Gossip Learning in Edge-Retentive Time-Varying Random Graphs with Node Churn

Mina Aghaei Dinani
University of Neuchatel,and HES-SO Valais
Switzerland
mina.aghaei@unine.ch

Antonio Di Maio
University of Bern
Switzerland
antonio.dimaio@unibe.ch

Gianluca Rizzo
HES-SO Valais, Switzerland, and
University of Foggia, Italy
gianluca.rizzo@hevs.ch

***Abstract*—Fully distributed learning schemes based on opportunistic exchanges among nodes, such as Gossip Learning (GL), have recently attracted attention due to their superior scalability, robustness, and enhanced privacy protection. However, their performance has only been characterized in static or application-specific trace-driven mobility scenarios, overlooking the issue of understanding how the structure of the interactions among nodes over time affects the learning process. To address this gap, we propose a new assessment approach for GL in dynamic settings, based on two novel classes of time-varying random graphs, which extend Erdős-Rényi (ER) and Barabási-Albert (BA) random graphs to characterize generic real-world dynamic networks (e.g., social or wireless networks), while accounting for node churn and the rate at which the graph evolves. Evaluating GL on such time-varying graphs allows us to abstract the relationship between the key parameters of GL algorithms, the communication and topology patterns, and the learning performance from factors specific to the experimental contexts, generalizing our findings to a large class of real-world networks. Simulation results show that the sparser the graph, the higher the positive impact edge dynamicity has on GL mean accuracy and convergence time. Surprisingly, we observe that in networks with the same average connectivity degree, regardless of their nodes' attachment style, a higher edge persistence reduces GL mean accuracy and convergence time, highlighting the value of a varied interaction among nodes. Finally, results show that real-world networks that exhibit preferential attachment can preserve a better GL performance in terms of mean accuracy and convergence time than random networks (i.e., ER-like), even under high node churn and edge dynamicity.**

I. Introduction

Distributed Machine Learning (ML) [1] has proven its potential to significantly enhance the performance and efficiency of real-world dynamic systems, such as bioinformatics networks [2], transportation networks [3], and social networks [4]. By distributing the computational load across multiple nodes in a dynamic network, we can process large volumes of data more quickly and accurately. This is particularly beneficial in Artificial Intelligence of Things (AIoT) environments, where each agent generates vast amounts of data and needs to be processed in real-time [5]. However, many distributed schemes face scalability issues relying on infrastructure-based information exchanges or a centralized coordination server. This has increased interest in fully decentralized approaches, exemplified by GL [6], [7]. GL adopts a server-less, fully decentralized model training approach, emphasizing knowledge transfer among agents through direct, peer-to-peer exchanges of models. They scale effectively with the number of agents, as each agent contributes to service capacity with its data computing, and communication capacity. Several GL schemes have been proposed for a large variety of learning architectures [7], [8], scenarios, and applications [9], [10], [11]. However, the majority of these schemes assume *static* network topologies (e.g., ring [12] or mesh [13], [8], [14]). Some studies [7] show that, in such static networks, GL converges and delivers comparable accuracy to that of distributed schemes with centralized aggregation, such as Federated Learning (FL). However, these results do not apply to dynamic settings, where the network topology changes over time, such as in Vehicle-to-Vehicle (V2V) networks or robot swarms. Several studies assess GL in dynamic scenarios (e.g., from fully-connected [15] to volatile vehicular networks [16], [17]) showing that convergence to high-accuracy models, even in trace-driven mobility scenarios, is possible. However, these assessments are limited to networks whose topology is induced by trace-based mobility patterns, making it hard to extrapolate findings to general settings and do not offer insights into the relationship between the main system parameters and GL's key performance indicators. Early work [18] studied the performance of GL on random Time-Varying Graphs (TVG)s, characterizing the relationships between connectivity and scale on the global model's convergence and accuracy. However, such works focus on a specific type of random graph where the number of agents remains constant over time and only the connections between agents change, not considering situations where agents join or leave the network (*node churn*). Furthermore, such works do not consider *edge persistence* over time, assuming that the connectivity lifetime between agents is independent of the network's historical state. The current literature does not yet provide evidence on how changes in the network, such as agents joining or leaving the network and their connectivity lifetime, impact the performance of GL algorithms. This work's main contributions are:

- We introduce Time-Varying Erdős-Rényi (TVER), and Time-Varying Barabási-Albert (TVBA) random graphs, which extend random ER and BA graphs, respectively, to generate time-varying scenarios by modeling node churn and edge persistence.
- We study GL performance on TVER, and TVBA random graphs by simulating GL over general synthetic scenarios that capture real-world topological dynamicity without relying on trace-driven connectivity patterns.
- Through simulations, we study the relationship between GL's accuracy and convergence performance in time-

varying graphs under variable network density, node churn, and edge persistence. Results show that the mean accuracy and convergence time performance of GL in a wide range of diverse TVBA graphs is robust to topological dynamicity regarding node scale, churn, and edge-connectivity persistence and magnitude. Conversely, results show that GL over TVER graphs are sensitive to changes in network parameters, such as in small and sparse networks that most substantially benefit from low edge retention.

