Distributed Sequence Prediction: A Consensus+Innovations approach

Anit Kumar Sahu

Joint work with Soummya Kar

Department of Electrical and Computer Engineering
Carnegie Mellon University

8th December, 2016

Electrical & Computer ENGINEERING

Motivation

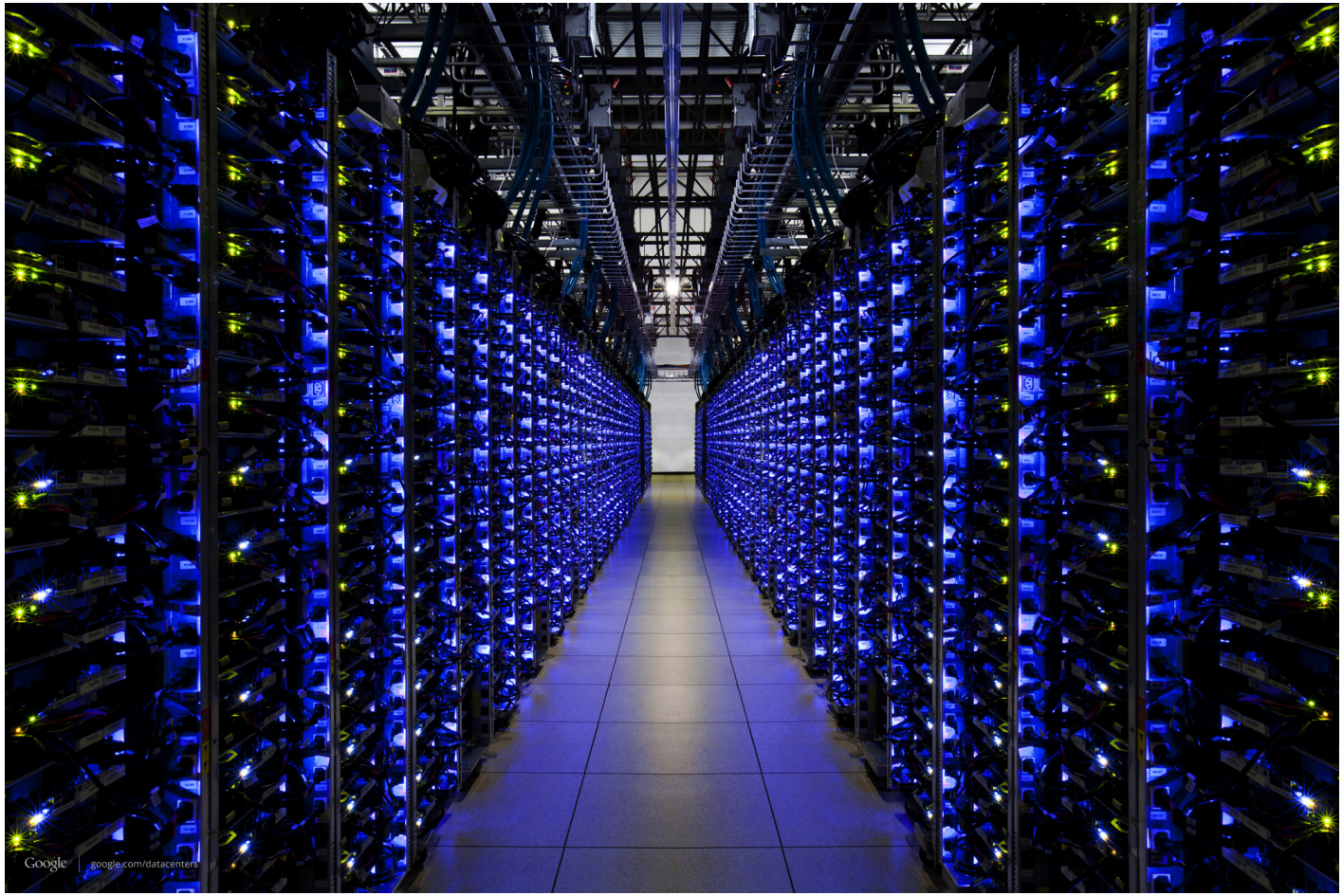


Image Source: Google Datacenter

Electrical & Computer ENGINEERING

Problem: Why Online Learning?

