Gossip Learning: Off the Beaten Path

Lodovico Giaretta
KTH Stockholm
lodovico@kth.se

Šarūnas Girdzijauskas
KTH Stockholm
sarunasg@kth.se

***Abstract*—The growing computational demands of model training tasks and the increased privacy awareness of consumers call for the development of new techniques in the area of machine learning. Fully decentralized approaches have been proposed, but are still in early research stages. This study analyses gossip learning, one of these state-of-the-art decentralized machine learning protocols, which promises high scalability and privacy preservation, with the goal of assessing its applicability to real-world scenarios.**

Previous research on gossip learning presents strong and often unrealistic assumptions on the distribution of the data, the communication speeds of the devices and the connectivity among them. Our results show that lifting these requirements can, in certain scenarios, lead to slow convergence of the protocol or even unfair bias in the produced models. This paper identifies the conditions in which gossip learning can and cannot be applied, and introduces extensions that mitigate some of its limitations.

I. Introduction

In recent years, new massively-distributed data sources have emerged, such as smart sensors, smartphone apps and connected devices. This shift towards decentralized data production poses new challenges to traditional machine learning approaches, which have focused on processing the data in a central location, such as a datacenter. Collecting data from decentralized sources can be hard and costly, due to storage and bandwidth limitations and due to the speed and scale at which new data is produced. Furthermore, these sources may contain sensitive information, which poses additional burdens and limitations to its collection, due to the increased regulations and consumer awareness regarding data privacy.

Decentralized, peer-to-peer machine learning protocols can alleviate some of these issues. These protocols can scale more easily than a centralized approach: as new data sources are added, the amount of data to process grows, but so do the available computing power and network bandwidth, thanks to the participation of these new devices in the protocol. Furthermore, decentralized protocols represent an interesting starting point for the development of privacy-preserving systems, by limiting the amount of information that has to be shared.

One state-of-the-art approach in this field is gossip learning [1]. This decentralized machine learning protocol has been shown to be very efficient, scalable and flexible. It has been successfully applied to many different machine learning problems, including classification with SVMs [1], k-means clustering [2] and matrix decomposition [3]. However, to the extent of the authors' knowledge, gossip learning has not been implemented in any industrial application, and has only been tested in restricted experimental conditions, which raises questions on its performance in real-world scenarios.

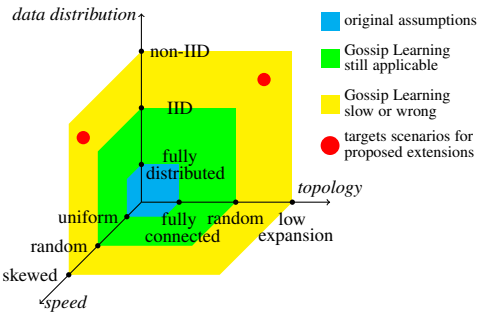

Fig. 1: The space of configurations explored in this study.

Thus, the goal of this study is to assess the applicability of the protocol in real-world conditions, by testing it outside its "beaten path". To do this, we first identify the three assumptions, stated in previous papers on this technique, that are likely to be violated in non-controlled environments. These are 1) that each device stores a single data point (referred to as the *fully-distributed* data model), 2) that each device is able to communicate with all others (*unrestricted* network topology), and 3) that the processing and communication speeds of the devices are homogeneous. We then simulate the protocol on different workloads as we lift these assumptions to different levels, in order to identify the circumstances in which the protocol keeps working, and those in which it fails.

According to our results, gossip learning shows poor performance on restricted communication topologies, only maintaining its original convergence speed in those networks that exhibit good expansion properties. Furthermore, some common real-world network topologies, such as power-law and community-based ones, can lead to very slow convergence and even incorrectly biased models, when paired with non-IID data distributions. The results also show that gossip learning, while able to handle heterogeneous distributions of the communication speeds of the nodes, fails to converge to the correct models when these speeds are correlated with the distribution of the data.

To widen the scope of applicability of gossip learning, we also propose and test different extensions to the original

This work was funded in part by H2020 Marie Curie ITN RAIS (Grant agreement no. 813162)

Algorithm 1 Skeleton of the gossip learning protocol

```
procedure MAIN
    currentModel ← INITMODEL()
    lastModel ← currentModel
    loop
        WAIT(Δ)
        p ← RANDOMPEER()
        SEND(p, currentModel)
    end loop
end procedure
procedure ONMODELRECEIVED(m)
    currentModel ← UPDATE(MERGE(m, lastModel))
    lastModel ← m
end procedure
```

protocol. One allows each node to train on multiple data points per node, even non-IID distributed, achieving rapid convergence. The others mitigate the erroneous model biases that the original protocol exhibits in some common real-world scenarios, such as data-dependent communication speeds and data-dependent power-law communication topologies.

