Online Decentralized Frank-Wolfe: From theoretical bound to applications in smart-building

Angan Mitra, Kim Thang Nguyen, Tuan-Anh Nguyen, Denis Trystram, Paul Youssef

Online Decentralized Frank-Wolfe: From theoretical bound to applications in smart-building*

Angan Mitra[1,3], Nguyen Kim Thang[2], Tuan-Anh Nguyen[1], Denis Trystram[1], and Paul Youssef[1]

[1] Univ.Grenoble-Alpes, LIG, France
[2] IBISC, Univ.Evry, Université Paris-Saclay, France
[3] Qarnot Computing, France

Abstract. The design of decentralized learning algorithms is important in the fast-growing world in which data are distributed over participants with limited local computation resources and communication. In this direction, we propose an online algorithm minimizing non-convex loss functions aggregated from individual data/models distributed over a network. We provide the theoretical performance guarantee of our algorithm and demonstrate its utility on a real life smart building.

1 Introduction

The popularity of sensors and IoT devices has the potential of generating and equivalently accumulating data in order of Zeta bytes [15] annually. High throughput, low latency, data consumption, networking dependencies are often the key metrics in designing high-performance learning algorithms under the constraint of low powered computing. In recent times, there has been an alternate trend to process data on cloud or dump into a centralized database. Commonly known as edge computing, the new paradigm embraces the idea of using interconnected computing nodes to reduce high bandwidth consuming data uploads, privacy preservation of data and knowledge on the fly.

Smart building applications typically have a profound implication on environment in terms of energy savings, reduction of green house emission, etc. Predicting the future often forms the basis of corrective actions taken by such apps and can be regarded as a predominant use-case of machine learning. Usually the data is generated across multiple zones from heterogeneous sensors and forms a setting of decentralized learning. In recent times, the hardware-software interface has benefited from advances in network communication coupled with edge computing. Thus deploying a machine learning model in site and processing data on the fly has become a realistic alternative of sending data to a centralised data base. Optimizing problems to maintain robust solutions under the uncertainty

* Supported by the Multidisciplinary Institute in Artificial Intelligence, Univ.Grenoble Alpes, France (ANR-19-P3IA-0003)

of future is a nice to have feature for such cyber physical systems. Contrary to the classical train-test-deploy framework, online learning offers continual learning where during run time, a batch of sensor data has the potential to update an AI model on site.

This work aligns with the edge computing paradigm by proposing an online and decentralized learning algorithm. Online learning helps better adapt to the uncertainty of the future where the data pattern continually changes over time. The designed algorithm repeatedly chooses a high-performance strategy given a set of actions compared to the best-fixed action in hindsight. Instead of having a centralized mediator, the decentralized setting promotes peer-to-peer knowledge exchanges while prohibiting data sharing between learners. Many proposed online decentralized algorithms use gradient descent-based methods to solve constraint problems. Such an approach requires projection into the constraint set that usually involves intensive computation, which is not best suited in the context of sensors and IoT. We aim to design a competitive, robust algorithm in the decentralized and online setting that has the flexibility of being projection-free.

Problem setting Formally, we are given a convex set $\mathcal{K} \subseteq \mathbb{R}^d$ and a set of agents connected over a network represented by a graph $G = (V, E)$ where $n = |V|$ is the number of agents. At every time $1 \leq t \leq T$, each agent $i \in V$ can communicate with (and only with) its immediate neighbors, i.e., adjacent agents in G and takes a decision $\boldsymbol{x}_i^t \in \mathcal{K}$. Subsequently, a batch of new data is revealed exclusively to agent i and from its own batch, a non-convex cost function $f_i^t : \mathcal{K} \to \mathbb{R}$ is induced locally. Although each agent i observes only function f_i^t, agent i is interested in the cumulating cost $F^t(\cdot)$ where $F^t(\cdot) := \frac{1}{n}\sum_{j=1}^n f_j^t(\cdot)$. In particular, at time t, the cost of agent i with the its chosen $\boldsymbol{x}_i^t$ is $F^t(\boldsymbol{x}_i^t)$. The objective of each agent i is to minimize the total cumulating cost $\sum_{t=1}^T F^t(\boldsymbol{x}_i^t)$ via local communication with its immediate neighbors.

