Decentralized Meta Frank-Wolfe For Neural Network Optimization

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

[1]Grenoble Laboratory of Informatics, Université Grenoble-Alpes

[2]IBISC, Université d'Evry, Paris Saclay

November 2021

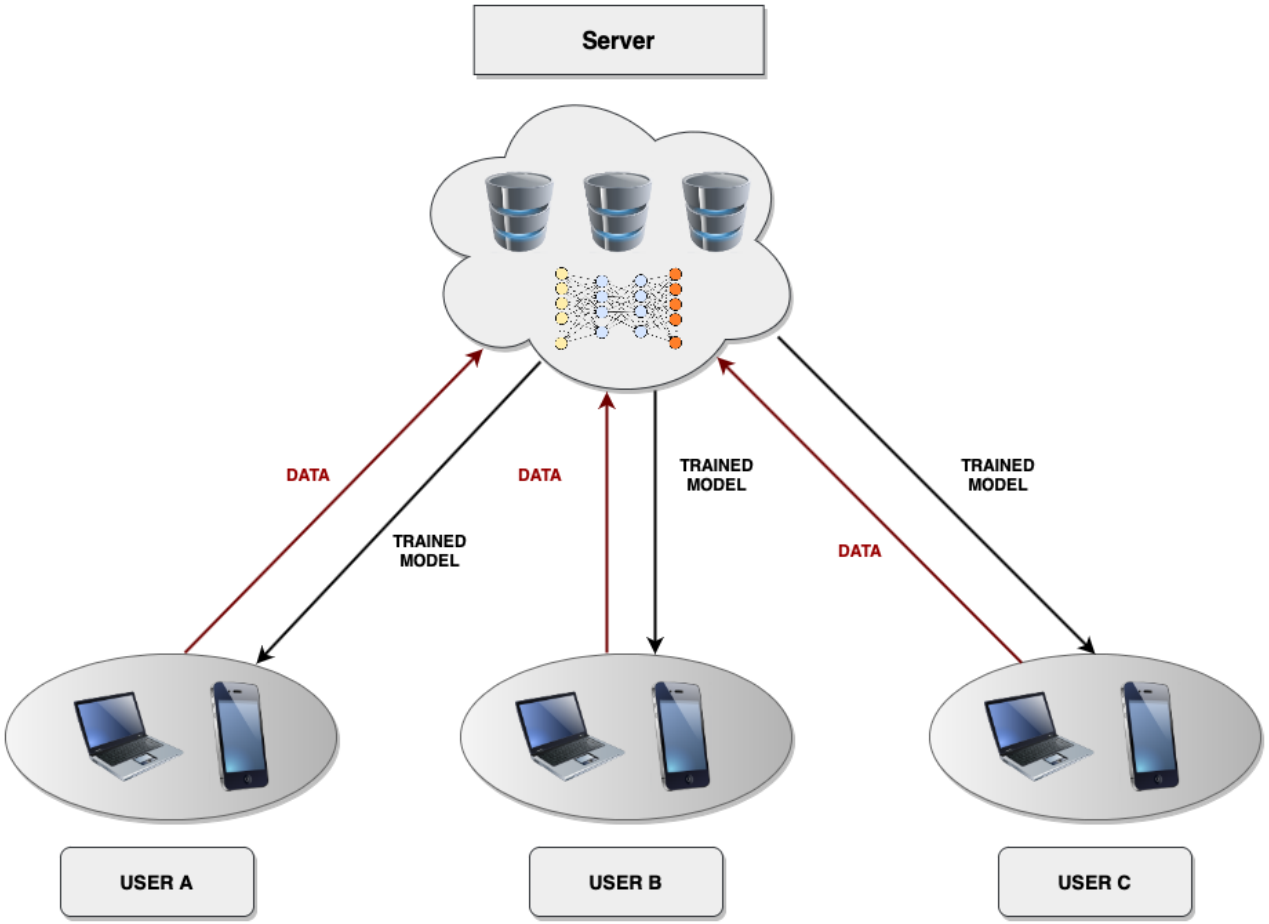
1 Introduction

2 Decentralized Meta Frank-Wolfe