Thus, the contribution of this paper is twofold: charting the area of applicability of gossip learning, identifying its main limitations, and enhancing the protocol, by proposing novel extensions that widen its real-world applicability. Fig. 1 visualizes the parameter space explored and the results obtained in each setting, along with the position of the proposed extensions in the parameter space.

II. The Gossip Learning Protocol

Originally introduced by Ormándi et al. [1], gossip learning is an asynchronous protocol designed to train a global model over decentralized data using a gossip communication approach [4]. Its simplicity, flexibility and efficiency make it an interesting starting point for the development of next-generation decentralized machine learning systems.

Conceptually, starting from a common initialization, multiple models perform random walks over the network, learning from the data stored in each device visited. This is accomplished by having the nodes update the received models on their local data and then gossip them out to a randomly-chosen peer. To speed up the learning process, the models are also merged with each other along their walks.

Algorithm 1 shows the generic skeleton of the protocol, which can be applied to different kinds of machine learning tasks. The main loop performed by each device is very simple: a random peer is chosen among the other participants in the network, and the current model is gossiped to it. When a device receives a new model, it merges it with the last model previously received and then updates the resulting model by performing local training. The resulting model is stored locally for prediction and for gossiping with peers, until a new model is received and the process is repeated.

The UPDATE and MERGE procedures can vary greatly depending on the kind of model to train. This work focuses on supervised models that can be trained using Stochastic Gradient Descent (SGD), with a decaying learning rate and regularization. In this scenario each model includes, in addition to its weights $\mathbf{w}$, a timestamp t, which represents the age of the model, defined as the number of data points it has been trained on. The timestamp is used to compute the decaying learning rate, which is applied to the problem-specific gradient, as shown in Algorithm 2. To implement the MERGE operator, we use a simple average of the model weights.

Algorithm 2 Skeleton of the model update using SGD

```
λ                                   ▷ The regularization parameter
x, y          ▷ The features and label of the local data point
procedure UPDATE(m)
    w, t ← m
    η ← 1/(λ · t)                   ▷ The decaying learning rate
    w ← (1 − η · λ) · w + η·GRADIENT(w, x, y)
    return (w, t + 1)
end procedure
```

III. Limitations of Gossip Learning and Potential Solutions

Unfortunately, gossip learning, as developed in state-of-the-art research, holds some assumptions that limit its applicability to real-world scenarios. It is thus fundamental to understand whether it is possible to lift them, and to what extent. In fact, none of these assumptions presents a "binary choice". Rather, there are different levels at which each can be lifted. This paper presents extensive simulations to identify what parameters affect the correctness and performance of the protocol, and how far it can be pushed before hitting a hard limit.

A. Fully Distributed Data Model

The first limitation is the fully-distributed data model, where each device is assumed to own a single, private data point. This may be the case in some circumstances, such as in certain recommender systems; however, there are many scenarios where a single user might have multiple useful data points, such as text completion and image classification.

To address this limitation, we propose a simple extension of the original protocol. The UPDATE function, which performs one step of the learning process, can be called multiple times on different data points. In the case of stochastic gradient descent, multiple SGD steps can be performed sequentially. Thus, after training on a node with d data points, the model $(\mathbf{w}, t)$ will become $(\mathbf{w}', t + d)$.

With this simple extension enabling training on multiple data points, one can question *whether the size of the dataset sample stored on each node affects the behaviour of the protocol*. In general, the number of data points on each device i follows some distribution $D_i \sim D$, which in real-world scenarios could be very skewed. It would be reasonable to believe that the characteristics of the sample size distribution might affect the behaviour of the protocol, and that the protocol might thus be limited to some of them. This study analyses many different scenarios, with the results showing that the convergence speed and accuracy of gossip learning are not affected by this choice.

When each node stores a sample with multiple data points, another consideration is that *the way the samples are drawn from the overall data set might also influence the behaviour of the protocol*. Intuitively, if samples are IID, the protocol should be expected to behave better than if the samples are non-IID. Our results confirm this intuition, showing, in the latter scenario, a divergence during the early phases of training.

B. Network Connectivity

The second limitation derives from the use of gossip communications to spread the models across the whole network. The robustness and dissemination efficiency of this approach is based on the assumption that each node, at each iteration, can choose its peer uniformly at random from the entire network. This is typically achieved by using a *peer sampling service* [5], which provides each node with a uniform random sample of the network members. Unfortunately, this still requires each device to be able to communicate with any other participant in the protocol. In certain applications, this might be impossible, due to security or privacy limitations. In other cases, it might be inefficient and potentially expensive. Thus, in many real-world scenarios, the devices might be limited to a small, fixed set of neighbours for their communications, based on an underlying *restricted topology*. It is therefore necessary to understand *whether gossip learning works correctly and efficiently on restricted communication topologies*.