When the cost functions f_i^t are convex, a standard measure is the *regret* notion. An online algorithm is $R(T)$-*regret* if for every agent $1 \leq i \leq n$,

$$\frac{1}{T}\left(\sum_{t=1}^{T} F^t(\boldsymbol{x}_i^t) - \min_{\boldsymbol{o}\in\mathcal{K}} \sum_{t=1}^{T} F^t(\boldsymbol{o})\right) \leq R(T)$$

As the cost functions in the paper are not necessarily convex, we consider a stationary measure on the quality of solution based on the Frank-Wolfe gap [11], and that can be considered the counter-part of the regret in the non-convex setting. Specifically, we aim to bound the *convergence gap*, for every agent $1 \leq i \leq n$:

$$\max_{\boldsymbol{o}\in\mathcal{K}} \frac{1}{T}\sum_{t=1}^{T} \langle \nabla F^t(\boldsymbol{x}_i^t), \boldsymbol{x}_i^t - \boldsymbol{o}\rangle \tag{1}$$

In the same spirit as the regret, the measure of convergence gap compares the total cost of every agent to that of the best stationary point in hindsight. Note that when the functions F^t are convex, the convergence gap is always upper

bounded by the regret. Moreover, when the problem becomes offline, i.e., all F^t are the same, the convergence gap measures the speed of convergence to a stationary solution.

1.1 Our contribution

The challenge in designing robust and efficient algorithms for the problem is to resolve the following issues together: the uncertainty (online setting, agents observe their own loss functions only after choosing their decisions), the partial information (decentralized setting, agents know only its own loss functions while aiming to minimize the cumulating cost), and the non-convexity of the loss functions. As a starting point, we consider the Meta Frank-Wolfe (MFW) algorithm [3] in the (centralized, convex) online setting and the Decentralized Frank-Wolfe (DFW) algorithm [22] in the decentralized (offline) setting. However, these algorithms work either in the online setting or in the decentralized one but not both together. The difficulty in our problem, as mentioned earlier, is to resolve all issues together.

In the paper, we present algorithms, subtly built on MFW and DFW algorithms, that achieves the convergence gap of $O(T^{-1/2})$ and $O(T^{-1/4})$ in cases where the exact gradients or only stochastic gradients of loss functions are available, respectively. Note that in the former, the convergence gap of $O(T^{-1/2})$ asymptotically matches the best regret guarantee even in the centralized offline settings with convex functions. Besides, one can convert the algorithms to be projection-free by choosing appropriate oracles used in the algorithm. This property provides a flexibility to apply the algorithms to different settings depending on the computing capacity of local devices. Our work applies to online neural network optimization amongst a group of autonomous learners. We demonstrate the practical utility of our algorithm in a smart building application where zones mimic learners optimizing a temperature forecasting problem. We provide a thorough analysis of our algorithms in different angles of the performance guarantee (quality of solutions), the effects of network topology and decentralization, which are predictably explained by our theoretical results.

1.2 Related Work

Decentralized Online Optimization. Authors [24] introduced decentralized online projected subgradient descent and showed vanishing regret for convex and strongly convex functions. In contrast, Hosseini et al. [10] extended distributed dual averaging technique to the online setting using a general regularized projection for both unconstrained and constrained optimization. A distributed variant of online conditional gradient [8] was designed and analyzed in [26] that requires linear minimizers and uses exact gradients. However, computing exact gradients may be prohibitively expensive for moderately sized data and intractable when a closed-form does not exist. In this work, we go a step ahead in designing a distributed algorithm that uses stochastic gradient estimates and provides a better regret bound than in [26].

Learning on the edge. Over the year, edge computing has become an exciting alternative for cloud-based learning by processing the data closer to end devices while ensuring data confidentiality and reducing transmission. [23] proposes a distributed framework for non-i.i.d data using multiple gradient descent-based algorithms to update local models and a dedicated edge unit for global aggregation. Another popular approach is to reduce the memory size of classical machine learning models to meet edge resource constraints. [20] and [18] similarly takes this idea by building a tree-based learning framework with a considerable reduction in memory using compression and pruning. At the same time, [6] introduce an edge-friendly version of k-nearest neighbor [5] by projecting the data into a lower-dimensional space. Besides traditional machine learning algorithms, adapting deep learning models to work on edge devices is an emerging research domain. In [4, 14], the authors propose a pruning technique on convolutional network for faster computation while preserving the model ability. Another approach using weight quantization is proposed in [21]. The current dominant paradigm is federated learning [16, 12], where offline centralized training is performed through a star network with multiple devices connected to a central server. However, decentralized training is more efficient than centralized one when operating on networks with low bandwidth or high latency [13, 9]. In this paper, we go one step further by studying arbitrary communication networks without a central coordinator and the local data (so local cost functions) evolve.