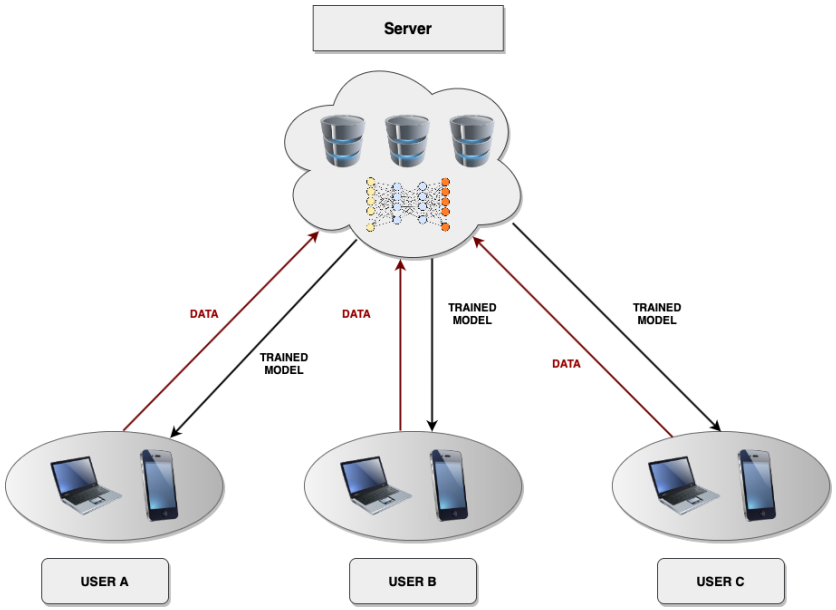
3 Conclusion

4 Future Direction

Centralized Learning



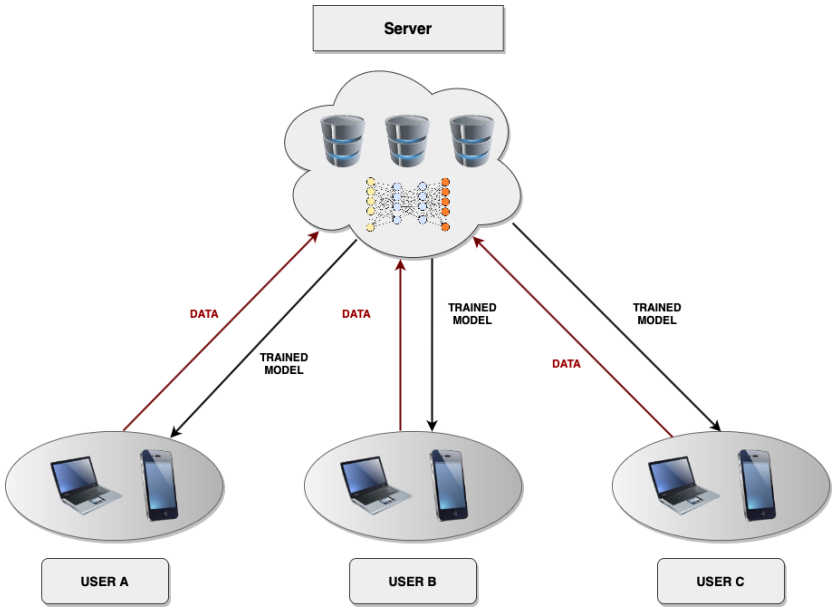
Centralized Learning



Drawbacks

- Privacy Preserving

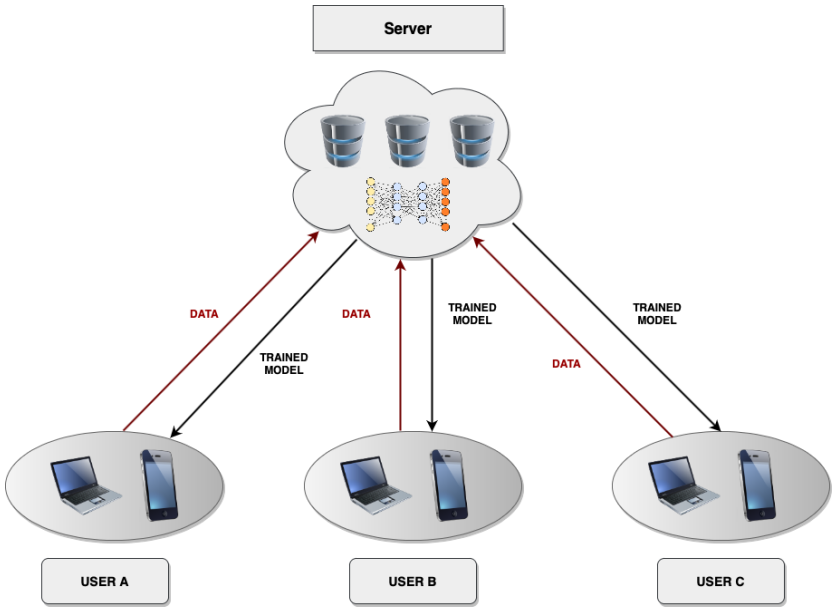
Centralized Learning



Drawbacks

- Privacy Preserving
- Data storage costs

Centralized Learning