Network topologies can present many different characteristics. 1) They can be generated randomly or according to some predefined pattern. 2) The distribution of node degrees can be tight, with a similar number of neighbours for each device, or very wide, following for example a power-law distribution. 3) The network can be more or less robust, based on the amount of redundant paths between the nodes. Also, 4) the network can be more or less well-connected, depending on the lengths of the shortest paths between pairs of nodes. The simulations show that, while the protocol converges to the correct model in all instances, these last two characteristics are critical to ensure a good convergence speed. Topologies with low *expansion*, a metric that includes both robustness and well-connectedness, require more iterations to converge. An Erdős-Rényi graph, presenting low diameter and a high number of alternative paths between any pair of nodes, shows the same performance as a fully-connected topology. The other side of the spectrum is represented by the tree topology, which has only a single route between any two nodes, and presents a quite large diameter of $log(N)$, where N is the number of nodes. According to the simulation results, this topology requires 100 times more iterations to converge. Thus, a "bad" topology can make gossip learning unfeasible for large applications.

Furthermore, in some real-world scenarios, the position of some nodes in the communication topology might be correlated with the data stored in them. For example, a network may present tight communities with similar data points, with a very low number of inter-community links. We show that in this very specific, yet common scenario, the protocol exhibits a very slow convergence rate. The lower the percentage of inter-community links, the longer the time needed for the models to spread through the whole network and reach convergence.

Another harmful interaction of communication topologies and data distribution is shown by power-law networks, where a small number of nodes exhibit a very high degree, while a majority of the devices only have a few neighbours. This kind of topology is known to be very common in social structures, and can thus be expected to appear in gossip learning networks. In this situation, most of the communications need to pass through the high-degree nodes, which act as hubs to keep the network connected. The simulations show that in these conditions the models develop a bias towards the kind of data points stored in the hubs. As such, the trained models result unrepresentative of the real data distribution, rendering the protocol useless.

This kind of bias is a known limitation of natural random walks: the probability of a random walk visiting a node is proportional to its degree [6]. Usually degree-biased random walks, based on the Metropolis-Hastings algorithm [7], represent an effective solution to this problem. This approach can be stated as follows: after choosing the candidate peer to send the model to, consider its degree. If it is lower than the degree of the current node, send the model unconditionally. Otherwise, send it with a probability inversely proportional to its degree. The Metropolis-Hastings algorithm "corrects" the distribution of the random walks by forcing them to spend additional time in low-degree nodes. In the gossip learning case, to make this additional time spent in low-degree nodes meaningful, an additional change is needed: when a node decides to not gossip its model, it performs one more round of local training on it. This counters the bias introduced by hubs, but in certain applications it might cause the models to overfit local data and lose generalization. Furthermore, our simulations show that this approach, while drastically improving the training result, does not completely remove the bias towards the data in high-degree nodes.

For this reason, we experiment with an additional technique, which we name *pass-through gossiping*. This approach consist in making hubs act as "bridges" between low-degree nodes, allowing the latter to indirectly gossip each other and thus hiding the power-law structure of the network. In practice, when node j receives a message from i, it only performs the usual merge and update steps with probability $p(i,j) = \min(1, d_i/d_j)$. Thus, if the sender has lower degree than the receiver, there is a chance the receiver might save the received model as its current model and later propagate it, without going through the usual update and merge operations. This approach provides slightly better results than Metropolis-Hastings, converging more quickly to a model very close to the expected one.

C. Communication Speeds

The third and last limitation of gossip learning regards the communication speeds of the devices. Previous research [1] considers speeds normally distributed, thus creating a scenario where most speeds are concentrated around the mean, with only a few outliers. In many real-world situations, where

Algorithm 3 Protocol extension for data-dependent speeds

```
procedure MAIN
    currentModel ← INITMODEL()                          ▷ Initialization
    lastModel ← currentModel
    for j ∈ Neighbours do
        M_j ← INITMODEL()
    end for
    loop                                                ▷ Main loop
        WAIT(Δ)
        k ← RANDOMPEER()                ▷ Pick received model to use
        currentModel ← UPDATE(MERGE(M_k, lastModel))
        lastModel ← M_k
        p ← RANDOMPEER()                                ▷ Gossip new model
        SEND(p, currentModel)
    end loop
end procedure
procedure ONMODELRECEIVED(m, j)
    M_j ← m
end procedure
```

different types of devices need to cooperate, the speed distribution might be much more heterogeneous.

The considerations regarding this assumption are similar to those presented for the sample size at each node: the communication speeds can be drawn from different kinds of distributions, and it is important to understand *whether the characteristics of the chosen distribution affect the behaviour of the protocol*. We show that gossip learning can handle different distributions, even very skewed, as long as they are independent from the data distribution.