Thermal Profiling a Building. Usually, building monitoring sensors are distributed across a building and thus acts as a scattered data lake with potentially heterogeneous patterns. Indoor temperature is an important factor in controlling Heating Ventilation Air Conditioning systems that maintain ambient comfort within a building [7]. Typically such embedded systems run in anticipatory mode where temperature prediction [2] of controlled building zones helps in maintaining thermal consistency. A multitude of factors effect the thermal profile like outdoor environment, opening/closing of windows, number of occupants, etc, which are hard to get and often rely on intrusive mechanisms to gather the data. Researchers have utilized deep learning models [25] in the context of online learning of temperature, but lack the benefit of interacting with multiple similar sensors. This study seeks to generate a thermal profile of a building by only utilizing temperature data from multiple zones of a building in order to extract patterns about thermal variation. The proposed methodology not only processes data on the fly [1], but also identifies meaningful topological data exchange networks that can best predict multi zonal temperature settings.

2 Conditional Gradient based Algorithm

In this section, after introducing and recalling useful notions, we will first provide an algorithm for the setting with exact gradients. Subsequently, building on the salient ideas of that algorithm, we extend to the more realistic setting with stochastic gradients.

2.1 Preliminaries and Notations

Given an undirected graph $G = (V, E)$, the set of neighbors of an agent $i \in V$ is $N(i) := \{j \in V : (i, j) \in E\}$. Consider a symmetric matrix $W \in \mathbb{R}_+^{n \times n}$ defined as follows. The entry W_{ij} has a value of

$$W_{ij} = \begin{cases} \dfrac{1}{1 + \max\{d_i, d_j\}} & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E, i \neq j \\ 1 - \sum_{j \in N(i)} W_{ij} & \text{if } i = j \end{cases}$$

where $d_i = |N(i)|$, the degree of vertex i. In fact, the matrix W is doubly stochastic, i.e $W\mathbf{1} = W^T\mathbf{1} = \mathbf{1}$ and so it inherits several useful properties of doubly stochastic matrices. We use boldface letter e.g $\boldsymbol{x}$ to represent vectors. We denote $\boldsymbol{x}_i^t$ as the decision vector of agent i at time step t. We suppose that the constraint set $\mathcal{K}$ is a bounded convex set with diameters $D = \sup_{\boldsymbol{x},\boldsymbol{y} \in \mathcal{K}} \|\boldsymbol{x} - \boldsymbol{y}\|$.

A function f is *β-smooth* if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{K}$:

$$f(\boldsymbol{y}) \leq f(\boldsymbol{x}) + \langle \nabla f(\boldsymbol{x}), \boldsymbol{y} - \boldsymbol{x} \rangle + \frac{\beta}{2} \|\boldsymbol{y} - \boldsymbol{x}\|^2$$

or equivalently $\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq \beta \|\boldsymbol{x} - \boldsymbol{y}\|$. Also, we say a function f is *G-Lipschitz* if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{K}$

$$\|f(\boldsymbol{x}) - f(\boldsymbol{y})\| \leq G \|\boldsymbol{x} - \boldsymbol{y}\|$$

In our algorithm, we make use of linear optimization oracles where its role is to resolve an online linear optimization problem given a feedback function and a constraint set. Specifically, in the online linear optimization problem, at every time $1 \leq t \leq T$, one has to select $\boldsymbol{u}^t \in \mathcal{K}$. Subsequently, the adversary reveals a vector $\boldsymbol{d}^t$ and feedbacks the cost function $\langle \cdot, \boldsymbol{d}^t \rangle$. The objective is to minimize the regret, i.e., $\frac{1}{T}\big(\sum_{t=1}^T \langle \boldsymbol{u}^t, \boldsymbol{d}^t \rangle - \min_{\boldsymbol{u}^* \in \mathcal{K}} \sum_{t=1}^T \langle \boldsymbol{u}^*, \boldsymbol{d}^t \rangle\big)$. Several algorithms [8] provide an optimal regret bound of $\mathcal{R}^T = O(1/\sqrt{T})$ for the online linear optimization problem. These algorithms include the online gradient descent algorithm or the follow-the-perturbed-leader algorithm (projection-free). One can pick one of such algorithms to be an oracle resolving the online linear optimization problem.

2.2 An Algorithm with Exact Gradients

Assume that the exact gradients of the loss functions f_i^t are available (or can be computed). The high-level idea of the algorithm is the following. In the algorithm, at every time t, each agent i executes L steps of the Frank-Wolfe algorithm where every update vector (for iterations $1 \leq \ell \leq L$ where the parameter L will be chosen later) is constructed by combining the outputs of linear optimization oracles $\mathcal{O}_{j,\ell}$ and the current vectors of its neighbors $j \in N(i)$. During this

