Boosting Methods for Federated Learning

Roberto Esposito[1], Mirko Polato[1] and Marco Aldinucci[1]

[1]Dipartimento di Informatica, Università di Torino, Corso Svizzera 185, 10145 Torino

Abstract

Federated Learning (FL) has been proposed to develop better AI systems without compromising the privacy of final users and the legitimate interests of private companies. Initially deployed by Google to predict text input on mobile devices, FL has been deployed in many other industries. Since its introduction, Federated Learning mainly exploited the inner working of neural networks and other gradient descent-based algorithms by either exchanging the weights of the model or the gradients computed during learning. While this approach has been very successful, it rules out applying FL in contexts where other models are preferred, e.g., easier to interpret or known to work better.

This paper proposes to leverage distributed versions of the AdaBoost algorithm to acquire strong federated models. In contrast with previous approaches, our proposal does not put any constraint on the client-side learning models and does not rely on inner workings of the learning algorithms used in the clients. We perform a large set of experiments on ten UCI datasets, comparing the algorithms in six non-iidness settings. Results show that the approach is effective, in the case of an IID setting, results are often near to the theoretical optimum (i.e., the performances of AdaBoost on the complete dataset). In case of non-IID settings, results very much depend on the severity of the non-IIDness.

Keywords

federated learning, cross-silo, boosting, adaboost, ensemble learning

1. Introduction

Recent years have been characterized by crucial advances in artificial intelligence and machine learning systems, by the widespread availability of massive computational resources, and by the availability of huge datasets. The consequent deployment of AI and ML methods throughout many industries has been a welcome innovation that generated, nonetheless, newfound concerns about the fairness of the results and the privacy of the involved data. As a result, it is often the case that data is dispersed into many isolated islands, and ML practitioners are forbidden by laws and by the legitimate owners from collecting, fusing, and ultimately using the data to improve their systems. While protecting the privacy of users and the competing advantages

SEBD 2023: 31st Symposium on Advanced Database System, July 02–05, 2023, Galzignano Terme, Padua, Italy

† This work has been partially supported by the European PILOT project. The European PILOT project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No.101034126. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Italy, Switzerland, Germany, France, Greece, Sweden, Croatia and Turkey.

roberto.esposito@unito.it (R. Esposito); mirko.polato@unito.it (M. Polato); marco.aldinucci@unito.it (M. Aldinucci)

http://www.di.unito.it/~esposito (R. Esposito); http://www.di.unito.it/~polato (M. Polato); http://www.di.unito.it/~aldinuc (M. Aldinucci)

0000-0001-5366-292X (R. Esposito); 0000-0003-4890-5020 (M. Polato); 0000-0001-8788-0829 (M. Aldinucci)

CEUR Workshop Proceedings (CEUR-WS.org)

of companies is arguably a fair objective, it nonetheless hampers the development of learning models that, by leveraging all the available data, could make a difference in the quality of life of many people who are subjected to the decisions made using AI systems.

Federated Learning (FL) has been proposed by McMahan et al. [1] as a way out of this conundrum, i.e., as a way to develop better AI systems without compromising the privacy of final users and the legitimate interests of private companies.

FL is a learning paradigm where multiple parties (clients) collaborate in solving a machine learning task using their private data under the coordination of an aggregator (a.k.a. server or coordinator). Each client's local data is not exchanged or transferred to any participant. The learning happens in rounds where model updates are computed by clients in insulation using local and private data, then aggregated on the server, then broadcast to the clients for the next round.

There are two main federated settings: cross-device and cross-silo. In cross-device FL, the parties can be edge devices (e.g., smart devices and laptops); they can be numerous (order of thousands or even millions). Parties are considered not reliable and with limited computational power. In the Cross-silo FL setting, the involved parties are instead organizations; the number of parties is limited, usually in the range [2, 100]. Given the nature of the parties, it can also be assumed that communication and computation are no real bottlenecks.

Since its introduction [1], Federated Learning mainly exploited the inner working of neural networks and other gradient descent-based algorithms by either exchanging the weights of the model or the gradients computed during learning. While this approach has been very successful, it rules out applying FL in contexts where other models would be preferred, either because they are more interpretable or known to work better. For instance in the case of medical studies, it is often the case that data comes in tabular form and examples are not numerous and distributed among several medical centers that need to respect hard privacy constraints. Also, medical doctors often require to be able to interpret the inferred models. In these situations decision trees or rule based system are often justifiably preferred to neural networks, but they cannot be readily applied without collecting the data in one single place (e.g., [2]), which makes the whole process hard or impossible to implement due to the aforementioned privacy constraints.