II. System model

We consider a set of mobile nodes modeling, e.g., smartphones, UAVs, or connected vehicles, moving in a region of space. Each node is endowed with an ML model that is identical in architecture across all nodes, needs to be trained, and is used by each node to perform a specific inference task. In addition, each node possesses a set of data points, denoted as *local dataset*, which we assume remains constant over time. Nodes can exchange information through direct wireless Device-to-Device (D2D) communications. Communication between two nodes may occur whenever they are in *contact*, i.e., within the transmission range of each other (which we assume to be the same for all nodes). We assume these exchanges to be instantaneous and without losses or errors, though our approach can be easily generalized to account for more realistic communication patterns. We assume time to be discrete and divided into equal-sized intervals called *slots*, indexed with $t \in \mathbb{N}$. In this work, we model the evolution over time of the connectivity graph of nodes as Time-Varying Graphs (TVG). At each time slot t, the set of vertices V_t models the set of nodes present in the area at time slot t. Each edge $e_{i,j} \in E_t$ models the existence of a bidirectional wireless channel between $v_i, v_j \in V_t$. We assume the connectivity graph is constant within each time slot, but not necessarily between consecutive slots. The resulting TVG $G = \{G_t = (V_t, E_t) : t \in \mathbb{N}\}$ is thus a sequence of graphs, each associated with a time slot. As for the evolution of V_t over time, we consider two cases. In *closed systems*, V_t does not change over time, whereas in *open systems* it changes due to nodes joining and leaving the graph. In what follows, we assume that the mean arrival and departure rates of nodes from the given region are the same (and equal to λ nodes per slot) so that the mean number of nodes in the given region is constant over time.

III. Basics of Gossip Learning Operation

In this work, we assume nodes employ the Gossip Learning (GL) scheme to train their models collaboratively. In what follows we take as a reference implementation of the GL protocol the one in [16], though our methodology of analysis is very general and extends to other variants and implementations of distributed learning schemes.

We assume that, at the start of the scheme, all nodes present in the region are endowed with an *initial local model instance*, identical for all nodes, and with random coefficients. Starting from slot $t = 0$, at every slot the algorithm proceeds through three *phases*, synchronized across all nodes. In the *training* phase, each node trains its local model instance over its local dataset. In the second phase, each node exchanges its local model instance with its neighbors. In the third phase (*merging*), each node *merges* the models received from neighbor nodes with its local model instance by computing a weighted average of them to generate a *meta-model* (e.g. similarly to what done in FedAvg [19]). In this work, we assume the *Decentralized Powerloss* (Dfed Pow) strategy [16], where the weight assigned to each model in the merging task is determined by its loss, computed over the context-specific validation set. Such a choice for model weights has proven among the most effective in several GL training scenarios [10], [16]. These three phases are repeated until a termination criterion is met. This can be in terms of minimum model accuracy or minimum increment in model accuracy over a given number of slots. Among the key performance indicators are the mean and the distribution of the model accuracy at convergence and the mean time to converge [10], [16].

IV. Time-varying Random Graph Models

As already noted, GL schemes are known to depend strongly on the spatio-temporal patterns of exchanges among nodes. To enable a scenario-independent, systematic, and controlled assessment of the impact of network structure and its evolution in space and time on the distributed training process. To this end, we elaborate two models of time-varying random graphs by extending the well-known ER [20] and BA [21] models to dynamic scenarios. BA graphs exhibit the scale-free property, with nodes of high degree playing the role of hubs. they emerge naturally in a variety of settings, including heterogeneous wireless sensor mesh networks, where few nodes have better coverage and high transmit power than other nodes. The ER model instead produces a random network with a more uniform degree distribution, and it is suitable for modelling, e.g. wireless sensor networks with high homogeneity in the spatial distribution of devices and their transmission range. Studying the properties of learning over these graphs is relevant because such generic time-varying graphs simplify the assumptions about the structure, evolution, and properties of realistic networks, offering the flexibility to adjust and control independently and accurately such parameters as the number of nodes, the arrival and departure rate, the connection patterns, and the frequency and duration of node interactions.

A. Time-Varying Erdős-Rényi (TVER) Graph

In a static ER random graph, known as a binomial graph, each pair of nodes has a fixed probability p of being connected by an edge [20]. We propose an extension to the ER model, called TVER, by combining a sequence of ER graphs with a memory feature to model different rates of temporal evolution accurately. It provides a more realistic representation of many real-world systems, where the future state depends on the past states. Specifically, in a closed system, the parameters that characterize a TVER graph are:

- *Edge creation probability* p_1. It is the probability that an edge is present between two vertices at time $t > 0$, conditioned to the fact that it was not present at time $t-1$,

i.e., $\forall t \in \mathbb{N}, \forall e \in E_t : \mathbb{P}[e \in E_t | e \notin E_{t-1}] = p_1$.