Data is huge: Hard to take a single pass across all the data points.

Problem: Why Online Learning?

Data is huge: Hard to take a single pass across all the data points.

Hard to verify any assumptions that we need so as to apply a provably efficient learning algorithm.

Problem: Why Online Learning?

Data is huge: Hard to take a single pass across all the data points.

Hard to verify any assumptions that we need so as to apply a provably efficient learning algorithm.

Even estimating the distribution is difficult.

Problem: Why Online Learning?

Data is huge: Hard to take a single pass across all the data points.

Hard to verify any assumptions that we need so as to apply a provably efficient learning algorithm.

Even estimating the distribution is difficult.

In such situations, it is hard to figure out which algorithm to use.

Expert-Assisted Learning

The task here is to predict the next outcome of a sequence.

Expert-Assisted Learning

The task here is to predict the next outcome of a sequence.

The distribution from which the sequence is drawn is unknown.

Expert-Assisted Learning

The task here is to predict the next outcome of a sequence.

The distribution from which the sequence is drawn is unknown.

Suppose the experts are some reference sequence predictors.

Expert-Assisted Learning

The task here is to predict the next outcome of a sequence.

The distribution from which the sequence is drawn is unknown.

Suppose the experts are some reference sequence predictors.

We would like to weight their predictions in a way so as to get the best prediction performance.

Expert-Assisted Learning : A toy example

Suppose the sequence $\{y_t\}_{t=1}$ is a binary sequence with 0's and 1's.

Expert-Assisted Learning : A toy example

Suppose the sequence $\{y_t\}_{t=1}$ is a binary sequence with 0's and 1's.

We have access to N experts. Suppose only one of them is always right.

Expert-Assisted Learning : A toy example

Suppose the sequence $\{y_t\}_{t=1}$ is a binary sequence with 0's and 1's.

We have access to N experts. Suppose only one of them is always right.

The prediction of the n-th expert at time t is given by $f_{n,t}$.

Expert-Assisted Learning : A toy example

Suppose the sequence $\{y_t\}_{t=1}$ is a binary sequence with 0's and 1's.

We have access to N experts. Suppose only one of them is always right.

The prediction of the n-th expert at time t is given by $f_{n,t}$.

At each time t, the forecaster makes his/her prediction based on the experts' predictions and comes up with a prediction $\widehat{p}_t$.

Expert-Assisted Learning : A toy example

Suppose the sequence $\{y_t\}_{t=1}$ is a binary sequence with 0's and 1's.

We have access to N experts. Suppose only one of them is always right.

The prediction of the n-th expert at time t is given by $f_{n,t}$.

At each time t, the forecaster makes his/her prediction based on the experts' predictions and comes up with a prediction $\widehat{p}_t$.

Our goal is to bound the number of time steps t in which $\widehat{p}_t \neq y_t$.

Expert-Assisted Learning : A toy example

This is basically a binary search problem and hence can be done in $\lceil \log_2 N \rceil$ mistakes.

Expert-Assisted Learning : A toy example

This is basically a binary search problem and hence can be done in $\lceil \log_2 N \rceil$ mistakes.

The forecaster starts by assigning a weight $w_j = 1$ to each expert $j = 1, \cdots, N$.

Expert-Assisted Learning : A toy example

This is basically a binary search problem and hence can be done in $\lceil \log_2 N \rceil$ mistakes.

The forecaster starts by assigning a weight $w_j = 1$ to each expert $j = 1, \cdots, N$.

The forecaster keeps track of which experts make a mistake and predicts according to the majority of the experts that have been always correct.