This is a position paper based on the work in [3], where we proposed a series of cross-silo FL algorithms for classification based on distributed versions of the AdaBoost algorithm [4, 5, 6, 7, 8] allowing gradient-free federated learning. The algorithms pose minimal constraints on the learning settings of the clients, thus allowing a federation of models not specifically designed for FL, such as decision trees and SVMs. While there is no technical barrier to using our approach in cross-device federated learning settings, we have not conducted experiments to clarify the issue. Our intuition is that the approach will best work with reliable clients that own many examples, and when communication cost is not high. We, therefore, believe that they are best suited for cross-silo settings and leave to future work investigating alternatives more kin to cross-device environments.

The main contributions of this work are:

i) we propose two new FL algorithms inspired by distributed AdaBoost literature, namely DistBoost.F and PreWeak.F;

ii) we introduce a third algorithm (AdaBoost.F) purposely developed for FL;
iii) we present a comprehensive evaluation of our solutions on ten UCI datasets and 6 data distribution settings.

For reproducibility purposes, all the code used to perform the experiments in this paper is available at https://github.com/ml-unito/federation_boosting.

2. Related Works

Ensemble Learning copes with the problem of strengthening the performances of a learning algorithm by iterating it and combining the results. Ensemble Learning is often employed by practitioners because it requires almost no parameters and can be used along with off-the-shelf algorithms to obtain strong models that are usually very robust to overfitting [6]. It is not surprising then that, at the beginning of this century a large swat of research has been devoted to the topic and that many flavors of ensemble learning have been proposed during those years (e.g., Bagging [9], Boosting and its variants [5], Stacking [10], ECOC [11], etc.). In this context, the original boosting algorithm from Schapire [12] is fundamental because by constructively solving the weak learnability problem [12] spawned massive interest in the field and posed the basis for the development of AdaBoost [5], arguably the best-known algorithm in the field. The main idea in Schapire's boosting algorithm [12], and hence in AdaBoost, under the assumption that the base learning algorithm (the weak learner) will always strictly better than random guess, one can leverage the distribution of the examples to force the weak learner to focus on specific portions of the examples space. This can be then used to drive down the error of the ensemble exponentially fast. AdaBoost appears particularly interesting as a candidate tool for FL, as it effectively combines classifiers which may be learned independently by the FL clients. Furthermore, it could be argued that, as long as at least one of the clients can find a model which is slightly better than the random guess over the complete dataset, AdaBoost should be able to drive the error of the ensemble on the training set to its theoretical minimum no matter other factors (such as the possible non-iidness of the data distribution).

Most of the FL literature focuses on gradient-based methods with very few exceptions. [13] proposes Federated Forest, a lossless federated version of the classical Random Forest (RF) algorithm for vertically partitioned data. In this method, trees are built on node splits selected by the aggregator that repeatedly asks clients for the impurity index and picks the minimum. Federated Forest guarantees privacy preservation mainly using features/labels' encoding. However, label encoding may fail in the case of binary classification tasks. A very different approach to learning RFs is presented in [14] where the federation is managed using Blockchain technology that guarantees security even against adversarial participants. Vertical FL (VFL) is the learning setting in [15] that presents federated algorithm for classification/regression trees based on Multi-Party Computation [16]. The authors also describe possible extensions of the methodology to gradient-boosting trees and linear regression. In [17], the VFL setting is considered in the context of kernel-based methods. The authors propose a privacy-preserving protocol to build dot-product kernel matrices, showing the technique's effectiveness on top-N recommendation tasks. To the best of our knowledge, we are the first to propose a federated version(s) of AdaBoost where the (weak) classifiers can be induced by any learning algorithm.

As briefly mentioned in the introduction, two of the algorithms presented in this paper are based on a distributed version of AdaBoost, namely DistBoost [7] and PreWeak [8] that we will describe in Section 3. In [18], a distributed agnostic boosting algorithm is described. Differently from AdaBoost, the method uses a non-exponential multiplicative weight update rule that is further adjusted using the *Bregman projection*. Here, we propose a federated adaptation of AdaBoost, and we would argue that a similar methodology may also apply to the approach in [18]. Boosting-based FL has been little studied in the literature. All published works on the topic focus on gradient-boosting trees [19, 20] and most of them are designed for vertically partitioned data [21, 22, 23, 24]. Homomorphic encryption and secret sharing schemes are used to guarantee privacy, with the only exception of [21, 19] that use a differential private approach. The cross-silo setting is considered in both [21] and [24] (decentralized FL).