- *Edge persistence probability* p_2. It represents the probability that an edge is present between two vertices at time $t > 0$, conditioned to the fact that it was already present at time $t-1$, i.e., $\forall t \in \mathbb{N}, \forall e \in E_t : \mathbb{P}[e \in E_t | e \in E_{t-1}] = p_2$.

The choice of p_1 and p_2 thus determines the mean node degree, the equivalent edge probability, and the mean duration in time of an edge between any two nodes. Parameters p_1 and p_2 thus tune the speed at which the graph evolves. The graph is memoryless (i.e., it consists of a sequence of independent ER graphs) when $p_1 = p_2$. If $p_1 > p_2$, having an edge at $t-1$ between two nodes makes it *less* likely to have it at time t. This models scenarios in which the end of an exchange between two mobile nodes is due to the two nodes moving in opposite directions (e.g., on opposite lanes of a road), making it unlikely for the two nodes to come in contact again soon. On the other hand, when $p_2 > p_1$ the graph evolves by keeping some memory of its past states. The algorithm which describes the evolution of an TVER graph in a closed system is thus as follows. At slot $t = 0$, an ER graph is created, with connection probability $p_0 \geq 0$. Then, at each $t > 0$, independently for each couple of vertices, if an edge already existed at $t-1$, it will persist with probability p_2 and disappear otherwise. Conversely, if there was no edge between these two nodes at $t-1$, there will be an edge at t with probability p_1. In open systems, in general, the graph at t has new nodes that were not there at $t-1$, and it does not contain some nodes that were in the graph at $t-1$. The algorithm for building the graph at t given the churn and the graph at $t-1$ is thus the same as in the closed system case. The only difference is in the fact that nodes that arrived at t have no links and that whenever a node present at $t-1$ exits the graph, all its edges are not present anymore in the graph at t. The algorithm that defines a TVER graph evolution over time implies the presence of a transient, which depends on p_0 but also p_1 and p_2. The following result derives a property for these graphs in both open and closed systems once the transient is exhausted:

Theorem 1. *It is given a TVER random graph $G_t = (V_t, E_t)$, $t \geq 0$, where v is the mean number of vertices in the graph. Then, at the mean-field limit, for any $p_0 \geq 0$, $p_2 < 1$, and $\lambda \geq 0$, the mean node degree $k(t)$ converges towards its stationary value k, as in Equation 1.*

$$k = \frac{(v-1)p_1}{1 + (p_2 - p_1)\left(\frac{\lambda}{v} - 1\right)} \tag{1}$$

Proof. Let $v_t = |V_t|$ denote the number of nodes at time slot t, and $\epsilon_t = |E_t|$ the number of edges in the graph at that slot. We assume churn occurs with intensity λ nodes per time slot. Let N_t denote the maximum number of edges at time slot t (i.e., $N_t = v_t(v_t - 1)/2$). We have that, $\forall t \in \mathbb{N}$, $\epsilon_t = p_2\left(\epsilon_{t-1} - \lambda\frac{\epsilon_{t-1}}{v_{t-1}}\right) + p_1\left(N_t - \epsilon_{t-1} + \lambda\frac{\epsilon_{t-1}}{v_{t-1}}\right) = (p_2 - p_1)\left(\epsilon_{t-1} - \lambda\frac{\epsilon_{t-1}}{v_{t-1}}\right) + p_1 N_t$. Starting from this, one can write the differential equation which governs the evolution over time of ϵ_t. We assumed that the system is stationary for what concerns the process of node arrival and departure. When the mean number of nodes v is large, the mean field approximation holds, and an equation for the stationary state of the system can be thus derived. Let N denote the mean number of possible edges in the graph, and $\epsilon \in \mathbb{R}$ the mean number of edges in the graph at the mean-field limit. Therefore, $\epsilon = \left(p_2 - p_1 - p_2\frac{\lambda}{v} + p_1\frac{\lambda}{v}\right)\epsilon + p_1 N$. As the mean probability of attachment p is ϵ/N, we can express p_2 as a function of the other variables, obtaining $p_2 = \frac{1 - p_1(N/e - 1 + \lambda/v)}{1 - \lambda/v} = \frac{p_1\left(1 - \frac{1}{p} - \frac{\lambda}{v}\right) + 1}{1 - \frac{\lambda}{v}}$. Finally, $k = (v-1)p$ gives (1). □

Note that the network converges towards a full mesh for $p_2 = 1$ and $p_1 > 0$, i.e., $\lim_{t \to +\infty} \epsilon_t \to v(v-1)/2$. Another issue is determining whether each graph in a TVER is an ER graph at each time slot t. Trivially, this holds $\forall t$ when $p_1 = p_2$, i.e., when there is no edge retention. In the more general case, for closed systems where $p_1 \neq p_2$, the following result (Theorem 2) holds.

