-based solution for node embedding over large networks, Intelligenza Artificiale 14 (2020) 77–88.

[33] G. Lombardo, A. Poggi, A scalable and distributed actor-based version of the node2vec algorithm., in: WOA, 2019, pp. 134–141.

[34] M. Pellegrino, G. Lombardo, S. Cagnoni, A. Poggi, High-performance computing and abms for high-resolution covid-19 spreading simulation, Future Internet 14 (2022) 83.

[35] G. Lombardo, M. Pellegrino, M. Tomaiuolo, S. Cagnoni, M. Mordonini, M. Giacobini, A. Poggi, Fine-grained agent-based modeling to predict covid-19 spreading and effect of policies in large-scale scenarios, IEEE Journal of Biomedical and Health Informatics 26 (2022) 2052–2062.

[36] G. A. Agha, Actors: A model of concurrent computation in distributed systems., Technical Report, Massachusetts Inst of Tech Cambridge Artificial Intelligence Lab, 1985.

[37] Y. Yu, X. Si, C. Hu, J. Zhang, A review of recurrent neural networks: Lstm cells and network architectures, Neural computation 31 (2019) 1235–1270.