execution, a set of feasible solutions $\{\boldsymbol{x}_{i,\ell}^t : 1 \leq \ell \leq L\}$ is computed. The solution $\boldsymbol{x}_i^t$ for each agent $1 \leq i \leq n$ is then chosen uniformly at random among $\{\boldsymbol{x}_{i,\ell}^t : 1 \leq \ell \leq L\}$. Subsequently, after communicating and aggregating the information related to functions f_j^t for $j \in N(i)$, the algorithm computes a vector $\boldsymbol{d}_{i,\ell}^t$ and feedbacks $\langle \boldsymbol{d}_{i,\ell}^t, \cdot \rangle$ as the cost function at time t to the oracle $\mathcal{O}_{i,\ell}$ for $1 \leq \ell \leq L$. The vectors $\boldsymbol{d}_{i,\ell}^t$'s are subtly built so that it captures step-by-step more and more information on the cumulating cost functions. The formal description is given in Algorithm 1 and a detailed proof of Theorem 1 is given in [17]

Algorithm 1 Online Decentralized algorithm

Input: A convex set $\mathcal{K}$, a time horizon T, a parameter L, online linear optimization oracles $\mathcal{O}_{i,1}, \ldots, \mathcal{O}_{i,L}$ for each agent $1 \leq i \leq n$, step sizes $\eta_\ell \in (0,1)$ for all $1 \leq \ell \leq L$

for $t = 1$ to T **do**
  for every agent $1 \leq i \leq n$ **do**
    Initialize arbitrarily $\boldsymbol{x}_{i,1}^t \in \mathcal{K}$
    for $1 \leq \ell \leq L$ **do**
      Let $\boldsymbol{v}_{i,\ell}^t$ be the output of oracle $\mathcal{O}_{i,\ell}$ at time step t.
      Send $\boldsymbol{x}_{i,\ell}^t$ to all neighbours $N(i)$
      Once receiving $\boldsymbol{x}_{j,\ell}^t$ from all neighbours $j \in N(i)$, set $\boldsymbol{y}_{i,\ell}^t \leftarrow \sum_j W_{ij} \boldsymbol{x}_{j,\ell}^t$.
      Compute $\boldsymbol{x}_{i,\ell+1}^t \leftarrow (1-\eta_\ell)\boldsymbol{y}_{i,\ell}^t + \eta_\ell \boldsymbol{v}_{i,\ell}^t$.
    end for
    Choose $\boldsymbol{x}_i^t \leftarrow \boldsymbol{x}_{i,\ell}^t$ for $1 \leq \ell \leq L$ with probability $\frac{1}{L}$ and play $\boldsymbol{x}_i^t$
    Receive function f_i^t
    Set $\boldsymbol{g}_{i,1}^t \leftarrow \nabla f_i^t(\boldsymbol{x}_{i,1}^t)$
    for $1 \leq \ell \leq L$ **do**
      Send $\boldsymbol{g}_{i,\ell}^t$ to all neighbours $N(i)$.
      After receiving $\boldsymbol{g}_{j,\ell}^t$ from all neighbours $j \in N(i)$, compute $\boldsymbol{d}_{i,\ell}^t \leftarrow \sum_{j \in N(i)} W_{ij} \boldsymbol{g}_{j,\ell}^t$ and $\boldsymbol{g}_{i,\ell+1}^t \leftarrow \big(\nabla f_i^t(\boldsymbol{x}_{i,\ell+1}^t) - \nabla f_i^t(\boldsymbol{x}_{i,\ell}^t)\big) + \boldsymbol{d}_{i,\ell}^t$.
      Feedback function $\langle \boldsymbol{d}_{i,\ell}^t, \cdot \rangle$ to oracles $\mathcal{O}_{i,\ell}$. (The cost of the oracle $\mathcal{O}_{i,\ell}$ at time t is $\langle \boldsymbol{d}_{i,\ell}^t, \boldsymbol{v}_{i,\ell}^t \rangle$.)
    end for
  end for
end for

Theorem 1. *Let $\mathcal{K}$ be a convex set with diameter D. Assume that functions F^t (possibly non convex) are β-smooth and G-Lipschitz for every $1 \leq t \leq T$. Then, by choosing the step size $\eta_\ell = \min\left(1, \frac{A}{\ell^\alpha}\right)$ for some $A \geq 0$ and $\alpha \in (0,1)$, Algorithm 1 guarantees that for all $1 \leq i \leq n$:*

$$\max_{\boldsymbol{o} \in \mathcal{K}} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{x}_i^t} \left[\langle \nabla F^t(\boldsymbol{x}_i^t), \boldsymbol{x}_i^t - \boldsymbol{o} \rangle \right] \leq O\left(\frac{GDA^{-1}}{L^{1-\alpha}} + \frac{AD^2\beta/2}{L^\alpha(1-\alpha)} + \mathcal{R}^T \right)$$

where $\mathcal{R}^T$ is the regret of online linear minimization oracles. Choosing $L = T$, $\alpha = 1/2$ and oracles as gradient descent or follow-the-perturbed-leader with regret $\mathcal{R}^T = O\big(T^{-1/2}\big)$, we obtain the gap convergence rate of $O\big(T^{-1/2}\big)$.