Theorem 2. *It is given a TVER random graph $G_t = (V_t, E_t)$, $t \geq 0$, and $p_2 < 1$. Then, in the closed system case, for any $p_0 > 0$ at the stationary state, the node degree has a mean-field limit distribution equal to that of an ER graph.*

Proof. (sketch) The result follows from Theorem 1 and from the fact that if $(1-p_1+p_1e^t)^{v-k}(1-p_2+p_2e^t)^k$ is the Moment Generating Function (MGF) of the node degree at the mean-field limit, the equation $(1-p_1+p_1e^t)^{v-k}(1-p_2+p_2e^t)^k = (1-p+pe^t)^v$ admits a single positive solution in p, which is thus the connection probability of the ER graph at the mean-field limit. □

Thus, for $p_1 \neq p_2$, a TVER without churn is a sequence of graphs which are ER at the mean field limit.

B. Time-Varying Barabási-Albert (TVBA) Graph

The second family of time-varying graphs is an extension of BA random graphs, characterized by the scale-free properties and based on the preferential attachment mechanism [22]. This generates the "rich get richer" phenomenon, by which the graph exhibits a power-law degree distribution. In what follows, we first introduce a generalized algorithm for building a BA random graph, which accepts as input m (the mean node degree of the graph), and the parameter h which tunes the degree to which the node attachment is preferential. Initially, the set of nodes V_0 in the system is partitioned into two subsets. The first set $\mathcal{A}$ comprises $m+1$ fully connected random nodes, while $\mathcal{B}$ is composed of all the remaining nodes, and they are all isolated. Then in turn, for each node $x \in \mathcal{B}$, the node is removed from it and added to $\mathcal{A}$. Then, for every couple (x, u) with $u \in \mathcal{A}$, $u \neq x$, an edge is added between the two nodes with a probability

$$\mathbb{P}[(x,u) \in E] = h\frac{1}{|\mathcal{A}|} + (1-h)\frac{\deg(u)}{\sum_{w \in \mathcal{A}} \deg(w)} \tag{2}$$

Where $\deg(x)$ denotes the degree of node x, $|\mathcal{A}|$ is the cardinality of $\mathcal{A}$. Note that when $h = 1$ the graph is ER.
The algorithm that describes the evolution of TVBA is thus as follows. At step $t = 0$, a BA graph is generated. At every step $t > 0$, a new graph is generated as follows. Let C_{t-1} be the set of all possible edges at $t - 1$. From it we derive a new set C'_{t-1} obtained by removing from C_{t-1} all those possible edges for which at least one of the two vertices left the system due to churn. Then for each possible edge C'_{t-1}, with probability p_2 it keeps in t the same status as in $t - 1$ (i.e. if it was not present in the graph at $t - 1$, it will not be present at t, and conversely). Otherwise, that edge will be added to the graph at t following the BA algorithm, i.e. with a probability given by (2). For all the other possible edges at t, due to the arrival of new nodes, they are too added to the graph at t with the probability given in 2. Note that this algorithm defines a TVG with the same statistical properties as its static counterpart, including node degree distribution.

V. Numerical assessment

A. Simulation Setup

We assess the performance of the GL schemes on the proposed TVGs by considering a set of V nodes which need to perform an inference task. We adopted the MNIST datasets [23] for the handwritten digit recognition task, characterized by 10 classes and an image size of 28×28 pixels. Given the reference dataset, we have defined a *global dataset* as a fixed-size subset of the reference dataset in each setup. The global dataset is the set of all data points available for the training process in a given scenario. Thus, it directly impacts the maximum accuracy achievable in a given scenario. In our experiments, to consider scenarios in which collaborative model training brings benefits over simple local model training on behalf of each node, we have tuned the global dataset size to have a locally trained model whose accuracy is, on average, substantially lower than that achievable by centralized training over the union of the training sets of all nodes. Thus, unless otherwise stated, for MNIST the size of such a global dataset has been set to 600 data points. In each setup, the global dataset is partitioned among all nodes in the scenario. Thus, each node is endowed with a *local dataset*, whose size is equal for all nodes. Thus, the choice of keeping the global dataset size constant has been made to assess the impact of information fragmentation on GL performance.

The data points of local datasets are i.i.d. and randomly selected from the global dataset. We employed a random uniform sampling from the global dataset to create local datasets without replacement. To each global dataset, we associated a *global test set* obtained from the original dataset by random sampling, ensuring no data point of the global dataset is included and whose size is 20% of the global dataset size. Thus, the global dataset and test set are disjoint.

We assumed every node in the system executes a Convolutional Neural Network (CNN) model, whose architecture and hyperparameters (identical for all nodes) are designed for effective shape feature extraction. We assume that nodes use supervised models training with categorical cross-entropy loss, batch size of 32, the momentum of 0.9, and a 10^{-4} learning rate to perform the inference task. Unless otherwise stated, in GL, each node trains its local model over one epoch at every slot before exchanging it with neighbors. In a dynamic network, connections may be lost and change, so training one epoch at a time makes the learning process more robust to these changes, as it is less time-consuming than training a large number of epochs.