Expert-Assisted Learning : A toy example

This is basically a binary search problem and hence can be done in $\lceil \log_2 N \rceil$ mistakes.

The forecaster starts by assigning a weight $w_j = 1$ to each expert $j = 1, \cdots, N$.

The forecaster keeps track of which experts make a mistake and predicts according to the majority of the experts that have been always correct.

The key insight here is to track the sum of weights assigned to the different experts.

Related Work

Prediction with expert advice: Littlestone 89', Vovk 90', Cesa-Bianchi 97'

Potential functions in sequential prediction: Cesa-Bianchi 03'

Exponentially weighted average forecasting and Sub-Linear Regret: Cesa-Bianchi 97', Auer 02'

Centralized Sequence Prediction: Expert assisted Learning

- Task: Predict sequences $\{y_t\}_{t=1}$ where $y_t \in \mathcal{Y}$ and $\mathcal{Y}$ is an action space.

Centralized Sequence Prediction: Expert assisted Learning

- Task: Predict sequences $\{y_t\}_{t=1}$ where $y_t \in \mathcal{Y}$ and $\mathcal{Y}$ is an action space.
- Information available: Expert i's prediction at time t, $f_{i,t}$.

Centralized Sequence Prediction: Expert assisted Learning

- Task: Predict sequences $\{y_t\}_{t=1}$ where $y_t \in \mathcal{Y}$ and $\mathcal{Y}$ is an action space.
- Information available: Expert i's prediction at time t, $f_{i,t}$.
- The forecaster's predictions: $\{\hat{p}_i\}$ which belong to a decision space $\mathcal{D}$.

Centralized Sequence Prediction: Expert assisted Learning

- Task: Predict sequences $\{y_t\}_{t=1}$ where $y_t \in \mathcal{Y}$ and $\mathcal{Y}$ is an action space.
- Information available: Expert i's prediction at time t, $f_{i,t}$.
- The forecaster's predictions: $\{\hat{p}_i\}$ which belong to a decision space $\mathcal{D}$.
- Loss Function: The predictions are evaluated using a non-negative loss function $l : \mathcal{D} \times \mathcal{Y} \to \mathbb{R}$.

Centralized Sequence Prediction: Expert assisted Learning

- Task: Predict sequences $\{y_t\}_{t=1}$ where $y_t \in \mathcal{Y}$ and $\mathcal{Y}$ is an action space.
- Information available: Expert i's prediction at time t, $f_{i,t}$.
- The forecaster's predictions: $\{\hat{p}_i\}$ which belong to a decision space $\mathcal{D}$.
- Loss Function: The predictions are evaluated using a non-negative loss function $l : \mathcal{D} \times \mathcal{Y} \to \mathbb{R}$.

Centralized Sequence Prediction: Assumptions

- **Assumption on Loss function**: *The loss function* $l : \mathcal{D} \times \mathcal{Y} \to \mathbb{R}$ *is convex in its first argument and it takes values in* $[0, 1]$.
- **Assumption on Decision space**: *The decision space* $\mathcal{D}$ *is a convex subset of a vector space.*

Electrical & Computer ENGINEERING

Centralized Sequence Prediction: Objective

Forecaster's prediction: $\widehat{p}_t = \sum_{j=1}^{N} w_{j,t-1} f_{j,t}$, where $w_{j,t}$ is chosen on the basis of past performance.

Goal

The goal of the forecaster is to keep the cumulative regret with respect to each *reference forecaster* as low as possible. Formally, the regret with respect to expert n is defined as

$$R_{n,t} = \sum_{s=1}^{t} \left(l\left(\hat{p}_s, y_s\right) - l(f_{n,s}, y_s) \right) = \hat{L}_t - L_{n,t},$$

where $\hat{L}_t = \sum_{s=1}^{t} l\left(\hat{p}_s, y_s\right)$ and $L_{n,t} = \sum_{s=1}^{t} l\left(f_{n,s}, y_s\right)$ which denote the cumulative loss of the forecaster and the expert n at time t respectively.