But, when the data distribution and the speed distribution are correlated, the protocol produces a model that is biased towards the subset of samples stored on faster devices. The reason is that these nodes output their models more often than the others, and thus trick the protocol into "thinking" that their data points are more numerous than they actually are.

In particular, this behaviour arises because those nodes that have both fast and slow neighbours receive more models from the former. Thus, the models of the former are propagated more often. This insight helps in defining a mitigation for this issue. If the receiving node could ensure that it receives and processes models picked uniformly at random from its neighbours, without being affected by their speed, the protocol would converge to a correct model.

We thus propose an extension to gossip learning, where each node i has as one model slot M_j for each of its neighbours $j \in N(i)$. When receiving a model from a neighbour j, instead of processing it immediately to update its current model, node i saves it in the corresponding slot M_j. Only when the time to gossip a new model comes, node i picks a random slot M_k and uses the model stored there to perform the MERGE and UPDATE steps. In this way, the receiving node has no bias in its choice. Furthermore, a fast node might choose the model of a slow one multiple times before receiving a new one, thus "boosting" it by propagating it more often than the original slow node could do. Our results show that this extension can completely mitigate the bias introduced by data-dependent speeds.

However, this extension introduces some drawbacks: the memory requirement at each node grows from $O(1)$ to $O(K)$, where K is the number of neighbours of the node, as it needs to store the last model received from each neighbour. In some scenarios, such as resource-constrained IoT networks, this kind of overhead could be unacceptable. Furthermore, this extension requires each node to have a fixed, small set of neighbours. While this is the case in most real-world scenarios, this extension cannot be used with full connectivity and random peer sampling, as used in the original protocol.

IV. METHODOLOGY

A. Machine Learning Algorithms

Support Vector Machines (SVMs) and linear regression were chosen as the algorithms to perform the tests. Many reasons drove this choice: 1) they represent different classes of tasks, namely binary classification and regression; 2) they are simple, well-known and extensively-studied algorithms, and it is thus easy to analyze them and reason on their behaviour; 3) they require little computational effort, allowing a larger number of simulations to be performed in a shorter time and thus allowing this paper to test a wide range of configurations.

To train the SVM, the Pegasos algorithm was chosen [8]. This algorithm is based on the primal formulation of the SVM problem, instead of the more common dual approach. This makes it more appealing for decentralized learning, as the dual approach requires frequent access to the entire dataset in order to train its weights. Furthermore, it is based on stochastic sub-gradient descent, and can be easily embedded in Algorithm 2.

B. Parameters

In order to simulate different real world conditions, three main parameters were modified throughout the experiments. These parameters, each related to one of the main limitations presented, are: 1) the distribution of data points to the nodes 2) the communication topology 3) the distribution of communication speeds among the nodes.

For the first parameter, different data distribution were tested, presenting different characteristics, but all sharing the same average sample size k, allowing a fair comparison. The *fixed size* distribution assigns k data points to each of the N nodes, while the *uniform* distribution picks the size of each sample randomly in the range $[1, 2k-1]$. The Pareto distribution, defined as $p(x) = a \cdot m^a / x^{a+1}$, produces very skewed sample sizes, while a Gaussian distribution with $\mu = k$ represents a middle ground between very homogeneous distributions and very skewed ones.

Performing an exhaustive test of all possible choices for the second parameter, namely the configuration topology, is challenging, due to large amount of different characteristics that these can present. A *full topology* corresponds to the behaviour of the original protocol, were all nodes can contact each other. *Erdős-Rényi* graphs are well-connected random topologies, parameterized by their average node degree. *Barabási-Albert* random graphs present a power-law distribution, with a few

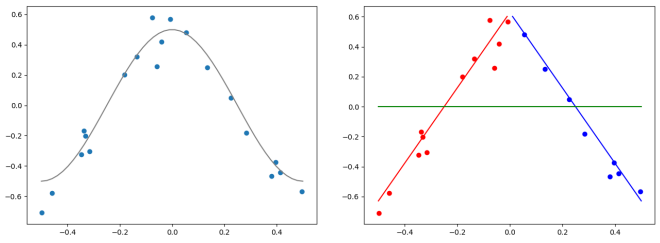

Fig. 2: *(left)* The cosine wave and a sample of the generated data points after adding noise. *(right)* The points with $x < 0$ *(red)* and $x > 0$ *(blue)*, with the linear models fitting them. In green, the best linear regressor for the whole dataset.

hubs and many low-degree nodes, and are parameterized by the minimum node degree m.

The experiments also included other random graphs. One is the graph obtained by randomly connecting a set of n well-connected *communities*, which can be seen as a *planted partition model* [9]. Some notable non-random graphs are also tested, including k*-ary trees* and *rings*.

Similarly to the data distributions, multiple speed distributions were also tested, including constant, uniform random and gaussian speed distributions.