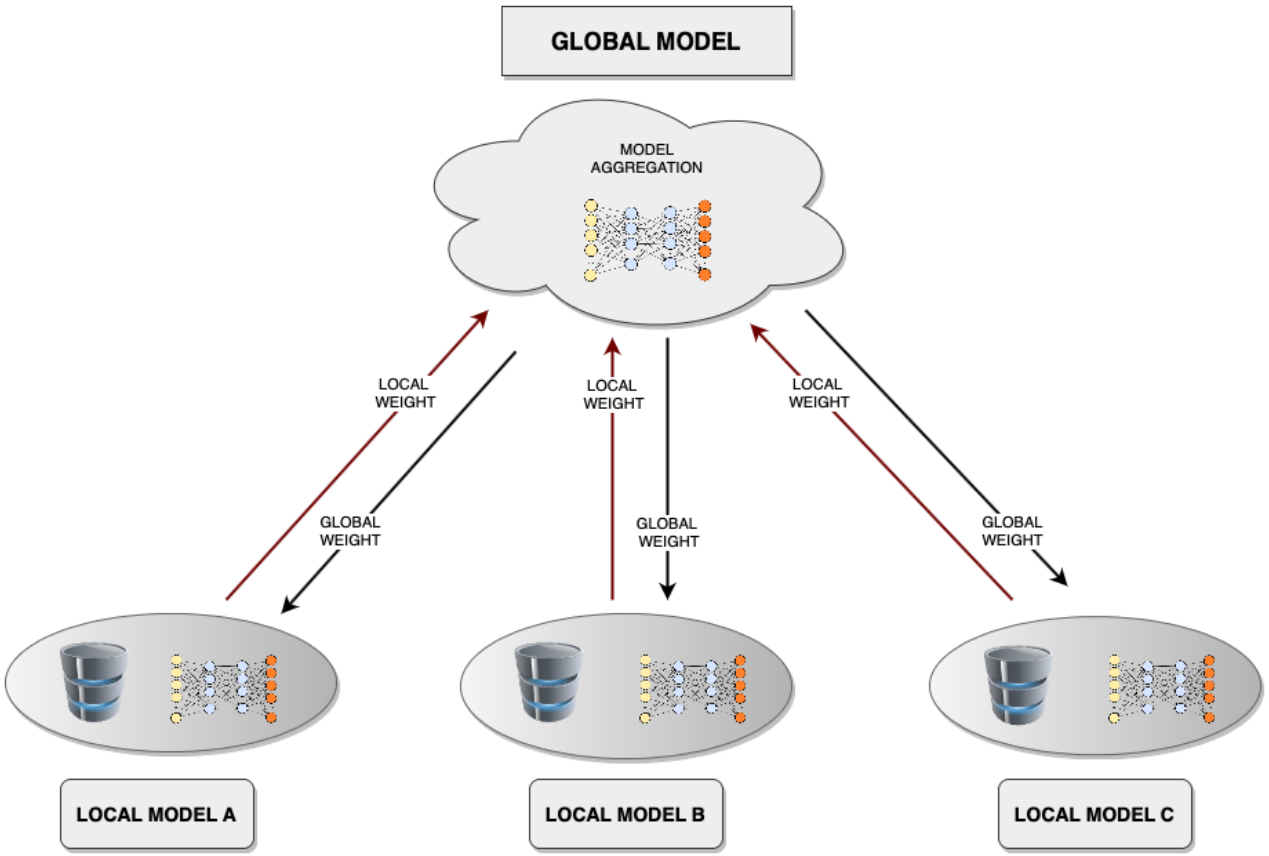
Drawbacks

- Privacy Preserving
- Data storage costs
- Computational resources

Federated Learning

Solution

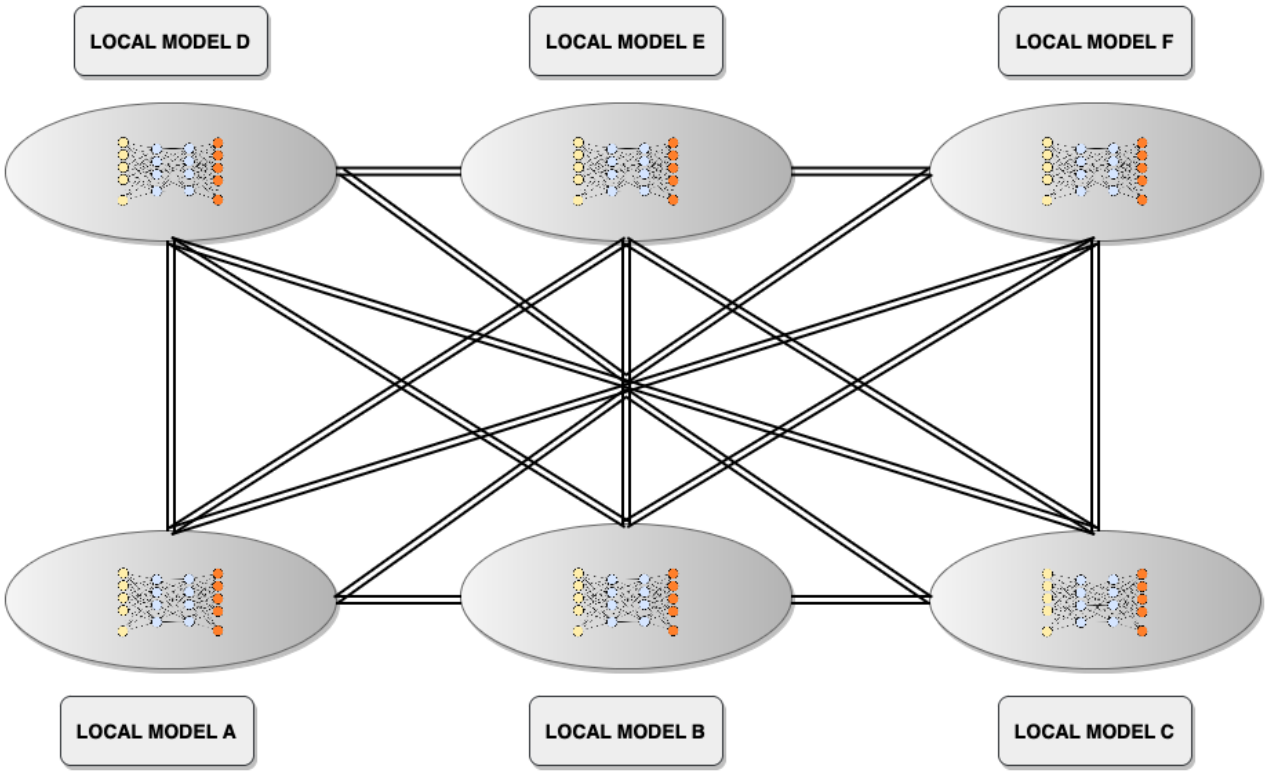
→ Distributed/Federated Learning [MMR+17]



Decentralized Online Learning

Our studies

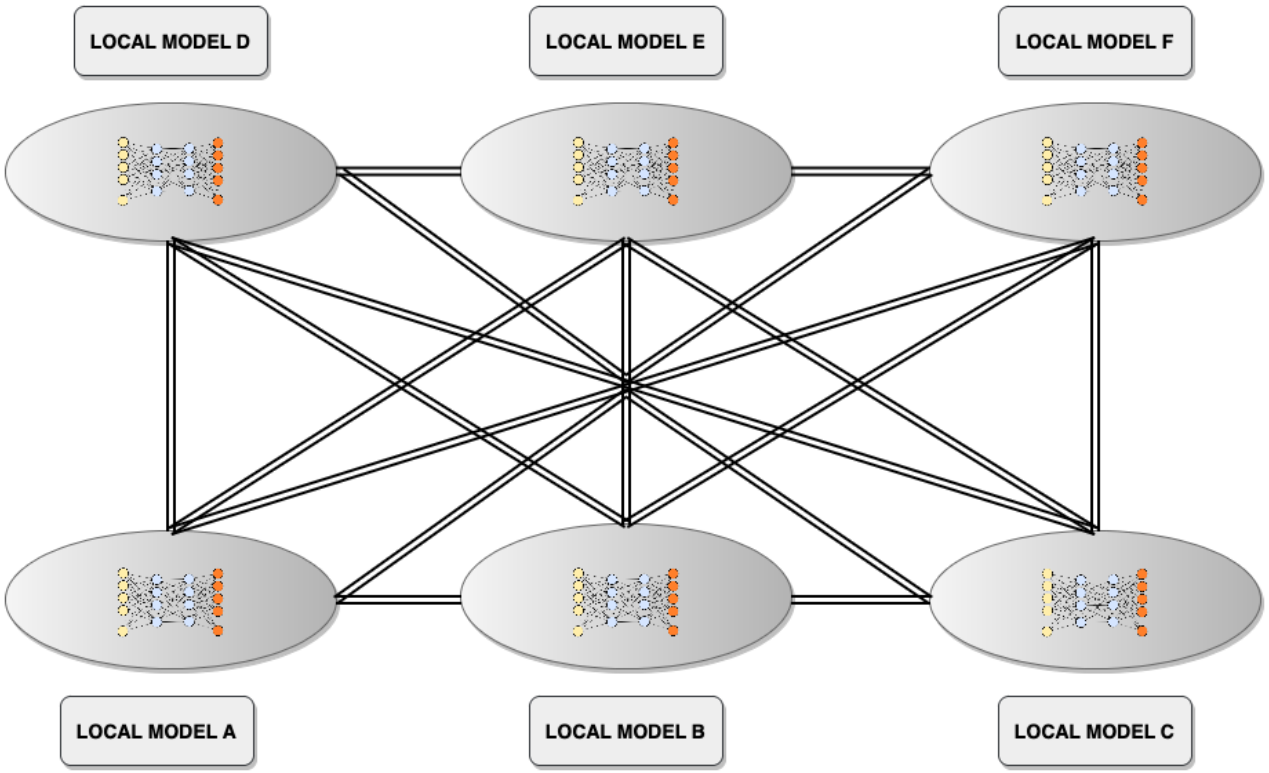
- Fully Decentralized Federated Learning