Exponentially weighted Sequence Prediction

Exponential Weighting Scheme: The weights for the prediction at time t is computed as

$$w_{n,t-1} = \frac{e^{\eta R_{n,t-1}}}{\sum_{n=1}^{N} e^{\eta R_{n,t-1}}},$$

where η is positive and referred to as the *learning parameter*.

Exponentially weighted Sequence Prediction

Exponential Weighting Scheme: The weights for the prediction at time t is computed as

$$w_{n,t-1} = \frac{e^{\eta R_{n,t-1}}}{\sum_{n=1}^{N} e^{\eta R_{n,t-1}}},$$

where η is positive and referred to as the *learning parameter*.

The prediction of the forecaster at time t is given by,

$$\hat{p}_t = \frac{\sum_{n=1}^{N} e^{-\eta L_{n,t-1}} f_{n,t}}{\sum_{n=1}^{N} e^{-\eta L_{n,t-1}}}.$$

Electrical & Computer ENGINEERING

Exponentially weighted Sequence Prediction: Sub-linear Regret

Sub-linear regret over a finite horizon[a]

[a] Prediction, Learning and Games, Cesa-Bianchi et.al.

For any fixed time t and $\eta > 0$, and for all sequence of outcomes $\{y_s\}_{s=1}^{t} \in \mathcal{Y}$ the regret of the exponentially weighted average predictor satisfies

$$\hat{L}_t - \min_{n=1,\cdots,N} L_{n,t} \leq \frac{\ln N}{\eta} + \frac{t\eta}{8}. \tag{1}$$

In particular, if η is chosen to be $\sqrt{\frac{8 \ln N}{t}}$, the right hand side (RHS) in (1) becomes $\sqrt{\frac{t \ln N}{2}}$.

Exponentially weighted Sequence Prediction: Sub-linear Regret

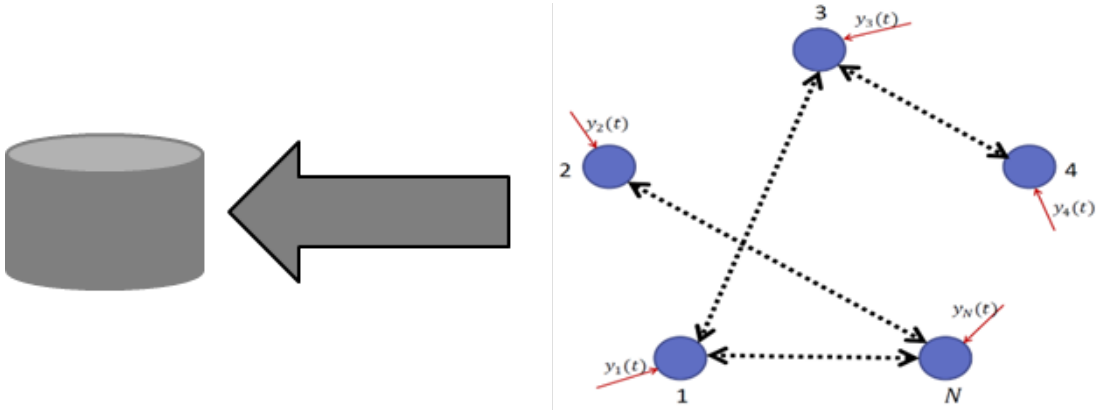
Sub-linear regret for any time[a]

[a] Prediction, Learning and Games, Cesa-Bianchi et.al.