To better appreciate the performance of GL algorithms in dynamic network settings, in our experiments we also consider the following ML training schemes:

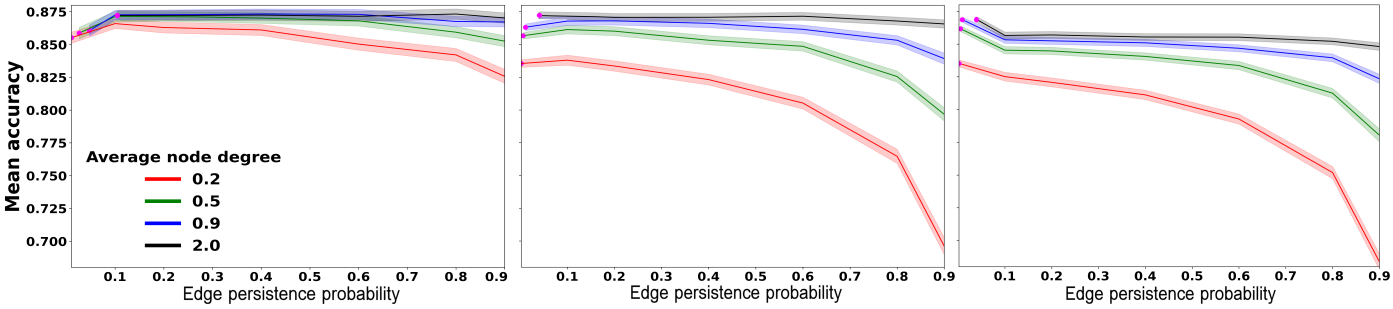
- *Centralized Learning (CL)*. This approach involves a central server endowed with a training set coinciding with the scenario's global dataset.
- *Individual Local Training (ILT)*, in which each node trains its local model instance using only its local dataset without exchanging data or models with other nodes or with a centralized server.
- *Federated Learning* [19], a client-server approach in which clients train their model locally. In it, at each iteration, a parameter server collects the models from all participants, aggregates them, and redistributes the resulting aggregated model to all participants. Aggregation is done via model averaging, as in the FedAvg approach [19].
- *GL over Static random graphs (GLS)*. This learning scheme is the GL scheme described in Section III, applied on static random graphs.

Whenever node churn is present, we assume the number of nodes that arrive and leave at each time slot to be distributed according to two independent Poisson point processes, with the same intensity (churn rate) of λ nodes per slot. We also assume every node has the same probability of leaving the system at every time slot. Nodes that join the network possess a local dataset and a local model instance. We follow the same set of criteria, similar to the closed system, to initialize the local model instance and create the local dataset. The size of the local dataset remains constant across all nodes within the network. In the open system, new arrivals add information to the network. However, when nodes leave the network, they do so with their data. This guarantees that the number of data points in each slot is equal to the size of the global dataset.

In our experiments, we continue simulations until either the system converges or we reach the maximum iteration limit. We define the gossip convergence time as the number of time slots needed for the system to converge. We assumed our schemes to have converged when the mean accuracy of local models across all nodes does not increase by more than 0.5% over an interval of 30 slots.

B. Closed system

In the first set of experiments, we considered scenarios without node churn, and we evaluated the effect of dynamicity on the performance of GL algorithms on TVER and TVBA graphs in terms of mean accuracy, for different values of edge persistence probability p_2, number of nodes, and average node degree.



(a) 20 nodes (b) 80 nodes (c) 100 nodes

Fig. 1: Mean accuracy at convergence as a function of the speed of evolution of the graph (measured by the edge persistence probability p_2) for TVER in the closed system case. The dots at the beginning of each curve indicate those configurations in which edge persistence and creation probability are equal, i.e. the memoryless TVER case. The bands around each curve denote the 95% confidence intervals.

TABLE I: Performance at convergence of the considered algorithms on TVER graphs with 40 nodes and average node degree equal to one. Closed system case.

Algorithm	Convergence time	Acc	Loss	F1
Centralized Learning	60	0.87	0.45	0.87
FedAvg	160	0.85	0.57	0.84
Individual Local Training	52	0.38	2.8	0.46
GL on Static ER	81	0.67	1.1	0.69
GL, $p_2 = 0.02$	215	0.845	0.62	0.82
GL, $p_2 = 0.1$	224	0.86	0.55	0.84
GL, $p_2 = 0.9$	209	0.85	0.59	0.82

TABLE II: Spearman's correlation coefficient matrix of system parameters and performance metrics for GL schemes on TVBA and TVER random dynamic graphs for the closed system case. The 95% confidence interval is at most 2%.

Performance metric	Number of nodes		Node degree		Edge persistence	
	TVBA	TVER	TVBA	TVER	TVBA	TVER
Accuracy	-0.13	-0.19	0.09	0.45	-0.17	-0.42
Convergence time	0.08	0.14	-0.14	-0.17	-0.1	-0.3