Decentralized Online Learning

Our studies

- Fully Decentralized Federated Learning
- Online Decision Making



1 Introduction

2 Decentralized Meta Frank-Wolfe
- Problem Setting
- Algorithm Overview
- Experiments

3 Conclusion

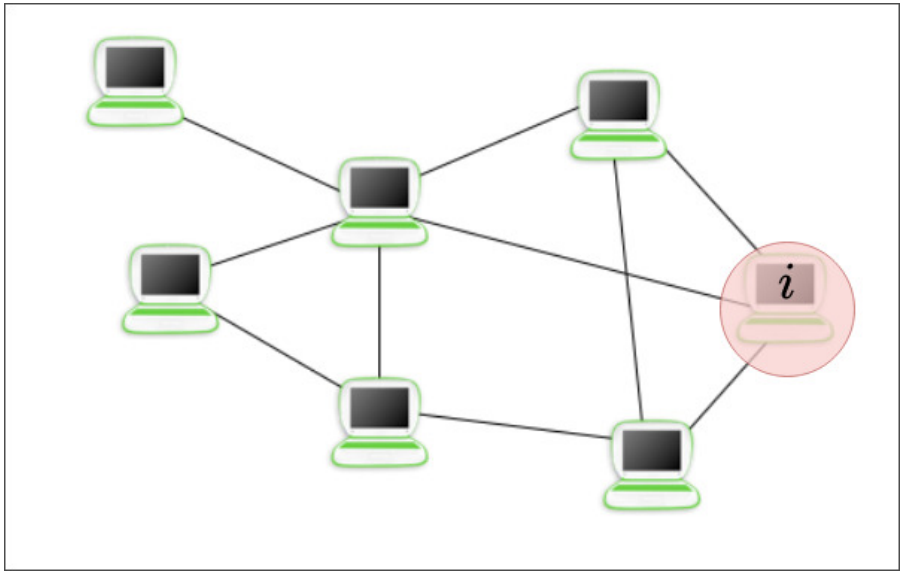
4 Future Direction

1 Introduction

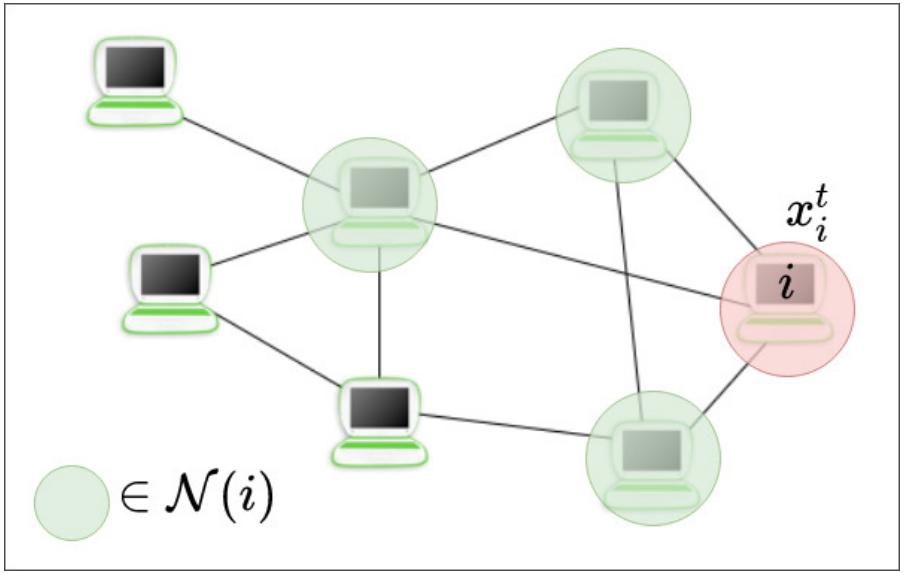
2 Decentralized Meta Frank-Wolfe
- Problem Setting
- Algorithm Overview
- Experiments

3 Conclusion

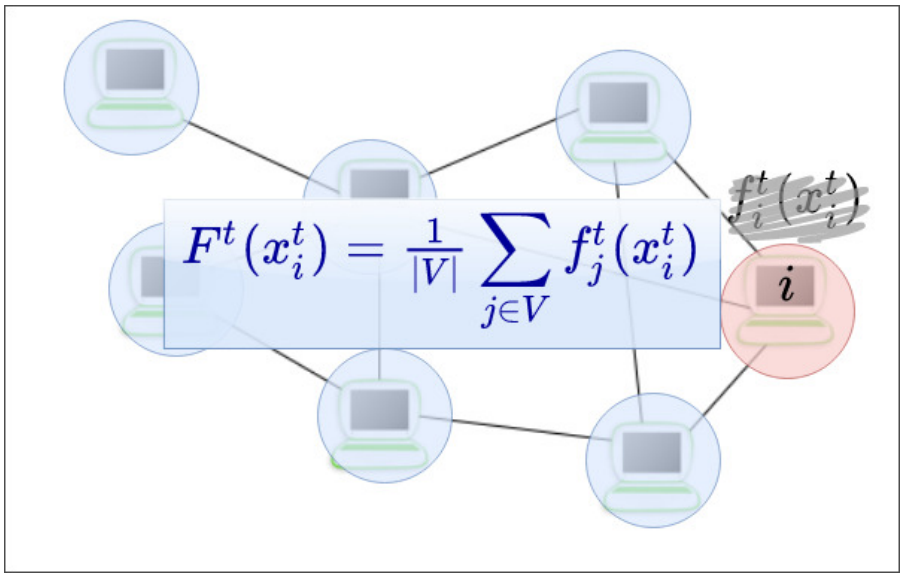
4 Future Direction



- Set V of n **agents connected** over a **network** $G = (V, E)$



- $G = (V, E)$
- **local communications**
- **at round t :**
  - each agent i takes a **decision** x_i^t **satisfying** $x_i^t \in \mathcal{K}$ where $\mathcal{K} \in \mathbb{R}^d$ is a convex set