Let $\eta_t = \sqrt{\frac{8 \ln N}{t}}$. For all $t \geq 1$, and for all sequence of outcomes $\{y_s\}_{s=1}^{t} \in \mathcal{Y}$ the regret of the exponentially weighted average predictor satisfies

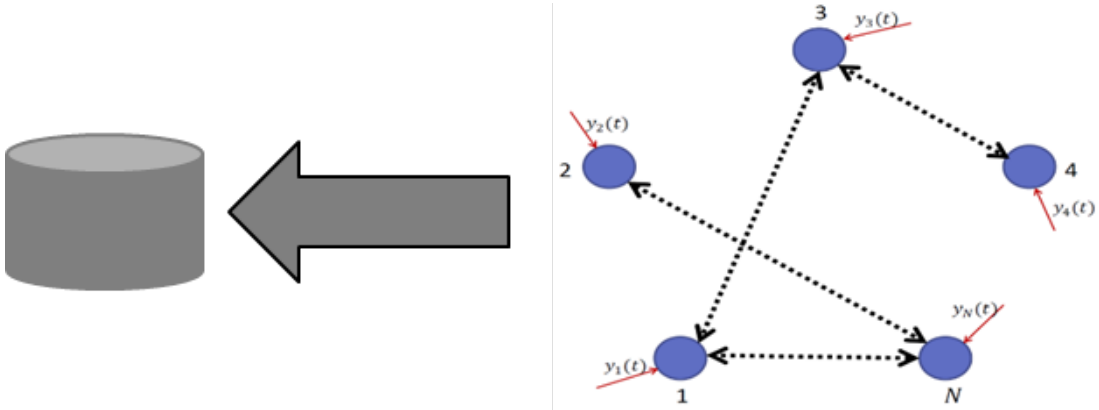
$$\hat{L}_t - \min_{n=1,\cdots,N} L_{n,t} \leq \sqrt{\frac{\ln N}{8}} + \sqrt{2t \ln N}. \tag{2}$$

Problem: Distributed Sequence Prediction



- There are N forecasting agents in the network.

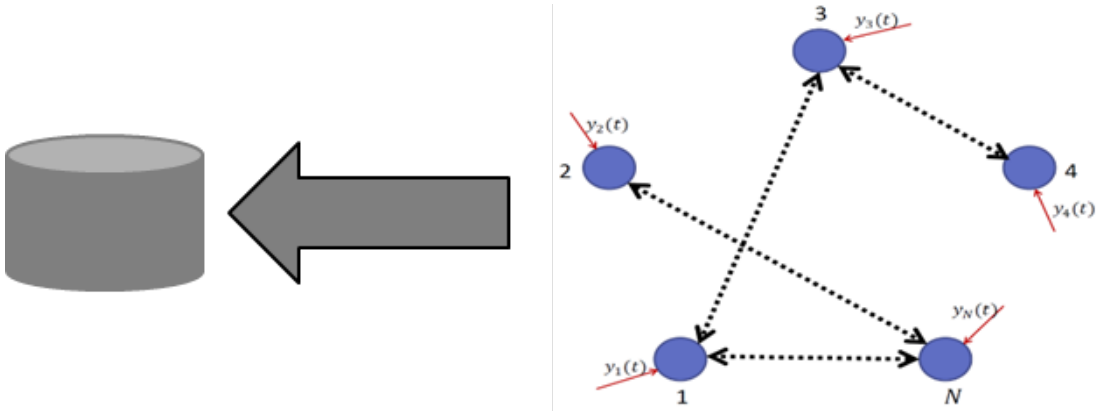
Problem: Distributed Sequence Prediction



- There are N forecasting agents in the network.
- Each agent n has access to its local expert n.