We differentiate from these previous works because our federated boosting algorithms can be used with any weak learner, and our setting is horizontal FL. Even if our work focuses on a very specific case (classification in a vertical setting) in federated learning, we believe the techiques proposed could be extended and generalized to cover other learning tasks (e.g., regression, clustering, ...) and FL settings.

3. Ensemble Learning based Federated Learning

In this work, we set ourselves in a cross-silo FL setting, we assume that the clients are reliable and have enough computational power as well as a stable and secure connection [25, 26]. With these assumptions, our proposals expects a certain degree of synchronicity between the clients and the aggregator. However, all the proposed techniques can easily handle clients' failure, for instance, by using a timeout on the clients that exclude their participation from that federated round.

In [5] Freund and Schapire formally proved that, provided that the weak learner can induce a decision rule which is consistently better than random guessing, AdaBoost reduces the ensemble error over the training set exponentially fast in the number T of the combined weak models. It is worth emphasizing that this is the only constraint posed by the algorithm. As shown by Freund and Schapire [4], this holds true even when the weak learner behaves adversarially towards the ensemble learner. While this is not relevant in most scenarios, in the federated learning case, the weak learners only work with a subset of the available data. In a sense, it can be thought that malevolent learners try to make the ensemble learner fail on that part of the data (the data they do not own). This argument shows that, as long as at least one client can produce a model better than random guess over the entire dataset, a distributed version of AdaBoost, modified to guarantee that no information about the local dataset is exchanged, should be able to drive the ensemble error to its minimum exponentially fast. This is the main idea on the basis of our work.

In the past, there have been several attempts to build distributed versions of AdaBoost [7, 8]. In these works, the main aim was to distribute the computation; there was no attempt to provide privacy over the data and, indeed, all clients were supposed to hold the complete dataset. In [3], we have shown how to adapt two of these algorithms to work in a FL setting and also proposed an additional original algorithm. The main contributions were to provide

mechanisms to cope with the fact that different clients hold different portions of the dataset, which have repercussions over the way the distribution over the examples is handled (e.g., how weights are normalized). For a detailed description of the working of the algorithms, we refer to the original publication [3]; for details of their implementation in actual (not simulated) FL environments, please refer to [27, 28]; here we only provide a brief summary of the ideas on which the algorithms are based. A common trait of the algorithms is that care is put in ensuring that the necessary statistics over the examples are computed in a privacy preserving way. To do that, all clients maintain unnormalized statistics over the examples and communicate them to the aggregator. The aggregator collect all statistics and uses them to compute a common normalization factor. The normalization factor can then be used to properly compute the ϵ_t and α_t values that are central to the working of algorithms based on AdaBoost. The α_t terms are then broadcasted to all clients so that they can update their local set of statistics and the process repeats.

DistBoost.F At each round, all clients build weak hypotheses over their local dataset; the hypotheses are sent to the aggregator that forms a bagged ensemble and uses that as the weak hypotesis for the current round. That weak hypothesis is transferred to each client so that they can use it to measure the performances of the newly learnt hypothesis and communicate them back to the aggregator (needed to let everyone to update the weights distribution).

PreWeak.F In an initial step all clients train an AdaBoost classifier over their local datasets. In this step a fixed number T of weak hypotheses are built in each client without exchanging any information with the aggregator. Once all clients complete, all the learnt weak hypotheses are transmitted to the aggregator, which starts a global AdaBoost process. In this step, only the weak hypotheses already learnt in the previous step are considered as candidates to be added to the ensemble. At the end of each round the selected hypothesis is communicated to the clients so to allow the computation of the statistics necessary to maintain the global distribution of weights.

AdaBoost.F At each round each client builds a weak hypothesis which is communicated to the aggregator. The aggregator distribute these hypotheses to the clients so that they can compute the necessary statistics over the local dataset. These statistics are then used to pick the best weak hypothesis that is then added to the ensemble.