- $G = (V, E)$
- **local communications**
- **at round t :**
  - **decision :** $x_i^t \in \mathcal{K}$
  - **partial cost :** $f_i^t(x_i^t)$
  - **the objective function :** $F^t(x_i^t) := \frac{1}{|V|} \sum_{j \in V} f_j^t(x_i^t)$

$R(T)$-regret

$$\frac{1}{T}\left(\sum_{t=1}^{T} F^t(x_i^t) - \min_{x\in\mathcal{K}} \sum_{t=1}^{T} F^t(x)\right) \leq R(T) \qquad \forall i \in V$$

$R(T)$-regret

$$\frac{1}{T}\left(\sum_{t=1}^{T} F^t(x_i^t) - \min_{x\in\mathcal{K}} \sum_{t=1}^{T} F^t(x)\right) \leq R(T) \qquad \forall i \in V$$

Convergence Gap [LJ16] [Jag13]

$$\max_{v\in\mathcal{K}} \frac{1}{T} \sum_{t=1}^{T} \langle \nabla F^t(x_i^t), x_i^t - v\rangle \qquad \forall i \in V.$$

1 Introduction

2 Decentralized Meta Frank-Wolfe
- Problem Setting
- Algorithm Overview
- Experiments

3 Conclusion

4 Future Direction

DFW *(Decentralized Frank-Wolfe)* $+$ **MFW** *(Meta Frank-Wolfe)* $=$ **DMFW**

FW(Offline)

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$

   (or $v^t \leftarrow \text{argmax}_{v \in \mathcal{K}} < -\nabla F(x^t), v >$)

2. Update $x^{t+1} \leftarrow x^t + \eta^t(v^t - x^t)$

DFW (*Decentralized Frank-Wolfe*) $+$ **MFW** (*Meta Frank-Wolfe*) $=$ **DMFW**

FW(Centralized) $\rightarrow$ **DFW**(Decentralized) [WLSM17]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

Decentralized version

1. Compute $\overline{x}_i^t \leftarrow \text{LocalNetworkAvg}_i(\{x_j^t\}_{j \in \mathcal{N}(i)})$

DFW (*Decentralized Frank-Wolfe*) $+$ **MFW** (*Meta Frank-Wolfe*) $=$ **DMFW**

FW(Centralized) $\rightarrow$ **DFW**(Decentralized) [WLSM17]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

Decentralized version

1. Compute $\overline{x}_i^t \leftarrow \text{LocalNetworkAvg}_i(\{x_j^t\}_{j \in \mathcal{N}(i)})$
2. Compute $\overline{\nabla F}^t \leftarrow \text{LocalNetworkAvg}_i(\{\nabla F(x_j^t)\}_{j \in \mathcal{N}(i)})$

DFW (*Decentralized Frank-Wolfe*) $+$ **MFW** (*Meta Frank-Wolfe*) $=$ **DMFW**

FW(Centralized) $\rightarrow$ **DFW**(Decentralized) [WLSM17]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

Decentralized version

1. Compute $\overline{x}_i^t \leftarrow \text{LocalNetworkAvg}_i(\{x_j^t\}_{j \in \mathcal{N}(i)})$
2. Compute $\overline{\nabla F}^t \leftarrow \text{LocalNetworkAvg}_i(\{\nabla F(x_j^t)\}_{j \in \mathcal{N}(i)})$
3. Similarly to Frank Wolfe $\Rightarrow$
   1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \overline{\nabla F}^t, v >$
   2. Update $x^{t+1} \leftarrow (1 - \eta^t)\overline{x}_i^t + \eta^t v^t$

DFW *(Decentralized Frank-Wolfe)* $+$ **MFW** *(Meta Frank-Wolfe)* $=$ **DMFW**

FW(Offline) $\rightarrow$ **MFW**(Online) [CHK18]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

- We want the best decision for each function F^t

DFW *(Decentralized Frank-Wolfe)* $+$ **MFW** *(Meta Frank-Wolfe)* $=$ **DMFW**

FW(Offline) $\rightarrow$ **MFW**(Online) [CHK18]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

- We want the best decision for each function F^t
  - $\Rightarrow$ Use Frank-Wolfe update in (2) multiple times

DFW *(Decentralized Frank-Wolfe)* $+$ **MFW** *(Meta Frank-Wolfe)* $=$ **DMFW**

FW(Offline) $\rightarrow$ **MFW**(Online) [CHK18]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

- We want the best decision for each function F^t
  - $\Rightarrow$ Use Frank-Wolfe update in (2) multiple times
- **Problem :** Can not compute (1) since $\nabla F^t(x^t)$ is only known after playing some point for F^t.

DFW *(Decentralized Frank-Wolfe)* $+$ **MFW** *(Meta Frank-Wolfe)* $=$ **DMFW**

FW(Offline) $\rightarrow$ **MFW**(Online) [CHK18]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

- We want the best decision for each function F^t
  - $\Rightarrow$ Use Frank-Wolfe update in (2) multiple times
- **Problem :** Can not compute (1) since $\nabla F^t(x^t)$ is only known after playing some point for F^t.
- **Solution :** use L **online optimization oracles** (gradient descent, FTPL, ...)

DFW *(Decentralized Frank-Wolfe)* $+$ **MFW** *(Meta Frank-Wolfe)* $=$ **DMFW**

FW(Offline) $\rightarrow$ **MFW**(Online) [CHK18]

1. Compute $v^t \leftarrow \arg\min_{v \in \mathcal{K}} < \nabla F(x^t), v >$
2. Update $x^{t+1} \leftarrow (1 - \eta^t)x^t + \eta^t v^t$

- We want the best decision for each function F^t
  - $\Rightarrow$ Use Frank-Wolfe update in (2) multiple times
- **Problem :** Can not compute (1) since $\nabla F^t(x^t)$ is only known after playing some point for F^t.
- **Solution :** use L **online optimization oracles** (gradient descent, FTPL, ...)
- $v_\ell^t \leftarrow$ **output of oracle** $\mathcal{O}_\ell$