2.3 Algorithm with Stochastic Gradients

We extend the previous algorithm to the setting of stochastic gradients estimates. As only stochastic gradient estimates are available, we use a variance reduction technique in order to upgrade Algorithm 1 to its stochastic version (Algorithm 2). The difference between the two algorithms is stochastic gradient estimation and an additional step for variance reduction. After making the decision, the agent receives an unbiased gradient to perform updates and communication to obtain stochastic estimates $\widetilde{\boldsymbol{g}}_{i,\ell}^t$ and $\widetilde{\boldsymbol{d}}_{i,\ell}^t$ of $\boldsymbol{g}_{i,\ell}^t$ and $\boldsymbol{d}_{i,\ell}^t$, respectively. (Note that the stochastic variables are denoted by the same letter as its exact counterpart with an additional tilde symbol.) Then the agent uses Step 17 in Algorithm 2 to get the reduced variance version $\widetilde{\boldsymbol{a}}_{i,\ell}^t$ of $\widetilde{\boldsymbol{d}}_{i,\ell}^t$. The function $\langle \widetilde{\boldsymbol{a}}_{i,\ell}^t, \cdot \rangle$ is then feedbacked to the oracle.

The formal description is given in Algorithm 2 in which all previous steps are the same as Algorithm 1 and the additional variance reduction step is marked in red. A detailed proof of Theorem 2 can be found in [17].

Algorithm 2 Stochastic online decentralized algorithm

...

12: Receive function f_i^t and an unbiased gradient estimate $\widetilde{\nabla} f_i^t$
13: Set $\widetilde{\boldsymbol{g}}_{i,1}^t \leftarrow \widetilde{\nabla} f_i^t(\boldsymbol{x}_{i,1}^t)$
14: **for** $1 \le \ell \le L$ **do**
15: Send $\widetilde{\boldsymbol{g}}_{i,\ell}^t$ to all neighbours $N(i)$.
16: After receiving $\widetilde{\boldsymbol{g}}_{j,\ell}^t$ from $j \in N(i)$, compute $\widetilde{\boldsymbol{d}}_{i,\ell}^t \leftarrow \sum_{j \in N(i)} W_{ij} \widetilde{\boldsymbol{g}}_{j,\ell}^t$ and set $\widetilde{\boldsymbol{g}}_{i,\ell+1}^t \leftarrow \big(\widetilde{\nabla} f_i^t(\boldsymbol{x}_{i,\ell+1}^t) - \widetilde{\nabla} f_i^t(\boldsymbol{x}_{i,\ell}^t)\big) + \widetilde{\boldsymbol{d}}_{i,\ell}^t$.
17: $\widetilde{\boldsymbol{a}}_{i,\ell}^t \leftarrow (1-\rho_\ell) \cdot \widetilde{\boldsymbol{a}}_{i,\ell-1}^t + \rho_\ell \cdot \widetilde{\boldsymbol{d}}_{i,\ell}^t$.
18: Feedback function $\langle \widetilde{\boldsymbol{a}}_{i,\ell}^t, \cdot \rangle$ to oracles $\mathcal{O}_{i,\ell}$. (The cost of the oracle $\mathcal{O}_{i,\ell}$ at time t is $\langle \widetilde{\boldsymbol{a}}_{i,\ell}^t, \boldsymbol{v}_{i,\ell}^t \rangle$.)
19: **end for**

Theorem 2. *Let $\mathcal{K}$ be a convex set with diameter D. Assume that for every $1 \le t \le T$.*

1. *functions f_i^t are β-smooth and G-Lipschitz,*
2. *the gradient estimates are unbiased with bounded variance σ^2,*
3. *the gradient estimates are Lipschitz.*

Then, choosing the step-sizes $\eta_\ell = \min\{1, \frac{A}{\ell^{3/4}}\}$ for some $A \ge 0$, we have for all $1 \le i \le n$,

$$\max_{\boldsymbol{o} \in \mathcal{K}} \mathbb{E}\Big[\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{x}_i^t}\big[\langle \nabla F_t\left(\boldsymbol{x}_i^t\right), \boldsymbol{x}_i^t - \boldsymbol{o}\rangle\big]\Big] \le O\left(\frac{DG + 2ADQ^{1/2}}{L^{1/4}} + \frac{2AD^2\beta}{L^{3/4}} + \mathcal{R}^T\right)$$

Choosing $L = T$ and oracles with regret $\mathcal{R}^T = O\left(T^{-1/2}\right)$, we obtain the convergence gap of $O\left(T^{-1/4}\right)$.