4. Experiments

We compare the federated algorithms introduced in [3], namely DistBoost.F, PreWeak.F, and AdaBoost.F, with the centralized algorithm SAMME [29] (multiclass AdaBoost). For all methods, we fix the number of weak learners (federated rounds) $T = 300$. As weak learners, we employ Decision Trees with up to 10 leaves (as in [29]). However, it is worth mentioning that the proposed algorithms are agnostic to the choice of the weak learner; better still, there is nothing preventing building a system where each client adopts a different model. The simulated federation contains 10 clients, which is a standard choice [26] in the cross-silo setting. We assumed that all clients correctly participated in all rounds during the simulation.

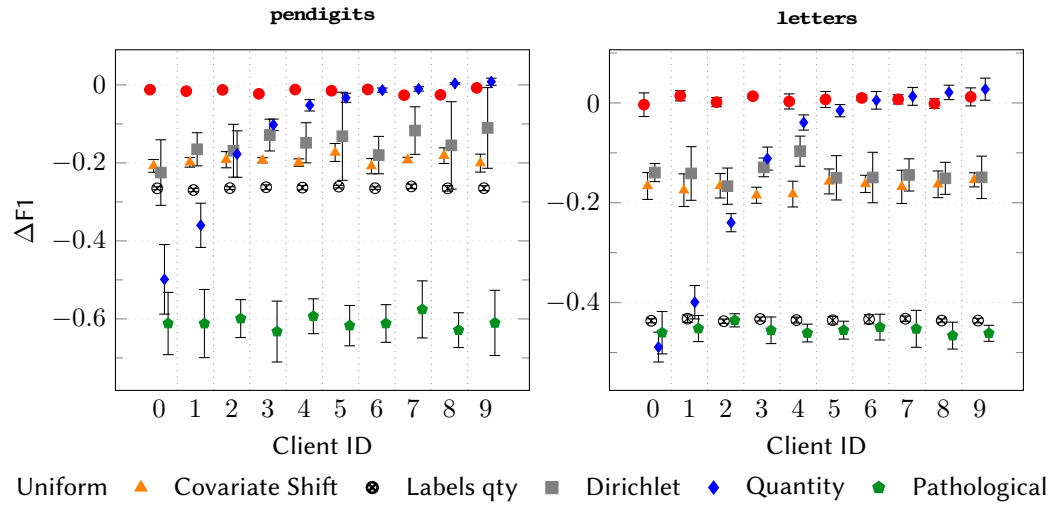

Figure 1: Difference in F1 score (ΔF1) between the local run of SAMME and the best results with the federation on the *pendigits* dataset. Results are reported for each clients and non-iidness. The scores are the average ($\pm$ standard deviation) over 5 runs.

We evaluated the methods on the following 10 datasets from the UCI [30] repository: *adult*, *kr-vs-kp*, *forestcover*, *splice*, *vehicle*, *segmentation*, *sat*, *pendigits*, *vowel*, *letter*. The datasets have been distributed across the clients using six different data distributions. Besides the iid case (uniform data distribution), we also consider the following types of non-iidness: quantity skew, prior shift (pathological, Dirichlet, and labels quantity), and covariate shift [26]. For more details about the implementation of these data distribution please refer to [3]. It is worth noticing that each client has at least two examples of different classes in every type of skewness.

The methods have been compared using standard classification metrics like accuracy, precision, recall, and F1. For space reasons, we only report the F1 score, which is the harmonic mean of the precision and recall, and it considers how the data is distributed.

Each experiment has been repeated five times. The reported results are the averages (with their standard deviation) over these runs. The python implementation of the methods and their evaluation is available at https://github.com/ml-unito/federation_boosting.

4.1. Results

We start by investigating how beneficial are the federations built by the proposed algorithm. To do that, we need to evaluate the performance of a possible competitor built only on local data. Then, for each non-iidness type, we ran the SAMME algorithm on each client, using only the local data for training and recorded the F1 score over a fixed independent test set.

Figure 1 shows, for all the clients and all the data distributions, the difference in F1 score (ΔF1) between the local run of SAMME (local SAMME in the following) and the best F1 score achieved by one of the federated algorithms on the *pendigits* and the *letters* datasets. The lower (more negative) ΔF1 is for a given point, the more beneficial is the federation for the corresponding client and data distribution setting.