Figure 1 shows the impact of edge persistence probability on mean accuracy at convergence for different network sizes and average node degrees. As expected, for the same network size, with increasing average node degree, the mean accuracy increases, too, reaching values very close to those achieved via federated learning and centralized learning schemes. This suggests that in dense, connected networks, the spreading of knowledge among nodes is effective in greatly reducing the performance gap with respect to server-based, centralized training schemes. In addition, in denser networks, the performance is less sensitive to the speed at which the network evolves. Instead, it is evident that in less dense networks, where knowledge delivery takes place mainly through store-carry-and-forward of ML models, instead of store-merge-and-forward, mean accuracy at convergence is not only inferior (as a lower node degree brings to a lower rate of knowledge transfer) but also highly sensitive to the speed at which the graph evolves, as knowledge transfer is now directly implemented through node mobility. Interestingly, in these cases, the performance of GL benefits from a higher dynamicity (lower edge persistence probability). However, for small network sizes, there seems to be an optimum rate of network change, beyond which mean accuracy starts decreasing, albeit slightly. This feature, which disappears for larger network sizes and thus for smaller local datasets at each node, suggests the existence of a tradeoff between exploration (which involves the creation of new edges and exchange of new information through network dynamics) and exploitation (the stable exchange of models among neighbours through the persistence of existing edges). It assumes significant importance in networks characterized by lower average node degree and larger network size, where information availability is limited. In these scenarios, nodes derive greater benefits from establishing new connections and acquiring new information in each iteration, as opposed to maintaining existing edges. This underscores the critical role of dynamic connections and the adaptability of nodes in sparse networks.

Table I reports the mean convergence time, as well as the mean accuracy, mean loss, and F1 score, for each of the considered algorithms. Together with Figure 2, these results show the value of collaborative learning. Indeed, GL schemes whether on static or dynamic graphs, consistently outperform individual local training. Accuracy of GL schemes on TVER graphs matches individual local training accuracy at the beginning of the training process, but eventually aligns with the accuracy levels of centralized learning and FedAvg. In addition, all GL schemes on TVER graphs outperform the GL scheme on static graphs, thus further illustrating the value of dynamicity. The box charts in Figure 2 depict the uniform performance distribution across all nodes when dynamicity is high and information spreads freely. On the other hand, the variance in performance is higher when dynamicity is low, $p_2 = 0.9$.

Table I illustrates that the impact of evolution in speed is negligible, as the difference between the performance metrics of the GL scheme on TVER graphs with three different p_2 values is comparable at convergence time, and they only slightly differ in convergence speed. Another key element that affects the performance of distributed learning schemes is the structure of the network and, thus, of interaction among nodes. To this end, for both TVER and TVBA graphs we conducted a series of experiments on scenarios characterized by various combinations of the following parameters: the

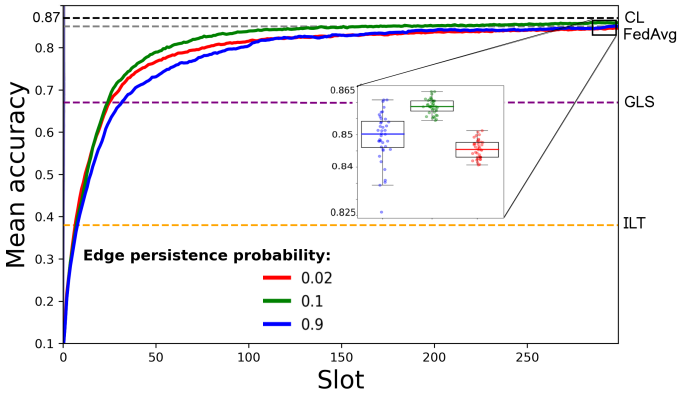


Fig. 2: Mean accuracy over training time (in the number of slots) in a closed-system TVER random graphs with 40 nodes, and average node degree one for GL schemes against the baselines, for different values of edge persistence probability p_2. The zoom-in highlights the distribution of mean accuracy across nodes at convergence. The 95% confidence intervals report an error of at most 2%.

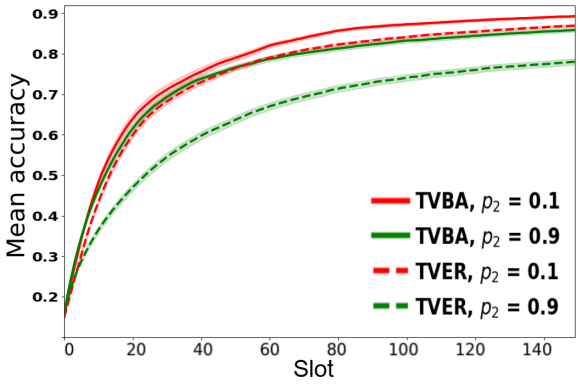


Fig. 3: Evaluation of the effect of edge persistence probability p_2 and network structure on the mean accuracy versus slots for GL schemes in a closed system. Each scenario contains 1000 nodes, each having 4 samples, and an average degree of 1. All curves are associated with a 95% confidence interval.