3 Experiments

The data-set used for experimentation comes from a 7 storey building with 24 sensor equipped zones [19].The zone-wise knowledge exchange happens through the edges of an undirected graph of n nodes participating in the learning process. For every round t, each node i receives a batch $\mathcal{B}_i^t$ of 32 time-series sequences corresponding to a look-back period 13 timestep to predict the temperature of the next timestep. We extract the data from March 7^{th} to April 20^{th} for training, set L equal to 360, $\alpha = 0.95$ and $A = 1$. A min-max scaler is used to normalize the data and we apply a rolling window with stride 1 on the original time series. Each node is embedded with a model built from a two-layers long-short-time-memory (LSTM) network followed by a fully connected layer. Denote the output of the model i for a data sequence b at time t by $\hat{y}_{i,b}^t$ and its ground truth by $y_{i,b}^t$. Consider the ℓ_1 loss as the objective function :

$$\mathcal{L}(\hat{y}_{i,b}^t, y_{i,b}^t) = \begin{cases} \dfrac{(\hat{y}_{i,b}^t - y_{i,b}^t)^2}{2} & \text{if } |\hat{y}_{i,b}^t - y_{i,b}^t| \leq 1 \\ |\hat{y}_{i,b}^t - y_{i,b}^t| - \frac{1}{2} & \text{otherwise.} \end{cases}$$

Consider the constraint set $\mathcal{K} = \{\boldsymbol{x} \in \mathbb{R}^d, \|\boldsymbol{x}\|_1 \leq r\}$, where $\boldsymbol{x}$ is the model's weight, d its dimension and $r = 1$. The (normalized) loss incurred by the data of agent i is $\frac{1}{|\mathcal{B}_i^t|} \sum_{b \in \mathcal{B}_i^t} \mathcal{L}(\hat{y}_{i,b}^t, y_{i,b}^t)$. The global loss function incurred by the overall data is

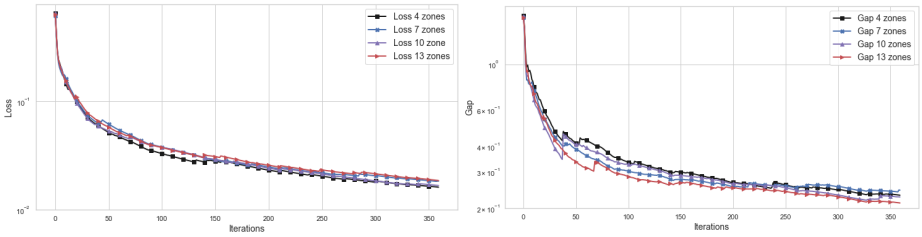
$$F^t(\boldsymbol{x}) = \frac{1}{|\cup_{i=1}^n \mathcal{B}_i^t|} \sum_{b \in \cup_{i=1}^n \mathcal{B}_i^t} \mathcal{L}(\hat{y}_{i,b}^t, y_{i,b}^t),$$

that can be written as $F^t(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^n f_i^t(\boldsymbol{x})$ where $f_i^t(\boldsymbol{x}) = \frac{1}{|\mathcal{B}_i^t|} \sum_{b \in \mathcal{B}_i^t} \mathcal{L}(\hat{y}_{i,b}^t, y_{i,b}^t)$. Note that the non-convexity here is due to the non-convexity of $\hat{y}_{i,b}^t$ as a function of $\boldsymbol{x}_i^t$. In the following section, if not specify otherwise, we call *loss* the temporal average of the global loss function F^t defined as $\frac{1}{T} \sum_{t=1}^T F^t$.

3.1 Prediction Performance

Figures 1a and 1b show the loss and gap values for different network sizes. The implementation justifies our theoretical results about the convergence of the gap. Besides, we also observe the convergence of loss value, an expected implication of the gap convergence. We set M the number of prediction points between the 21^{st} and 24^{th} of April and n the number of zones within one configuration. We use the mean absolute error (MAE $= \frac{1}{nM} \sum_{i=1}^n \sum_{m=1}^M |\hat{y}_{i,m} - y_{i,m}|$) and mean square error (MSE $= \frac{1}{nM} \sum_{i=1}^n \sum_{m=1}^M (\hat{y}_{i,m} - y_{i,m})^2$) as a measure between the

prediction and the ground truth. We observe that increasing nodes in a network does not always lead to better online performance. In-fact, a 7 node configuration achieves the lowest MSE (0.65) and MAE (0.78) for floors 6 and 7. We see a 40 % drop in MSE and 20 % reduction in MAE for floor 6 zonal models when 3 extra peers from floor 7 joined the group. We observe 19 % and 25 % increase in MSE and MAE values by adding zonal nodes from floor 7 to a 10 node group. This can be best argued by the fact that the top floor of a building has a non identical thermal variation with the rest of the storeys.