C. Datasets

The datasets were initially chosen from the UCI repository [10]. In particular, SpamBase and WineQuality [11] were used, respectively for binary classification and regression.

Unfortunately, testing some of the target scenarios using these datasets proved difficult. To be able to test the effect of data-dependent topologies and data-dependent speed distributions, it must be possible to split the data set in subsets that show different characteristics in terms of features. Furthermore, it should be easy to analyze the evolution of the models trained on these subsets, in order to understand whether any unfair bias is introduced by the protocol.

In order to fulfill these requirements, we introduce a custom synthetic dataset, based on a scaled cosine wave, defined as $y = 0.5 \cos(2\pi x)$, with $x \in [-0.5, 0.5]$.

By sampling this cosine wave and adding some white Gaussian noise, a dataset suitable for regression tasks can be built. This dataset presents two clearly distinct patterns: for $x < 0$, y grows with x, while for $x > 0$, higher x values correspond to lower y values. Thus, while each of the two sides can be easily learned by a linear regressor, a single model trained on their union would not be able to give a better approximation than $y = 0$, and would thus perform very poorly, as shown in Fig. 2. The same dataset can also be adapted to a classification task, by adding a $G \cdot c$ term to y, where G is the width of the "gap" between the two classes, while $c \in \{-1, 1\}$ is the randomly-chosen class label.

Fig. 3 provides an example of the kind of analysis that can be performed on this synthetic dataset. Internally, the weights of an SVM models represent the components of the normal vector to the hyperplane that separates the two classes to identify. The picture shows the evolution of the angle between these normal vectors and the x axis. An horizontal model would have a vertical normal vector, thus showing an angle of $\pi/2 \approx 1.57$. A higher angle corresponds to a model with a positive slope, while a lower angle identifies a negative slope. In the picture, the red and blue lines correspond to the average model angles for those devices storing points with $x < 0$ and $x > 0$, respectively. Incidentally, this picture confirms one of the statements in Section III-A: when the distribution of data points is not IID, as in this case, where each device only stores points with either $x < 0$ or $x > 0$, the protocol takes longer to converge, as in the initial phases the models trained by different nodes diverge towards their respective solutions.

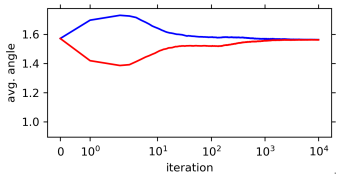

Fig. 3: Angles of the SVM models trained on $x < 0$ *(blue)* and $x > 0$ *(red)* subsets of the cosine-generated dataset.

V. Experimental Results

A. Multiple Data Points per Node

The first set of experiments aims to verify whether the distribution of sample sizes among the nodes affects the performance of gossip learning. To check this, different sample size distributions were tested, the only invariant being the average sample size. The results, shown in Fig. 4, demonstrate that different distributions do not affect the behaviour of the protocol: the same average sample size results in the same convergence speed, no matter how heterogeneous the values of the distribution are.

Furthermore, training on multiple data points provides a clear advantage over the original protocol, as the models see more data in the same number of iterations, and thus converge faster. It must be noted, though, that the number of model merges is still limited to one per node per iteration, which means that having an average of k samples per node does not provide a true k-fold convergence speed increase.

B. Restricted Communication Topologies

The second set of experiments is designed to test the effects of restricted communication topologies on the behaviour of the protocol. To this end, many different topologies were tested. As most topology can be further modified by the choice of specific parameters, different parameter sets were tested for each topology, as shown in Fig. 4.

The results show that well-connected, robust topologies, such as the Erdős-Rényi, Barabási-Albert and community-based graphs, present a convergence speed that matches a fully-connected topology. On the other hand, topologies with high diamater and low link redundancy, such as trees and rings, show a clearly slower convergence.

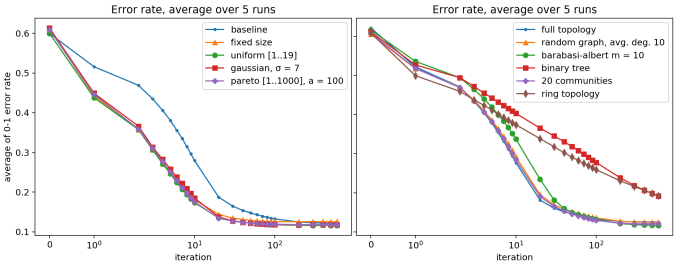

Fig. 4: Comparison of different topologies. Good expanders perform similarly to a fully-connected network, while bad expanders show slow convergence.

It can thus be concluded that gossip learning only provides reasonable performance when applied on topologies with good *expansion*, as this property summarises both low distance and high redundancy. It must be noted though that, given enough time, all topologies eventually converge to the same accuracy.