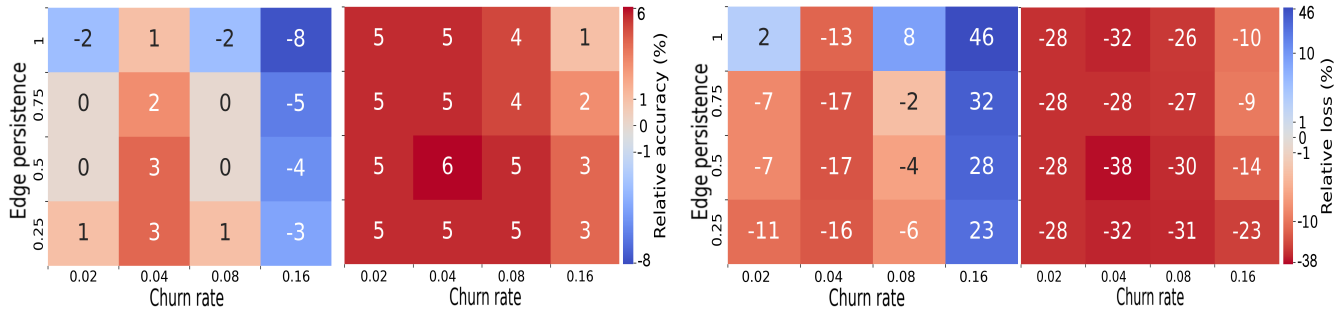
number of nodes $|V| \in \{5, 10, 20, 40, 80, 100, 1000\}$, the average node degree $\in \{0.2, 0.5, 0.9, 1, 2, 4\}$; and the edge persistence probability $p_2 \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Table II shows a weak correlation between the number of nodes and GL performance metrics on both graphs. It aligns with the observation in Figure 1. This is demonstrated by the consistent performance levels when the average node degree is 2, regardless of the number of vertices in the network. It shows that data fragmentation has a minimal impact on performance in a connected network. This phenomenon can be attributed to the fact that information can flow freely and efficiently between nodes in a connected network, mitigating the effects of data fragmentation. Across all the scenarios implying TVER graphs, there is a moderate positive correlation between node degree and accuracy. This evidence supports what Figure 1 suggested. A higher average node degree is associated with higher accuracy. On the other hand, edge persistence probability has a moderate negative correlation with accuracy and convergence time. Increasing edge persistence probability decreases dynamicity. When edges remain stable over time, information spreads more slowly. As an iteration of local learning is performed at each slot, slower evolution tends to train a lot over local data, exposing it to the risk of overfitting. Consequently, networks with higher edge persistence values may reach convergence faster but at the cost of accuracy. These findings align with Figure 1 observations. We discerned only weak correlations between the performance metrics and network parameters in TVBA graphs. This implies that GL algorithms utilizing TVBA graphs exhibit robustness to variations in network parameters. This is likely because information dissemination across multiple high-degree nodes can be accomplished more rapidly, enhancing the system's resilience to changes in network parameters.

We also examined how dynamicity affects the network diameter and influences overall performance. Figure 3 shows GL schemes on TVBA graph with $p_2 = 0.1$, and diameter equal to 3 outperform others. GL schemes performance on TVBA graphs with $p_2 = 0.9$ matches those using TVER graphs with $p_2 = 0.1$, as in both cases, the length of the longest shortest path between any two vertices in the graphs is almost equal, 9, and 11 respectively. Using TVER graphs with $p_2 = 0.9$, the diameter is 70, and it leads to a 15% performance deficiency. In TVER graphs, node connections are formed randomly, and the node degree is consistent. In the majority of these scenarios, dynamism is advantageous as it decreases the diameter. However, in TVBA graphs, diameter does not change much by changing the value of p_2. A shorter diameter implies more efficient information dissemination, reducing the steps required for information to traverse across the network. This can lead to less information loss and distortion during transmission.

C. Open system

Most real-world networks experience some degree of churn. In an open system, the dynamicity is influenced by two parameters: the probability of edge persistence p_2 and churn rate λ. Similar to the closed system, Figures 4a and 4c show the performance of GL schemes on TVER graphs diminishes with an increase in the value of edge persistence probability. Moreover, the introduction of churn and having new information in the system enhance the performance marginally by up to 3% more than centralized learning when the churn rate is equal to 0.04, but it is mainly detrimental to the learning process. On the other hand, GL schemes on TVBA graphs leverage the introduction of new information by churn, enhancing the model's accuracy by up to 6% more than centralized learning. Furthermore, these models demonstrate increased reliability, with a maximum reduction in loss value of 38%, as shown in Figure 4 b, and d. Similar to closed systems, the performance remains robust across changes in network configuration.

In general, GL schemes on TVBA graphs demonstrate superior stability compared to those on TVER graphs. This could be attributed to the preferential attachment-based connection of new arrivals to nodes with higher degrees. As observed in previous experiments conducted in closed systems, a higher node degree contributes to improved performance, thereby enabling new nodes to acquire a high-performance model. In contrast, in TVER graphs, new arrivals randomly join existing nodes in the scenario, regardless of their performance. In the



(a) % variation in accuracy, TVER (b) % variation in accuracy, TVBA (c) % variation in loss, TVER (d) % variation in loss, TVBA

Fig. 4: Percentage variation in accuracy relative to the mean accuracy achieved by centralized learning (Figure 4a and 4b), and percentage variation in loss value in comparison to the average loss achieved by centralized learning (Figures 4c and 4d) over TVER and TVBA graphs, as a function of edge persistence probability p_2 and churn rate λ, in the open system case. 20 nodes, mean node degree equal to one. 98% confidence interval of at most 2%.

event of a departure in TVER graphs, all edges connected to the departing nodes are simply removed, and the system does not adapt to the loss of information. However, in TVBA graphs, nodes connected to the departing node establish new links to existing nodes based on preferential attachment. This mechanism maintains network stability through changes, ensuring the network's resilience to information loss.