Algorithm

Input: $\mathcal{K}$, $\mathcal{O}_{i,1}, \ldots, \mathcal{O}_{i,L}$, η_ℓ

for $t = 1$ to T **do**
  for every agent $1 \leq i \leq n$ **do**
    for $1 \leq \ell \leq L$ **do**
      Get output of $\mathcal{O}_{i,\ell}$
      LocalNetworkAvg$_i(\{x_{j,\ell}^t\}_{j \in \mathcal{N}(i)})$
      Frank-Wolfe Update
    end for
    Choose x_i^t uniformly random in $\{x_{i,1}^t, \cdots x_{i,L}^t\}$
    Receive f_i^t
    for $1 \leq \ell \leq L$ **do**
      LocalNetworkAvg$_i(\{\nabla F(x_j^t)\}_{j \in \mathcal{N}(i)})$
      Update Gradient
      Feedback to oracles $\mathcal{O}_{i,\ell}$
    end for
  end for
end for

Theoretical Results

- convergence gap of $\mathcal{O}(T^{-1/2})$ when choosing number of oracles $L = T$ with regret $\mathcal{O}(\sqrt{T})$

Theoretical Results

- convergence gap of $\mathcal{O}(T^{-1/2})$ when choosing number of oracles $L = T$ with regret $\mathcal{O}(\sqrt{T})$
- using stochastic gradient estimates, convergence gap of $\mathcal{O}(T^{-1/4})$.

Theoretical Results

- convergence gap of $\mathcal{O}(T^{-1/2})$ when choosing number of oracles $L = T$ with regret $\mathcal{O}(\sqrt{T})$
- using stochastic gradient estimates, convergence gap of $\mathcal{O}(T^{-1/4})$.
- flexibility to be converted into **projection-free** (using FTPL as oracles)

1 Introduction

2 Decentralized Meta Frank-Wolfe
- Problem Setting
- Algorithm Overview
- Experiments

3 Conclusion

4 Future Direction

Data

- Building dataset contains temperature time-series of 13 zones on 4 floors
- At every round t, each node receive a batch $\mathcal{B}_i^t$ of 32 time-series sequences.
- With 5 minutes resolution, using 1 hour past data (13 timesteps) to predict the next 5 minute.

Model

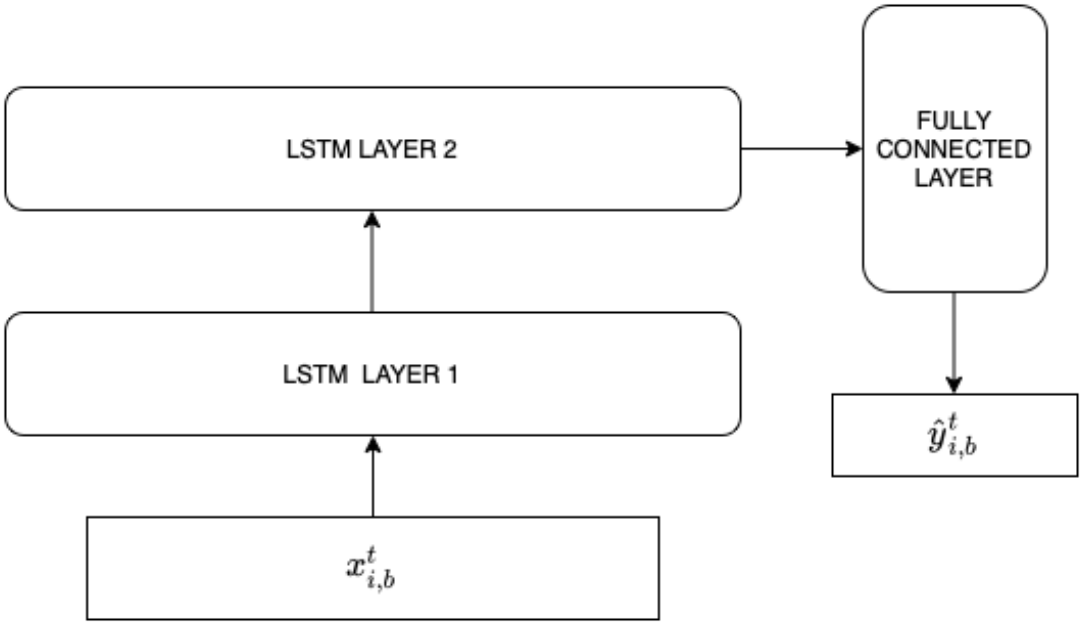


Figure 1: Model Diagram

$\forall b \in \mathcal{B}_i^t$

- $x_{i,b}^t$: input time-series.
- $\hat{y}_{i,b}^t$: model output.

Model

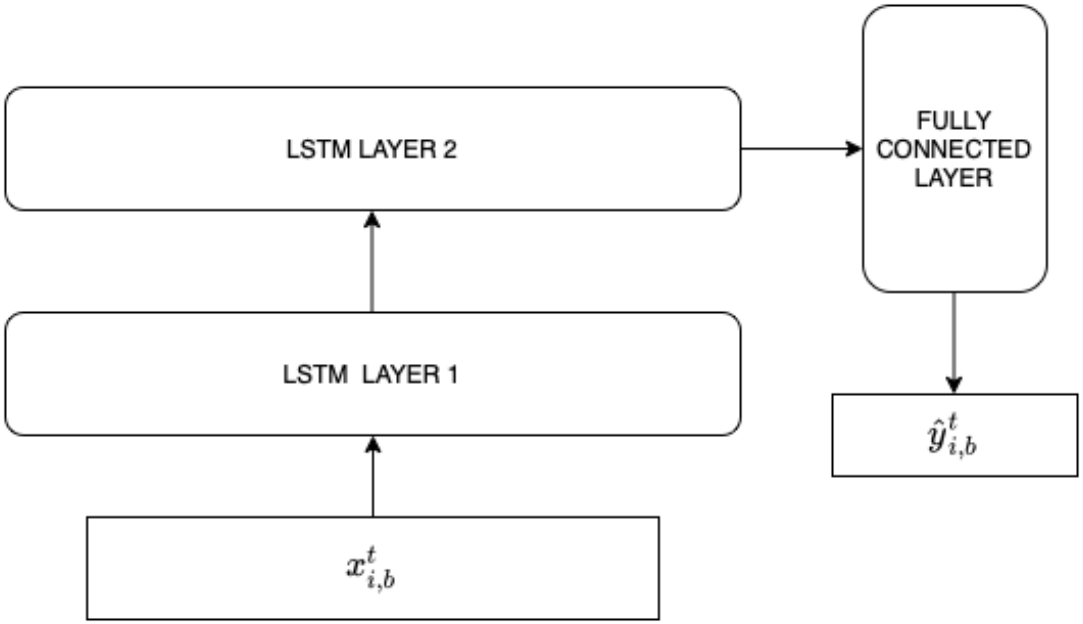


Figure 1: Model Diagram

$\forall b \in \mathcal{B}_i^t$

- $x^t_{i,b}$: input time-series.
- $\hat{y}^t_{i,b}$: model output.
- $y^t_{i,b}$: ground truth.
- $\mathcal{L}(\hat{y}^t_{i,b}, y^t_{i,b})$: $L1$ loss