	Dataset	Model	Uniform	Quantity	Cov. Shift	Dirichlet	Pathological	Labels qty
binary datasets	*adult*	Samme	86.45 ± 0.13	86.45 ± 0.13	86.45 ± 0.13			
		DistBoost.F	84.71 ± 0.15	84.66 ± 0.39	84.64 ± 0.24			
		PreWeak.F	84.97 ± 0.16	85.24 ± 0.07	**85.28 ± 0.04**			
		AdaBoost.F	**85.58 ± 0.06**	**86.01 ± 0.14**	85.12 ± 0.10			
	forestcover	Samme	81.64 ± 0.34	81.26 ± 0.40	81.70 ± 0.28			
		DistBoost.F	83.27 ± 0.20	82.10 ± 0.20	76.90 ± 1.48			
		PreWeak.F	**84.46 ± 0.18**	**84.09 ± 0.26**	**83.70 ± 0.13**			
		AdaBoost.F	83.67 ± 0.21	83.47 ± 0.14	79.42 ± 0.60			
	kr-vs-kp	Samme	99.75 ± 0.21	99.75 ± 0.21	99.75 ± 0.21			
		DistBoost.F	98.84 ± 0.36	99.09 ± 0.49	92.94 ± 2.58			
		PreWeak.F	97.78 ± 0.86	96.50 ± 0.79	95.16 ± 1.31			
		AdaBoost.F	**99.38 ± 0.29**	**99.41 ± 0.17**	**95.94 ± 0.56**			
multi-class datasets	*splice*	Samme	95.74 ± 0.63	95.74 ± 0.63	95.74 ± 0.63	95.74 ± 0.63	95.74 ± 0.63	95.74 ± 0.63
		DistBoost.F	94.67 ± 0.79	93.48 ± 1.43	94.26 ± 0.45	92.92 ± 0.99	91.41 ± 5.93	93.64 ± 0.76
		PreWeak.F	94.89 ± 1.03	94.64 ± 0.72	94.67 ± 0.73	**94.86 ± 0.67**	94.15 ± 1.02	**96.08 ± 1.01**
		AdaBoost.F	**95.61 ± 0.62**	**95.67 ± 0.81**	**94.83 ± 1.01**	94.51 ± 0.95	**94.17 ± 0.98**	94.70 ± 0.55
	vehicle	Samme	74.47 ± 1.48	74.47 ± 1.48	74.47 ± 1.48	74.47 ± 1.48	74.47 ± 1.48	74.47 ± 1.48
		DistBoost.F	68.82 ± 2.53	68.94 ± 4.17	66.35 ± 5.12	66.82 ± 2.22	67.65 ± 3.72	55.76 ± 8.03
		PreWeak.F	72.24 ± 3.66	**72.24 ± 3.98**	70.00 ± 4.83	**71.88 ± 5.37**	**70.47 ± 4.29**	**68.12 ± 3.26**
		AdaBoost.F	**72.94 ± 3.40**	69.88 ± 3.82	**70.82 ± 4.02**	69.76 ± 3.54	68.47 ± 1.29	65.29 ± 5.38
	segmentation	Samme	95.09 ± 0.20	95.09 ± 0.20	95.09 ± 0.20	95.09 ± 0.20	95.09 ± 0.20	95.09 ± 0.20
		DistBoost.F	85.91 ± 1.62	85.58 ± 2.31	82.46 ± 3.20	81.20 ± 3.84	**81.80 ± 2.25**	**54.84 ± 7.60**
		PreWeak.F	**87.55 ± 1.28**	**87.60 ± 1.74**	**87.42 ± 1.55**	**86.70 ± 3.69**	81.34 ± 5.54	38.70 ± 13.44
		AdaBoost.F	86.07 ± 2.86	87.34 ± 2.87	85.38 ± 1.48	83.01 ± 3.69	70.36 ± 8.81	49.95 ± 10.88
	sat	Samme	85.14 ± 0.34	85.14 ± 0.34	85.14 ± 0.34	85.14 ± 0.34	85.14 ± 0.34	85.14 ± 0.34
		DistBoost.F	81.78 ± 1.61	81.18 ± 0.71	82.08 ± 1.76	81.39 ± 1.53	78.54 ± 5.79	48.82 ± 6.14
		PreWeak.F	**86.41 ± 0.69**	**85.26 ± 1.57**	**85.87 ± 0.31**	**85.20 ± 0.15**	**82.65 ± 4.97**	**66.61 ± 8.97**
		AdaBoost.F	83.52 ± 0.58	83.79 ± 1.32	82.58 ± 0.50	81.56 ± 0.90	77.01 ± 5.17	55.18 ± 7.62
	pendigits	Samme	94.56 ± 0.41	94.56 ± 0.41	94.56 ± 0.41	94.56 ± 0.41	94.56 ± 0.41	94.56 ± 0.41
		DistBoost.F	88.63 ± 0.78	87.80 ± 1.53	91.53 ± 1.08	89.81 ± 1.83	87.01 ± 1.21	36.70 ± 6.89
		PreWeak.F	**93.83 ± 0.80**	93.48 ± 0.87	91.49 ± 1.17	92.62 ± 1.50	**94.42 ± 0.97**	46.16 ± 15.30
		AdaBoost.F	93.21 ± 0.80	**93.94 ± 0.54**	**93.88 ± 0.26**	**92.93 ± 0.99**	89.32 ± 2.00	**46.21 ± 3.87**
	vowel	Samme	86.16 ± 2.19	86.16 ± 2.19	86.16 ± 2.19	86.16 ± 2.19	86.16 ± 2.19	86.16 ± 2.19
		DistBoost.F	74.14 ± 4.25	70.51 ± 1.70	70.71 ± 2.79	68.89 ± 4.75	56.46 ± 8.12	27.27 ± 9.77
		PreWeak.F	77.88 ± 1.49	**80.71 ± 1.49**	**78.18 ± 2.12**	**77.78 ± 2.05**	**71.62 ± 5.81**	26.06 ± 10.75
		AdaBoost.F	**79.80 ± 1.47**	80.30 ± 2.72	77.27 ± 2.88	75.15 ± 4.78	66.67 ± 6.37	**31.21 ± 6.41**
	letter	Samme	75.33 ± 0.52	75.33 ± 0.52	75.33 ± 0.52	75.33 ± 0.52	75.33 ± 0.52	75.33 ± 0.52
		DistBoost.F	62.37 ± 2.61	61.26 ± 1.79	55.75 ± 0.98	61.11 ± 1.59	46.57 ± 8.63	21.12 ± 1.78
		PreWeak.F	**71.46 ± 1.85**	**71.93 ± 1.61**	**70.21 ± 1.50**	**69.90 ± 0.45**	64.34 ± 3.77	45.71 ± 3.92
		AdaBoost.F	68.32 ± 1.63	69.88 ± 0.78	66.58 ± 1.37	68.20 ± 0.93	**64.98 ± 4.04**	**54.96 ± 3.12**
	Avg. rank	Samme	-	-	-	-	-	-
		PreWeak.F	1.6	**1.5**	**1.5**	**1.143**	**1.333**	**1.5**
		DistBoost.F	2.9	2.9	2.9	3.0	2.429	2.9
		AdaBoost.F	**1.5**	1.6	1.6	1.875	2.000	1.6