Electrical & Computer ENGINEERING

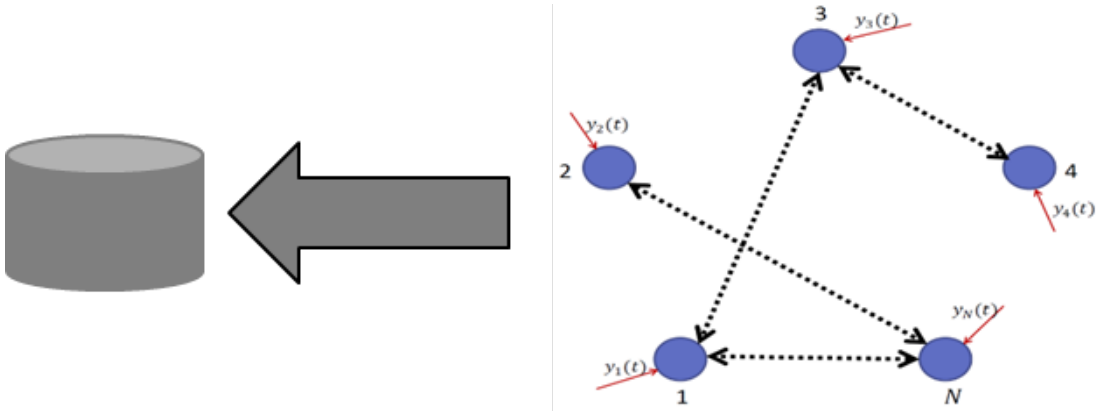
Problem: Distributed Sequence Prediction



- There are N forecasting agents in the network.
- Each agent n has access to its local expert n.
- Information Exchange at time t: Prediction from local expert, a surrogate cumulative loss vector $\mathbf{S}_{d,n}(t) \in \mathbb{R}^N$.

Electrical & Computer ENGINEERING

Problem: Distributed Sequence Prediction



- There are N forecasting agents in the network.
- Each agent n has access to its local expert n.
- Information Exchange at time t: Prediction from local expert, a surrogate cumulative loss vector $\mathbf{S}_{d,n}(t) \in \mathbb{R}^N$.
- Formally, every agent n tracks the network losses across all the experts, albeit in a constrained manner.
- The update for $\mathbf{S}_{d,n}(t)$ can be written as:

$$\mathbf{S}_{d,n}(t+1) = \underbrace{w_{nn}\mathbf{S}_{d,n}(t) + \sum_{l \in \Omega_n} w_{nl}\mathbf{S}_{d,l}(t)}_{\text{neighborhood consensus}} + \underbrace{w_{nn}\mathbf{L}_{d,n}(t) + \sum_{l \in \Omega_n} w_{nl}\mathbf{L}_{d,l}(t)}_{\text{innovation}},$$

Information Exchange

- The information exchange in the update is limited to a pre-specified possibly sparse inter-agent communication graph.
- The weights are designed according to $\mathbf{W} = \mathbf{I} - \delta\mathbf{L}$.
- The inter-agent communication network modeling the information exchange among the forecasting agents is connected.

Distributed Sequence Prediction: Algorithm

- The weight for the forecasting expert l at forecasting agent n at time t, $w_{l,t}^{n}$ is computed as follows:

$$w_{l,t}^{n} = \frac{e^{\mathbf{e}_l^{\top} \eta S_{d,n}(t)}}{\sum_{m=1}^{N} e^{\mathbf{e}_m^{\top} \eta S_{d,n}(t)}},$$

  where η is positive and referred to as the *learning parameter*.
- The forecasting agent doesn't have access to the predictions of other forecasting experts except in its neighborhood.

Electrical & Computer ENGINEERING

Distributed Sequence Prediction: Algorithm

- We assume that there is a *genie* which takes the weights assigned to the forecasting experts at each time t and then computes the prediction of the forecasting agent n.
- Formally the prediction of agent n at time t is given by

$$\hat{p}_{n,t} = \frac{\sum_{l=1}^{N} w_{l,t}^{n} f_{l,t}}{\sum_{l=1}^{N} w_{l,t}^{n}}.$$

Electrical & Computer ENGINEERING

Best Performing Expert: Assumption

- We also assume that the cumulative loss of the best performing forecasting expert satisfies

$$\min_{n=1,\cdots,N} L_{n,t} = o(t)$$

Best Performing Expert: Assumption

- We also assume that the cumulative loss of the best performing forecasting expert satisfies

$$\min_{n=1,\cdots,N} L_{n,t} = o(t)$$

- It ensures that the experts' predictions are reasonable.