(a) Loss Value

(b) Gap Value

Fig. 1: Loss and Gap values of different network size on complete topology *(Plot on log-scale)*

3.2 Effect of Network Topology

We study the effect of topology in learning for a 7 node configuration with a complete, cycle and line graph containing 28, 7 and 6 edges respectively and with 13 nodes having 78,13 and 12 edges respectively. For both 7 (Table 1a) and 13 (Table 1b) node configurations, we observe that the complete graph yields the least amount of prediction error, mean absolute error $\in [0.66, 1.3]^{\circ}C$. However we note the peculiarity that the line graph can perform better than a cycle graph and has roughly a 10 % error margin compared to the complete configuration.

3.3 Effect of Decentralization

We are interested in understanding the role of decentralization in terms of accuracy of zonal learners. Let $L_{MFW}(t)$ be the loss from Meta Frank Wolfe (MFW) at time t. The approximation ratio $A(t) = \frac{L_{DMFW}(t)}{L_{MFW}(t)}$ at time t represents how worse is our decentralized version compared to a centralized optimization. $A(t) \leq B_{max}$ will mean our algorithm performs no worse than B_{max} times of the MFW. On figure 2, we plot the ratio $A(t)$ for a 13 node network and show that $A(t) \leq 1.4$. The 7 node network has the closest approximation bounded by 1.35 which can be explained by earlier insights on performance accuracy. We notice that the 10 node network performs worse till $t = 200$ and after $t \geq 250$ or

Topology	Metric	Mean	Var	Max	Min
cycle	MAE	1.09	0.48	1.80	0.56
cycle	MSE	0.78	0.21	1.09	0.52
complete	MAE	**0.77**	**0.38**	**1.47**	0.27
complete	MSE	**0.64**	**0.20**	**1.04**	0.39
line	MAE	0.81	0.53	1.95	**0.24**
line	MSE	0.66	0.28	1.26	**0.34**

(a) Impact of Topology on 7 learners configuration.

Topology	Metric	Mean	Var	Max	Min
cycle	MAE	1.51	1.46	6.16	0.36
cycle	MSE	0.94	0.38	1.90	0.48
complete	MAE	**1.26**	**0.82**	3.64	**0.32**
complete	MSE	**0.85**	**0.27**	**1.50**	**0.42**
line	MAE	1.38	0.91	3.17	0.50
line	MSE	0.90	0.35	**1.66**	0.49

(b) Impact of Topology on 13 learners configuration.

Table 1: Temperature forecasting performances on different network topologies

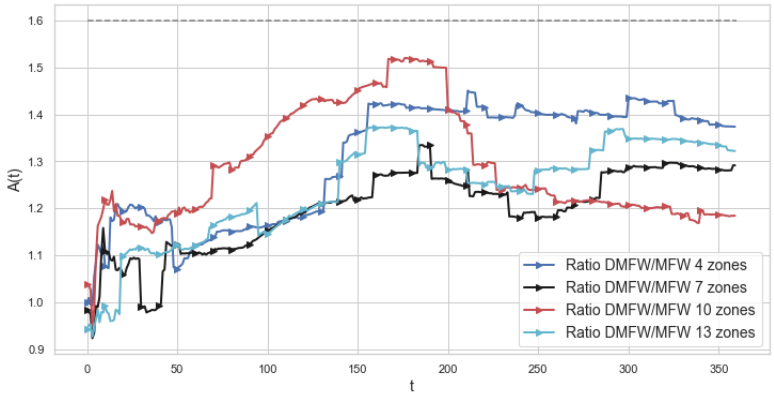


Fig. 2: Loss ratio of decentralized and centralized Meta Frank-Wolfe on different network size.

21 hours, the approximation ratio becomes close to centralised version with less than 20 % error.

4 Concluding remarks

We proposed an online algorithm minimizing non-convex loss functions aggregated from local data distributed over a network. We showed the bounds of the convergence gap in both exact and stochastic gradient settings. In complement to the theoretical analysis, we run experiments on a real-life smart building dataset. The results make our offerings valuable for learning in distributed settings.