VI. Conclusion

This work introduces a novel approach to assess the performance of gossip-based learning schemes in dynamic network environments, based on an extended version of the well-known ER and BA graphs to account for network churn and evolution rate. Through extensive simulations, we provide insights into the impact of network dynamics on GL efficiency and effectiveness. Results demonstrate that GL schemes employing TVBA graphs exhibit resilience to changes in network topology, preserving performance across various configurations. Conversely, the efficacy of GL schemes on TVER graphs is influenced by changes in the network's parameters and dynamics, particularly in low-density networks. It underscores the importance of considering network dynamicity in distributed machine learning and provides insights into the trade-offs between network density, node churn, and edge persistence.

References

[1] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *ACM Comput. Surv.*, vol. 53, mar 2020.

[2] M. Marchetti-Bowick, J. Yin, J. A. Howrylak, and E. P. Xing, "A time-varying group sparse additive model for genome-wide association studies of dynamic complex traits," *Bioinformatics*, vol. 32, pp. 2903–2910, 06 2016.

[3] M. Gendreau, G. Ghiani, and E. Guerriero, "Time-dependent routing problems: A review," *Computers & Operations Research*, vol. 64, pp. 189–197, 2015.

[4] C. Wang, J. Tang, J. Sun, and J. Han, "Dynamic social influence analysis through time-dependent factor graphs," in *2011 International Conference on Advances in Social Networks Analysis and Mining*, pp. 239–246, 2011.

[5] S.-H. Sim and Y.-S. Jeong, "Multi-blockchain-based iot data processing techniques to ensure the integrity of iot data in aiot edge computing environments," *Sensors*, vol. 21, no. 10, 2021.

[6] R. Ormándi, I. Hegedűs, and M. Jelasity, "Gossip learning with linear models on fully distributed data," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 556–571, 2013.

[7] I. Hegedűs, G. Danner, and M. Jelasity, "Decentralized learning works: An empirical comparison of gossip learning and federated learning," *Journal of Parallel and Distributed Computing*, vol. 148, pp. 109–124, 2021.

[8] V. Zantedeschi, A. Bellet, and M. Tommasi, "Fully decentralized joint learning of personalized models and collaboration graphs," in *International Conference on Artificial Intelligence and Statistics*, pp. 864–874, PMLR, 2020.

[9] C. Li, G. Li, and P. K. Varshney, "Decentralized federated learning via mutual knowledge transfer," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1136–1147, 2022.

[10] M. A. Dinani, A. Holzer, H. Nguyen, M. A. Marsan, and G. Rizzo, "Vehicle position nowcasting with gossip learning," in *2022 IEEE WCNC*, pp. 728–733, 2022.

[11] A. A. Alkathiri, L. Giaretta, S. Girdzijauskas, and M. Sahlgren, "Decentralized word2vec using gossip learning," in *23rd Nordic Conference on Computational Linguistics (NoDaLiDa 2021)*, 2021.

[12] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[13] A. G. Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "Braintorrent: A peer-to-peer environment for decentralized federated learning," *arXiv preprint arXiv:1905.06731*, 2019.

[14] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi, "Personalized and private peer-to-peer machine learning," in *International Conference on Artificial Intelligence and Statistics*, pp. 473–481, PMLR, 2018.

[15] N. Majcherczyk, N. Srishankar, and C. Pinciroli, "Flow-FL: Data-driven federated learning for spatio-temporal predictions in multi-robot systems," in *IEEE ICRA*, pp. 8836–8842, IEEE, 2021.

[16] Dinani, Mina Aghaei and Holzer, Adrian and Nguyen, Hung and Marsan, Marco Ajmone and Rizzo, Gianluca, "Gossip learning of personalized models for vehicle trajectory prediction," in *2021 IEEE WCNC*, pp. 1–7, 2021.

[17] M. A. Dinani, A. Holzer, H. Nguyen, M. A. Marsan, and G. Rizzo, "A gossip learning approach to urban trajectory nowcasting for anticipatory ran management," *IEEE Transactions on Mobile Computing*, pp. 1–17, 2023.

[18] A. Di Maio, M. A. Dinani, and G. Rizzo, "The upsides of turbulence: Baselining gossip learning in dynamic settings," MobiHoc '23, (New York, NY, USA), p. 376–381, ACM, 2023.

[19] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.

[20] P. Erdős and A. Rényi, "On random graphs i," *Publicationes Mathematicae Debrecen*, vol. 6, pp. 290–297, 1959.

[21] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[22] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of modern physics*, vol. 74, no. 1, p. 47, 2002.

[23] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov. 1998.