Best Performing Expert: Assumption

- We also assume that the cumulative loss of the best performing forecasting expert satisfies

$$\min_{n=1,\cdots,N} L_{n,t} = o(t)$$

- It ensures that the experts' predictions are reasonable.
- In case, when the above assumption does not hold, the performance of not only the distributed algorithm, but also that of the centralized algorithm would be bad, i.e., the cumulative losses would be linear in time.

Main Results: Sub-Linear Regret over a fixed Horizon

Theorem 1

Consider the distributed sequence prediction algorithm . For any fixed time t and $\eta > 0$, and for all sequence of outcomes $\{y_s\}_{s=1}^{t} \in \mathcal{Y}$ the regret of the distributed exponentially weighted average predictor satisfies

$$\hat{L}_{n,t} \leq e^{\frac{2\eta N r}{1-r}} \left(\min_{n=1,\cdots,N} L_{n,t} + \frac{\ln N}{\eta} + \frac{t\eta}{8} + \frac{2\eta N r}{1-r} \right), \tag{3}$$

where $\hat{L}_{n,t}$ and r denote the loss cumulative loss of forecasting agent n and $\|\mathbf{W} - \mathbf{J}\|$ respectively.

Electrical & Computer ENGINEERING

Main Results: Sub-Linear Regret over a fixed Horizon

- if η is chosen as $\sqrt{\frac{8(1-r)\ln N}{t+2Nr}}$, then the regret bound becomes

$$\hat{L}_{n,t} \leq e^{\frac{2\sqrt{8\ln N}}{\sqrt{(t+2Nr)(1-r)}}} \left(\min_{n=1,\cdots,N} L_{n,t} + \frac{\sqrt{\ln N(t+2Nr)}}{\sqrt{8(1-r)}} + \frac{t\sqrt{(1-r)\ln N}}{\sqrt{8\,(t+2Nr)}} + \frac{2\sqrt{8\ln N}}{\sqrt{(t+2Nr)(1-r)}} \right). \quad (4)$$

.

- It is readily seen that when $r = 0$, the regret bound reduces to that of the one for the centralized case.
- $\lim_{t\to\infty} \frac{1}{t}\left(\hat{L}_{n,t} - L_{n,t}\right) = 0$ and that the regret bound is a function of network connectivity in terms of r.

Main Results: Sub-Linear Regret for any time

Theorem 2

For all $t \geq 1$, and for all sequence of outcomes $\{y_s\}_{s=1}^{t} \in \mathcal{Y}$ the regret of the distributed exponentially weighted average predictor with time-varying learning parameter $\eta_t = \sqrt{8 \ln N / t}$ satisfies

$$\hat{L}_{n,t} \leq e^{\frac{2Nr\sqrt{8 \ln N}}{1-r}} \left(\min_{n=1,\cdots,N} L_{n,t} + \sqrt{\frac{\ln N}{8}} + \sqrt{2t \ln N} + \frac{2Nr\sqrt{8 \ln N}}{1-r} \right). \tag{5}$$

Main Results: Sub-Linear Regret for any time

- As a consistency check, it can be verified that with $r = 0$, the regret bound reduces to the bound derived for the centralized case.
- Finally, it can also be seen that $\lim_{t\to\infty} \frac{1}{t}\left(\hat{L}_{n,t} - L_{n,t}\right) = 0$ which establishes that the regret is sub-linear with respect to the best performing expert.

Electrical & Computer ENGINEERING

Summary

Proposed a distributed sequence prediction algorithm based on the *consensus+innovations* framework.

Established sub-linear regret for the proposed expert-assisted learning algorithm.

Quantified the dependence on network connectivity.

Future work includes identify the set of best k experts in a constrained information setting.

Thank you!
Questions?