References

1. Abdel-Aziz, H., Koutsoukos, X.: Data-driven online learning and reachability analysis of stochastic hybrid systems for smart buildings. Cyber-Physical Systems **5**(1), 41–64 (2019)
2. Cai, M., Pipattanasomporn, M., Rahman, S.: Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. Applied energy **236**, 1078–1088 (2019)
3. Chen, L., Hassani, H., Karbasi, A.: Online continuous submodular maximization. In: Proc. 21st International Conference on Artificial Intelligence and Statistics (AISTAT) (2018)
4. Chiliang, Z., Tao, H., Yingda, G., Zuochang, Y.: Accelerating convolutional neural networks with dynamic channel pruning. In: 2019 Data Compression Conference (DCC). pp. 563–563 (2019)
5. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**(1), 21–27 (1967)
6. Gupta, C., Suggala, A.S., Goyal, A., Simhadri, H.V., Paranjape, B., Kumar, A., Goyal, S., Udupa, R., Varma, M., Jain, P.: ProtoNN: Compressed and accurate kNN for resource-scarce devices. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 1331–1340. PMLR (06–11 Aug 2017)
7. Gupta, S.K., Kar, K., Mishra, S., Wen, J.T.: Distributed consensus algorithms for collaborative temperature control in smart buildings. In: 2015 American Control Conference (ACC). pp. 5758–5763. IEEE (2015)
8. Hazan, E.: Introduction to online convex optimization. Foundations and Trends® in Optimization **2**(3-4), 157–325 (2016)
9. He, L., Bian, A., Jaggi, M.: Cola: Decentralized linear learning. In: Advances in Neural Information Processing Systems. pp. 4536–4546 (2018)
10. Hosseini, S., Chapman, A., Mesbahi, M.: Online distributed optimization via dual averaging. In: 52nd IEEE Conference on Decision and Control. pp. 1484–1489 (2013)
11. Jaggi, M.: Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In: Proceedings of the 30th International Conference on Machine Learning (2013)
12. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., et al.: Advances and open problems in federated learning. Foundations and Trends® in Machine Learning **14**(1) (2021)
13. Lian, X., Zhang, C., Zhang, H., Hsieh, C.J., Zhang, W., Liu, J.: Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In: Advances in Neural Information Processing Systems. pp. 5330–5340 (2017)
14. Lin, J., Rao, Y., Lu, J., Zhou, J.: Runtime neural pruning. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017)
15. MacCarthy, M.: In defense of big data analytics. The Cambridge Handbook of Consumer Privacy pp. 47–78 (2018)
16. McMahan, B., Ramage, D.: Collaborative machine learning without centralized training data. Google Research Blog **3** (2017)
17. Mitra, A., Thang, N.K., Nguyen, T.A., Trystram, D., Youssef, P.: Online Decentralized Frank-Wolfe: From theoretical bound to applications in smart-building (Apr 2022)

18. Nan, F., Wang, J., Saligrama, V.: Pruning random forests for prediction on a budget. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 29. Curran Associates, Inc. (2016)
19. Pipattanasomporn, M., Chitalia, G., Songsiri, J., Aswakul, C., Pora, W., Suwankawin, S., Audomvongseree, K., Hoonchareon, N.: Cu-bems, smart building electricity consumption and indoor environmental sensor datasets. Scientific Data (2020)
20. Shotton, J., Sharp, T., Kohli, P., Nowozin, S., Winn, J., Criminisi, A.: Decision jungles: Compact and rich models for classification. In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (eds.) Advances in Neural Information Processing Systems. vol. 26. Curran Associates, Inc. (2013)
21. Simons, T., Lee, D.J.: A review of binarized neural networks. Electronics **8**(6) (2019)
22. Wai, H., Lafond, J., Scaglione, A., Moulines, E.: Decentralized frank–wolfe algorithm for convex and nonconvex problems. IEEE Transactions on Automatic Control **62**(11), 5522–5537 (2017)
23. Wang, S., Tuor, T., Salonidis, T., Leung, K.K., Makaya, C., He, T., Chan, K.: When edge meets learning: Adaptive control for resource-constrained distributed machine learning. In: IEEE INFOCOM 2018 - IEEE Conference on Computer Communications. pp. 63–71 (2018)
24. Yan, F., Sundaram, S., Vishwanathan, S.V.N., Qi, Y.: Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. IEEE Transactions on Knowledge and Data Engineering **25**(11), 2483–2493 (2013)
25. Zamora-Martinez, F., Romeu, P., Botella-Rocamora, P., Pardo, J.: On-line learning of indoor temperature forecasting models towards energy efficiency. Energy and Buildings **83**, 162–172 (2014)
26. Zhang, W., Zhao, P., Zhu, W., Hoi, S., Zhang, T.: Projection-free distributed online learning in networks. In: Proceedings of the 34th International Conference on Machine Learning. pp. 4054–4062 (2017)