Table 1
Summary of F1 scores at $T = 300$. For each dataset/non-iidness type combination the federated method with the best F1 score is highlighted in bold. All F1 figures are multiplied by 100 for the sake of readability.

Barring small differences in the actual numbers, the two experiments narrate the same story. The first thing to notice is that participating in the federation is generally beneficial to all clients, especially in non-iid data distributions.

An interesting observation is that, in the quantity skew scenario, clients with many examples (the head of the power-law) can reach F1 scores that are even higher than the federation. This

is reasonable because those clients are close to having all the available data; i.e., they run in a setting similar to running SAMME over the fused dataset, that is generally better than having to deal with the split dataset scenario. We can also observe that the scenarios with a prior shift (i.e., Labels Quantity, Dirichlet, and Pathological) are the most challenging ones. This is particularly apparent for the label quantity skew and the pathological label skew where, by design, we assign only a small subset of labels per client. We note that, contrary to what the figure might suggest, *in absolute terms* the performances of local SAMME on the label quantity skew case are worse than those in the pathological skew: the corresponding points ($\otimes$ symbols) appear upper (w.r.t. ⬟) because the federation does not perform well in this particular case. This is particularly apparent for the *pendigits* dataset where the label quantity skew is not as detrimental to the performances as in the *letters* dataset.

In the uniform data distribution case, the federation is only slightly useful (*pendigits*) and slightly detrimental (*letters*).

Table 1 provides all the average F1 scores ($\pm$ standard deviation) for all methods, datasets, and skewness. Overall, the performance of PreWeak.F and AdaBoost.F are significantly better than DistBoost.F. We can observe that, in general, the federation tends to achieve F1 scores very close to the centralized SAMME on datasets with few labels (e.g., 2 and 3), even in non-iid settings. Clearly, as the number of classes increases, the prior shift scenario becomes more and more challenging. The Labels Quantity skew is the most demanding setting because each client only has two labels. Thus, their weak classifiers are not good enough to be boosted effectively.