Model

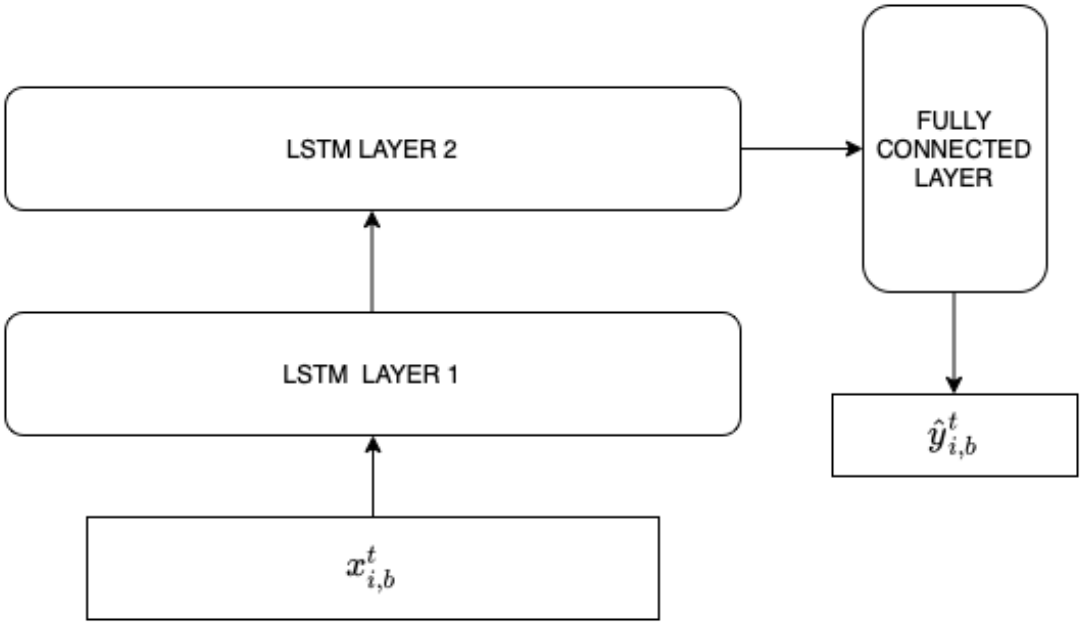


Figure 1: Model Diagram

$\forall b \in \mathcal{B}_i^t$

- $x_{i,b}^t$: input time-series.
- $\hat{y}_{i,b}^t$: model output.
- $y_{i,b}^t$: ground truth.
- $\mathcal{L}(\hat{y}_{i,b}^t, y_{i,b}^t)$: $L1$ loss

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

Prediction Performance

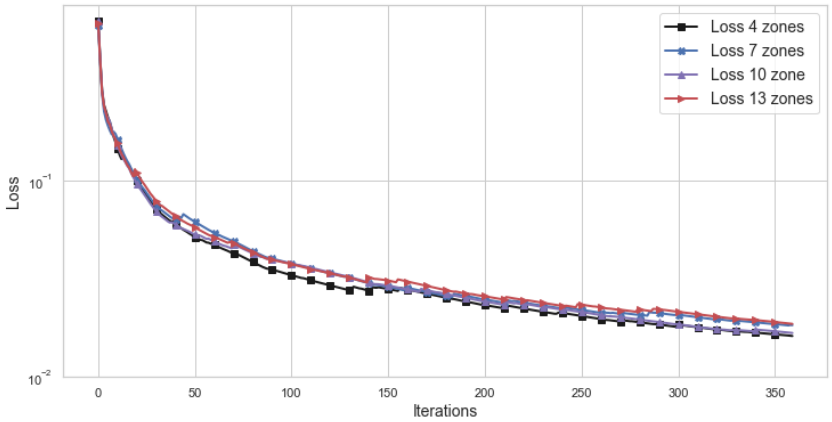


Figure 2: Loss values of different network size on complete topology. *(Plot on log-scale)*

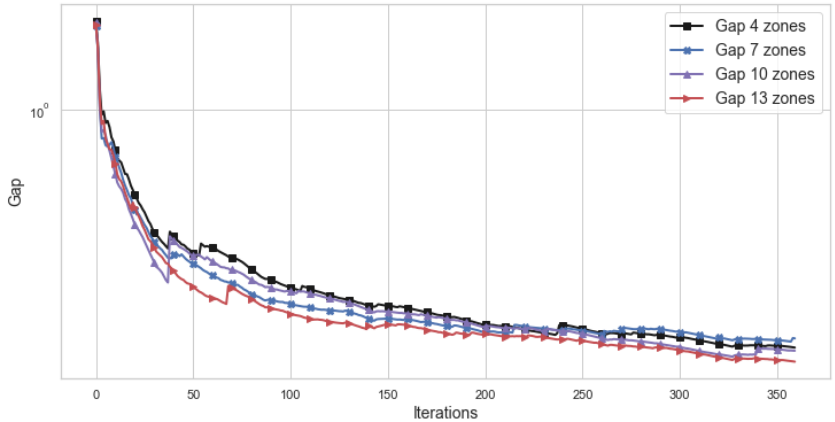


Figure 3: Gap values of different network size on complete topology. *(Plot on log-scale)*

Impact of Network Topology

Topology	Metric	Mean	Variance	Max	Min
cycle	MAE	1.511	1.456	6.159	0.361
cycle	MSE	0.938	0.384	1.897	0.483
complete	MAE	**1.257**	**0.82**	3.64	**0.32**
complete	MSE	**0.852**	**0.272**	**1.505**	0.417
line	MAE	1.385	0.915	3.169	0.5
line	MSE	0.905	0.352	**1.664**	0.492

Table 1: Impact of Topology on Temperature Forecasting Performance with 13 learners.