C. Heterogeneous Communication Speeds

The third set of simulations aims at verifying the suitability of gossip learning for scenarios where the nodes present very different speeds. The results show that assigning speeds randomly, even from a wide range, which converges to the same result in the same number of iterations, as if all speeds were equal. This can be seen by comparing the first and second columns of Fig. 5.

On the other hand, the behaviour of the protocol changes dramatically when the speed distribution is correlated with the data distribution. The third column of Fig. 5 shows the behaviour of gossip learning on the cosine dataset in the extreme case in which all nodes storing data points with $x > 0$ (red) are faster than all those storing data points with $x < 0$ (blue). In this case, the model quickly drifts in favor of $x > 0$, converging to a very definite negative slope.

The last column of Fig. 5 shows the performance of the extension introduced in Section III-C to deal with this limitation of the original protocol. It can be seen that the extension does not manage to fully re-establish the symmetry that data-independent speed distributions present. However, in the long term, it succeeds in inducing the models to converge to the expected horizontal hyperplane. Thus, this extensions successfully fulfills its overall goal, allowing the application of gossip learning to scenarios where the speeds of the nodes are unknown and possibly correlated with dataset features.

D. Data-Dependent Community-Based Topologies

The fourth set of simulation analyses a special scenario that may arise when a restricted topology is correlated with a non-uniform data distribution. In particular, when nodes with similar data points form tightly-connected groups, with only a small amount of links connecting separate communities.

In these circumstances, most of the communications happen within a single community, with only a few messages traversing the inter-community links and "contaminating" other groups. As can be seen in Fig. 6, moving from a random topology to a community-based one, or decreasing the percentage of inter-community links available for contamination, causes the models to diverge more and more towards the local optimum of each community.

However, after a certain number of iterations, the models start to converge towards the correct global optimum. This number of iterations does not depend on the percentage of inter-community links, as both the second and third column of Fig. 6 show the tipping point to be slightly after 100 iterations. A clue to the motivation is given by the last column of Fig. 6, which shows the evolution of the training when the timestamps of the models are artificially capped after 100 iterations. In these circumstances, the convergence never happens, and the models keep maintaining the angle reached at iteration 100.

The reason for this behaviour is that SGD-based gossip learning uses a decaying learning rate that is computed based on the timestamp of each model. Setting an upper bound to the timestamp is equivalent to setting a lower bound to the learning rate. Thus, it can be deduced that, in the community-based scenario, the convergence of the protocol is entirely determined by the decrease of the learning rate.

In the initial phases of the protocol, when the learning rate is still quite high, any "foreign" models that enters a community is quickly "erased", thanks to the high influence of the local updates. Later in the process, when the learning rate is low, the local updates become negligible, and the gossip learning is reduced to an averaging protocol. When this happens, the infrequent averaging of the local models with "foreign" ones is sufficient to lead the system to convergence.

E. Data-Dependent Power-Law Topologies

The fifth and last set of simulations is concerned with another special interaction between a restricted topology and a non-uniform data distribution. Specifically, the case in which, in a power-law topology, the nodes with high degree present a different data distribution than those with low degree.

Fig. 7 compares the evolution of the models on an Erdős-Rényi graph, with the degrees of all nodes very close to the overall average, and on a Barabási-Albert power-law graph, with the top 50% nodes in terms of degree storing data points with $x > 0$ (red), while the other nodes store points with $x < 0$ (blue). In the latter case, the models become quickly biased towards hubs.

The reason for this behaviour is that, according to Algorithm 1, the model gossiped by a node is the result of merging and training the last two received models. Due to their position, hubs receive, in each iteration, a large number of models from their low-degree neighbours. All but the last two of these models are discarded and will not contribute to any of the models gossiped in the following iteration. On the other hand, models trained by hubs are almost never discarded, as a vast majority of the potential recipients are low-degree nodes, which rarely receive more than two models in a single iteration. Thus, the overall effect is a strong bias towards the data stored in hubs.

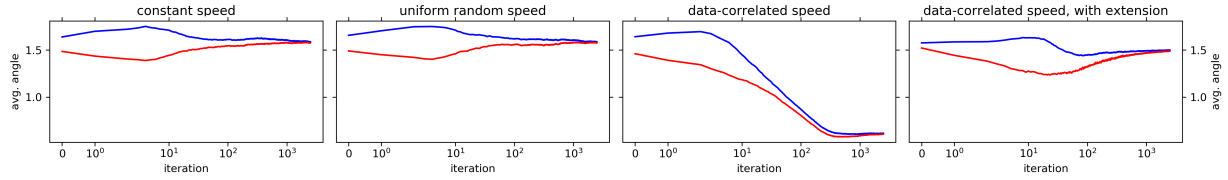

Fig. 5: Evolution of SVM models on the cosine-generated dataset, with different speed distributions