Overall, we believe that the evidence presented here is enough to conclude that the approach is beneficial and that DistBoost.F is not performing as well as the other two algorithms. There is evidence, albeit not conclusive, that PreWeak.F outperforms AdaBoost.F in terms of performances and that PreWeak.F might suffer more than AdaBoost.F from overfitting problems.

5. Conclusions

The possibility of applying federated learning beyond gradient-based methods may broaden the adaptation of this methodology. In this paper, we exploit ideas from distributed boosting literature to propose three algorithms DistBoost.F, PreWeak.F, and AdaBoost.F, which allow, for the first time ever, the federation of parties without putting constraints on the type of models learned in the clients. Indeed, to the best of our knowledge, our proposal is also the first to allow each client to choose a different local model.

Our experiments show that the federation works. The generalization error of the federation is driven down by the three algorithms and, except in trivial cases, the federated model largely outperforms the models that could have been learned locally. Experiments also show that non-iid data distributions can harm the quality of the federated model. Specifically, when an extreme skew on the labels is present, the federation might suffer, especially when the problem is multi-class and the number of possible labels is large. We leave as future work a comparison between our approach and traditional (gradient-based) federated algorithms. The comparison would also allow us to assess how much the problems we observed in some non-iid settings are specific to our methodology.

This work opens the doors to many possible future directions. We aim to perform an in-depth

analysis of these algorithms' security and privacy aspects in our future work. As mentioned, we would like to compare their behavior against gradient-based alternatives.

References

[1] McMahan et al., Communication-efficient learning of deep networks from decentralized data, in: Artificial intelligence and statistics, PMLR, 2017, pp. 1273–1282.

[2] F. D'Ascenzo, O. De Filippo, G. Gallone, G. Mittone, M. A. Deriu, M. Iannaccone, A. Ariza-Solé, C. Liebetrau, S. Manzano-Fernández, G. Quadri, et al., Machine learning-based prediction of adverse events following an acute coronary syndrome (praise): a modelling study of pooled datasets, The Lancet 397 (2021) 199–207.

[3] M. Polato, M. Aldinucci, Boosting the federation: Cross-silo federated learning without gradient descent, 2022 International Joint Conference on Neural Networks (IJCNN) (2022) 1–10.

[4] Y. Freund, R. E. Schapire, Game theory, on-line prediction and boosting, in: Proceedings of the ninth annual conference on Computational learning theory, 1996, pp. 325–332.

[5] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of computer and system sciences 55 (1997) 119–139.

[6] Y. Freund, R. Schapire, N. Abe, A short introduction to boosting, Journal-Japanese Society For Artificial Intelligence 14 (1999) 1612.

[7] A. Lazarevic, Z. Obradovic, Boosting algorithms for parallel and distributed learning, Distributed and Parallel Databases 11 (2002) 203–229. URL: https://doi.org/10.1023/A:1013992203485. doi:10.1023/A:1013992203485.

[8] J. Cooper, L. Reyzin, Improved algorithms for distributed boosting, in: 2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2017, pp. 806–813. doi:10.1109/ALLERTON.2017.8262822.

[9] L. Breiman, Bagging predictors, Machine learning 24 (1996) 123–140.

[10] D. H. Wolpert, Stacked generalization, Neural networks 5 (1992) 241–259.

[11] E. B. Kong, T. G. Dietterich, Error-correcting output coding corrects bias and variance, in: Machine learning proceedings 1995, Elsevier, 1995, pp. 313–321.

[12] R. E. Schapire, The strength of weak learnability, Machine learning 5 (1990) 197–227.

[13] Y. Liu, Y. Liu, Z. Liu, J. Zhang, C. Meng, Y. Zheng, Federated forest (2019). doi:10.1109/TBDATA.2020.2992755. arXiv:arXiv:1905.10053.

[14] L. A. C. de Souza, G. Antonio F. Rebello, G. F. Camilo, L. C. B. Guimarães, O. C. M. B. Duarte, Dfedforest: Decentralized federated forest, in: 2020 IEEE International Conference on Blockchain (Blockchain), 2020, pp. 90–97. doi:10.1109/Blockchain50366.2020.00019.