Effect of Decentralization

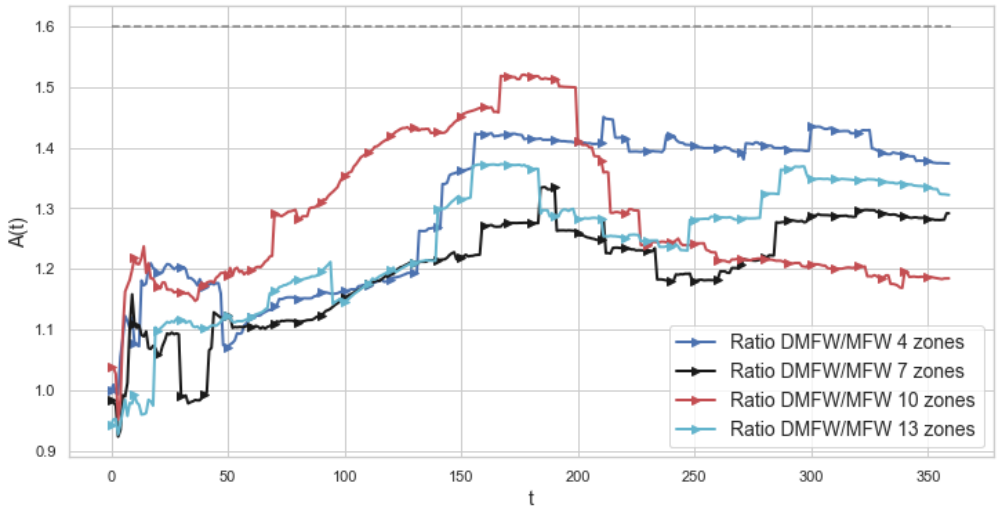


Figure 4: Loss ratio of decentralized and centralized Meta Frank-Wolfe on different network size.

Approximation Ratio

$$A(t) = \frac{L_{DMFW}(t)}{L_{MFW}(t)}$$

Effect of Decentralization

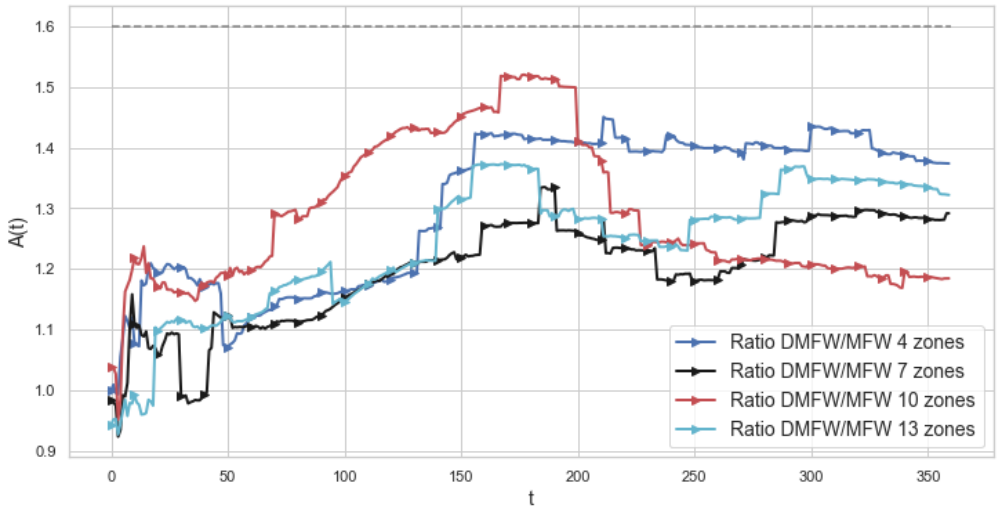


Figure 4: Loss ratio of decentralized and centralized Meta Frank-Wolfe on different network size.

Approximation Ratio

$$A(t) = \frac{L_{DMFW}(t)}{L_{MFW}(t)}$$

When $A(t) \leq B_{max}$, DMFW performs no worse than B_{max} times of MFW.

1 Introduction

2 Decentralized Meta Frank-Wolfe

3 Conclusion

4 Future Direction

Conclusion

What we propose ?

- Online algorithm minimizing non-convex loss functions aggregated from local data distributed over a network

Conclusion

What we propose ?

- Online algorithm minimizing non-convex loss functions aggregated from local data distributed over a network
  - Provable performance guarantee.

Conclusion

What we propose ?

- Online algorithm minimizing non-convex loss functions aggregated from local data distributed over a network
  - Provable performance guarantee.
  - Flexibility to become projection-free

Conclusion

What we propose ?

- Online algorithm minimizing non-convex loss functions aggregated from local data distributed over a network
  - Provable performance guarantee.
  - Flexibility to become projection-free
  - Robustness to stochastic gradient estimates

Conclusion

What we propose ?

- Online algorithm minimizing non-convex loss functions aggregated from local data distributed over a network
  - Provable performance guarantee.
  - Flexibility to become projection-free
  - Robustness to stochastic gradient estimates
  - Experiments prove a practical use case of the algorithm in an online prediction problem.

1 Introduction

2 Decentralized Meta Frank-Wolfe

3 Conclusion

4 Future Direction

Future Direction

Communication Reduction

- Using a blocking procedure $T = QL$ where Q is number of blocks, L is number of functions inside block Q.
- Query only one gradient per function.

Future Direction

Communication Reduction

- Using a blocking procedure $T = QL$ where Q is number of blocks, L is number of functions inside block Q.
- Query only one gradient per function.

Goal : less gradient computation $\Rightarrow$ less communication needed

Thanks!

[CHK18] Lin Chen, Hamed Hassani, and Amin Karbasi.
Online continuous submodular maximization.
In *Proc. 21st International Conference on Artificial Intelligence and Statistics (AISTAT)*, 2018.

[Jag13] Martin Jaggi.
Revisiting Frank-Wolfe: Projection-free sparse convex optimization.
In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[LJ16] Simon Lacoste-Julien.
Convergence rate of frank-wolfe for non-convex objectives, 2016.

[MMR+17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas.
Communication-Efficient Learning of Deep Networks from Decentralized Data.
In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research. PMLR, 2017.

[WLSM17] H. Wai, J. Lafond, A. Scaglione, and E. Moulines.
Decentralized frank–wolfe algorithm for convex and nonconvex problems.
IEEE Transactions on Automatic Control, 2017.