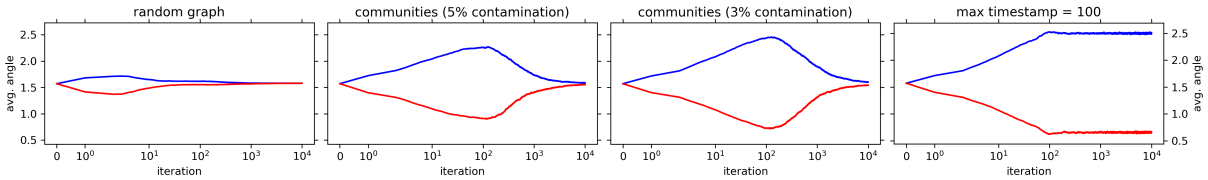

Fig. 6: Evolution of SVM models on the cosine-generated dataset, with random vs community based graphs. Last column: effect of capping the timestamp (and thus the learning rate) in a community-based scenario.

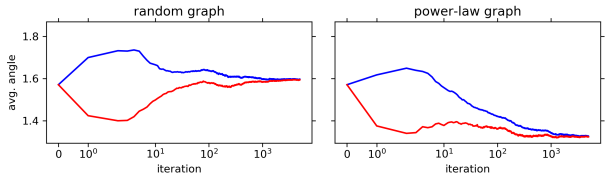

Fig. 7: Error rate of the models, on random vs power-law topologies. The latter produce erroneous biases.

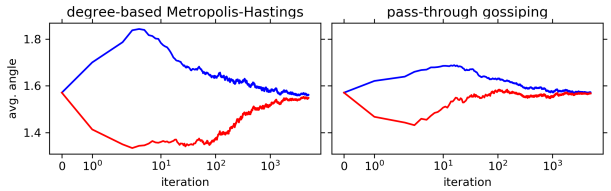

Fig. 8: Error rate of the models, trained with different extensions meant to mitigate the skewness of power-law topologies.

Fig. 8 shows the behaviour of the protocol with each of the extensions introduced in Section III-B. Metropolis-Hastings converges to a good result, similar to a non-power-law topology. However, in the first phases of training, this approach shows a more pronounced divergence between the models of the two subsets of nodes, especially on the low-degree side. This behaviour, induced by the additional local training steps, reinforces the conjecture that this approach might cause overfitting in certain scenarios. The pass-through gossiping approach exhibits the most accurate converged model and the lowest divergence during training, thus showing that directly countering the power-law nature of the topology, instead of its effects, provides the best results.

VI. Related Work

In [12], the authors also introduce the possibility for gossip learning to store multiple data points at each node and to only communicate over a restricted network topology. However, their research focuses on the comparison between gossip learning and federated learning [13], a massively-distributed but centralized machine learning technique. Thus, the authors of [12] do not consider many of the scenarios presented in this paper.

Some of the limitations of gossip learning presented in this paper relate to broader issues in the field of gossip communications and have thus been analysed by many previous works [14], [15]. However, those works focus on generic aggregation problems. Gossip learning, on the other hand, includes a continuous training process that interacts with the aggregation phase, leading to more complex behaviours that are outside the scope of most previous works. This has been shown, for example, in Section V-D, where the dynamics of the learning rate affect the convergence of the protocol.

Regarding the problem of achieving good information dissemination over a restricted communication topology, Khelghatdoust et al. [16] propose a technique to build an efficient random overlay over a restricted network, by routing communication through multiple hops. The resulting overlay, being a random graph, allows efficient gossip learning, as shown in Section V. The drawback of this approach is the need to route the messages through intermediate nodes. In the context of efficient information broadcasting, Kyasanur et al. [17] develop an approach to identify those nodes that are critical in achieving good dissemination. Similar techniques could be used to tune the performance of critical nodes.

To deal with data-dependent heterogeneous speeds, *pull-based* gossip communications [4] represent an alternative to the protocol extension presented in this paper. In pull-based gossip, nodes do not push messages to their neighbours. Rather, they pull messages from them. The drawback of this approach is the requirement for two-way request-response communications, that are more susceptible to packet losses and network delays, compared to the “fire and forget” push-based approach. Thus, pull-based gossip and the extension proposed in this paper provide different tradeoffs in order to guarantee correctness, and may thus be suitable for different applications.

VII. Future Work

While this paper maps the behaviour of gossip learning on a wide range of scenarios, there are many other conditions that need to be tested to ensure the applicability of the protocol to a larger number of real-world settings. In particular, this study did not model any failure condition. Previous work has modelled failures in gossip learning [1], but only within the scope of the strong assumptions of the original protocol. Furthermore, there are still configurations where gossip learning cannot be used effectively, such as low-expansion or community-based topologies.