[15] Y. Wu, S. Cai, X. Xiao, G. Chen, B. C. Ooi, Privacy preserving vertical federated learning for tree-based models, Proc. VLDB Endow. 13 (2020) 2090–2103. URL: https://doi.org/10.14778/3407790.3407811. doi:10.14778/3407790.3407811.

[16] R. Cramer, I. B. Damgård, J. B. Nielsen, Secure Multiparty Computation and Secret Sharing, Cambridge University Press, 2015. doi:10.1017/CBO9781107337756.

[17] M. Polato, A. Gallinaro, F. Aiolli, Privacy-preserving kernel computation for ver-

tically partitioned data, in: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), 2021. doi:https://doi.org/10.14428/esann/2021.ES2021-152.

[18] S.-T. Chen, M.-F. Balcan, D. H. Chau, Communication efficient distributed agnostic boosting, in: A. Gretton, C. C. Robert (Eds.), Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, volume 51 of *Proceedings of Machine Learning Research*, PMLR, Cadiz, Spain, 2016, pp. 1299–1307. URL: https://proceedings.mlr.press/v51/chen16e.html.

[19] F. Wang, J. Ou, H. Lv, Gradient boosting forest: a two-stage ensemble method enabling federated learning of gbdts., in: T. Mantoro, M. Lee, M. Ayu, K. Wong, A. Hidayanto (Eds.), Internation Conference On Neural Information Processing (ICONIP), volume Lecture Notes in Computer Science, vol 13109, Springer, Cham., 2021. doi:https://doi.org/10.1007/978-3-030-92270-2_7.

[20] Y. Liu, Z. Ma, X. Liu, S. Ma, S. Nepal, R. H. Deng, K. Ren, Boosting privately: Federated extreme gradient boosting for mobile crowdsensing, in: 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), 2020, pp. 1–11. doi:10.1109/ICDCS47774.2020.00017.

[21] Y. Liu, Z. Fan, X. Song, R. Shibasaki, Fedvoting: A cross-silo boosting tree construction method for privacy-preserving long-term human mobility prediction, Sensors 21 (2021). URL: https://www.mdpi.com/1424-8220/21/24/8282. doi:10.3390/s21248282.

[22] K. Cheng, T. Fan, Y. Jin, Y. Liu, T. Chen, D. Papadopoulos, Q. Yang, Secureboost: A lossless federated learning framework, IEEE Intelligent Systems 36 (2021) 87–98. doi:10.1109/MIS.2021.3082561.

[23] W. Fang, D. Zhao, J. Tan, C. Chen, C. Yu, L. Wang, L. Wang, J. Zhou, B. Zhang, Large-Scale Secure XGB for Vertical Federated Learning, Association for Computing Machinery, New York, NY, USA, 2021, pp. 443–452. URL: https://doi.org/10.1145/3459637.3482361.

[24] F. Fu, Y. Shao, L. Yu, J. Jiang, H. Xue, Y. Tao, B. Cui, VF2Boost: Very Fast Vertical Federated Gradient Boosting for Cross-Enterprise Learning, Association for Computing Machinery, New York, NY, USA, 2021, pp. 563–576. URL: https://doi.org/10.1145/3448016.3457241.

[25] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, H. Yu, Federated Learning, Synthesis Lectures on Artificial Intelligence and Machine Learning 13 (2019) 1–207. URL: https://www.morganclaypool.com/doi/10.2200/S00960ED2V01Y201910AIM043. doi:10.2200/S00960ED2V01Y201910AIM043.

[26] Kairouz et al., Advances and Open Problems in Federated Learning, Foundations and Trends in Machine Learning 14 (2021) 1–210. URL: http://www.nowpublishers.com/article/Details/MAL-083. doi:10.1561/2200000083.

[27] Y. Arfat, G. Mittone, I. Colonnelli, F. D'Ascenzo, R. Esposito, M. Aldinucci, et al., Pooling critical datasets with federated learning, in: Proc. of the 31st Euromicro Intl. Conference on Parallel Distributed and network-based Processing (PDP), IEEE, 2023, pp. 1–9.

[28] G. Mittone, W. Riviera, I. Colonnelli, R. Birke, M. Aldinucci, Model-agnostic federated learning, arXiv preprint arXiv:2303.04906 (2023).

[29] T. Hastie, S. Rosset, J. Zhu, H. Zou, Multi-class adaboost, Statistics and its Interface 2 (2009) 349–360.

[30] D. Dua, C. Graff, UCI machine learning repository, 2017. URL: http://archive.ics.uci.edu/ml.