Multiple extensions for gossip learning have been developed in previous research, including support for concept drift [18] and model compression [12]. Additional research is needed to merge these extensions with those presented in this work. Providing a single, widely-usable gossip learning algorithm would greatly simplify the deployment of decentralized learning in real-world applications.

Finally, to the extent of the authors' knowledge, no research has studied the impact of malicious devices on gossip learning. An attacker could try to abuse the protocol to either extract private data stored by a target device, or to bias the training towards an adversarial objective. The former could be achieved by sending specifically-crafted models to the target device, and observing the changes in the output model. The latter objective could be reached through model poisoning, with techniques similar to those employed against federated learning [19].

VIII. Conclusions

This paper analyzed the applicability of gossip learning to real-world scenarios. Three main limitations were identified in its original formulation, that significantly limit its applicability: the fully distributed data model, the requirement for full connectivity and the assumption of homogeneous communications speeds. Each of these was analyzed, in order to understand its impact, and potential extensions to expand the applicability of the protocol were proposed, where possible.

We show that gossip learning can be extended to handle multiple data points per node and that its performance is not affected by the distribution of sample sizes across the network. The results also show that the protocol can be used in networks with restricted topologies, without affecting the quality of the trained models. Unfortunately, the convergence can be extremely slow when the topology presents low expansion properties, potentially rendering gossip learning unfeasible in certain applications. The protocol can also handle nodes with different communication speeds, as long as these speeds are distributed independently from the features of the dataset.

Unfortunately, gossip learning is not able to correctly handle networks whose characteristics are correlated with the features of the dataset. This paper explored three such cases: 1) community-based networks with different subsets of the data in each community, 2) power-law topologies where the hubs have a different data distribution than the other nodes, and 3) networks in which the communication speed distribution is correlated with the dataset distribution. In the first case, the protocol converges to a correct model, but only after a high number of iterations, due to initial divergence, and this study found no mitigation to this problem. In the other two cases, the original protocol provides incorrect results, but we were able to provide extensions that mitigate these issues.

Overall, it appears that state-of-the-art gossip learning presents shortcomings that limit its successful deployment in real-world scenarios. This paper identifies these shortcomings and addresses some of them by suggesting potential solutions and research directions. However, more study is needed to clear the path for the use of gossip learning in uncontrolled environments.

References

[1] 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, Feb. 2013.

[2] Á. Berta and M. Jelasity, "Decentralized Management of Random Walks over a Mobile Phone Network," in *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, Mar. 2017, pp. 100–107.

[3] I. Hegedűs, Á. Berta, L. Kocsis, A. A. Benczúr, and M. Jelasity, "Robust Decentralized Low-Rank Matrix Decomposition," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 4, pp. 62:1–62:24, May 2016.

[4] A. Montresor, *Gossip and Epidemic Protocols*. American Cancer Society, 2017, pp. 1–15.

[5] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen, "Gossip-based peer sampling," *ACM Trans. Comput. Syst.*, vol. 25, no. 3, Aug. 2007.

[6] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger, "Sampling techniques for large, dynamic graphs," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, April 2006, pp. 1–6.

[7] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 04 1970.

[8] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: primal estimated sub-gradient solver for svm," *Mathematical Programming*, vol. 127, no. 1, pp. 3–30, Mar. 2011.

[9] A. Perry and A. S. Wein, "A semidefinite program for unbalanced multisection in the stochastic block model," in *2017 International Conference on Sampling Theory and Applications*, July 2017, pp. 64–67.

[10] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[11] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decis. Support Syst.*, vol. 47, no. 4, pp. 547–553, Nov. 2009.

[12] I. Hegedűs, G. Danner, and M. Jelasity, "Gossip learning as a decentralized alternative to federated learning," in *Distributed Applications and Interoperable Systems*. Springer, 2019, pp. 74–90.

[13] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," *arXiv:1602.05629 [cs]*, Feb. 2016.

[14] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, Oct 2003, pp. 482–491.

[15] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *ACM Trans. Comput. Syst.*, vol. 23, no. 3, pp. 219–252, Aug. 2005.

[16] M. Khelghatdoust and Š. Girdzijauskas, "Short: Gossip-based sampling in social overlays," in *Networked Systems*. Springer, 2014, pp. 335–340.

[17] P. Kyasanur, R. R. Choudhury, and I. Gupta, "Smart gossip: An adaptive gossip-based broadcasting service for sensor networks," in *2006 IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, Oct 2006, pp. 91–100.

[18] I. Hegedűs, R. Ormándi, and M. Jelasity, "Gossip-based learning under drifting concepts in fully distributed networks," in *2012 IEEE Sixth International Conference on Self-Adaptive and Self-Organizing Systems*, Sep. 2012, pp. 79–88.

[19] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. B. Calo, "Analyzing federated learning through an adversarial lens," *CoRR*, 2018.