8. Online learning II

COMP0078: Supervised Learning

Mark Herbster

28 November 2022

University College London
Department of Computer Science
PartIIOnlinelearning22v-draft

Today

- Online Learning with Partial Feedback (Bandits)
- Online Multitask Learning with Long-Term Memory

Part I
Learning with Partial Feedback

Recall Allocation Setting (HEDGE)

Full Feedback Protocol

For $t = 1$ To m Do

Predict $\hat{y}_t \in [n]$

Observe full loss vector $\ell_t \in [0,1]^n$

Goal: Design master algorithms with "small regret".

$$\sum_{t=1}^{m} \ell_t, \hat{y}_t - \sum_{t=1}^{m} \min_{i \in [n]} \ell_{t,i} \leq o(m)$$

In the bandit setting we see only feedback for our prediction/action.

Partial Feedback Protocol

For $t = 1$ To m Do

Predict $\hat{y}_t \in [n]$

Observe loss of prediction $\ell_{t,\hat{y}_t} \in [0,1]$

Goal: as above.

Note: Although (largely) unobserved $\ell_1, \ldots, \ell_m$ are assumed to exist.

Partial Feedback Setting – Examples

1. **[Online Advertising]:** We have n potential ads to display. If the user click through on the ad we incur 0 loss.
2. **[Medical Trials]:** We have n potential medical treatments. The better the response the less loss.
3. **[Game tree search]:** We have n potential branches to search, if our "evaluation function" increases we proportionally receive less loss.

Exploration and Exploitation

This *Partial Feedback Setting* is often called the *Bandit* setting. The etymology being that we have a slot/fruit machine (once called one-armed bandits) each with potentially different "payback" rates and we wish to play so as to minimise our loss. Metaphorically, we will think of each prediction/action as pulling one of n arms.

Intuitions (Exploration and Exploitation)

1. We need to *use trials* to explore by trying different arms to estimate the machine with the minimal expected loss.
2. We need to *use trials* to exploit playing the arm with minimal observed loss.
3. The tradeoff: the *number of trials* used to explore limits exploitation and vice versa.

Before considering our "solution" let's discuss a key tool Unbiased Estimators.

Unbiased Estimators

Definition

An **estimator** $\hat{\theta}$ estimates a parameter θ of a distribution from a sample. An estimator is **unbiased** iff $E[\hat{\theta}] = \theta$.

Example 1 (Sample Mean)

Suppose $X_1, \ldots, X_n$ are IID random variables from a distribution with mean μ then $\hat{\theta} := \frac{1}{n}(X_1 + \ldots + X_n)$ is an unbiased estimator of μ.

Example 2 (German tank problem)

Suppose X is a random variable with the discrete uniform distribution over $\{1, \ldots, n\}$. Suppose n is unknown and we wish to estimate it The estimator $\hat{\theta}_1 := X$ is the maximum likelihood estimator, since $\mathcal{L}(\theta; X = x) = \frac{1}{\theta}[\theta \geq x]$ which is maximised when $\theta = x$. However for $\hat{\theta}_1$ we have

$$E[\hat{\theta}_1; \theta = n] = \sum_{x=1}^{n} \frac{1}{n} x = \frac{n+1}{2}$$

However $\hat{\theta}_2 := 2X - 1$ is the unbiased estimator, i.e.,

$$E[\hat{\theta}_2; \theta = n] = \sum_{x=1}^{n} \frac{1}{n}(2x - 1) = 2\sum_{x=1}^{n} \frac{1}{n} x - 1 \sum_{x=1}^{n} \frac{1}{n} = n + 1 - 1 = n$$

Assumptions and Estimation

Suppose we have a distribution $\mathcal{D}_i$ over $[0, 1]$ for each of $i \in [n]$ arms then each time t we "play" i we receive an IID sample $\ell_{t,i}$ from $\mathcal{D}_i$. Suppose we play i on trials $S_{t,i} \subseteq [t]$ then $\hat{\mu}_{t,i} := \sum_{t \in S_i} \frac{\ell_{t,i}}{|S_{t,i}|}$ is an unbiased estimator of μ_i.

As we get more samples from arm i the law of large numbers implies $\hat{\mu}_i \to \mu_i$ and concentration inequalities (e.g., Hoeffding) allow one to quantatively estimate the likelihood that the estimate differs significantly from the parameter. Using these observations the algorithm UCB balances exploration versus exploitation to obtain good regret bounds for this model. However we would like to consider a more general *adversarial* model. For example suppose $\mathcal{D}_i$ is changing over time (now $\mathcal{D}_{t,i}$) then the estimate $\hat{\mu}_{t,i}$ is biased (now $\hat{\mu}_{t,i} := \frac{\sum_{j=1}^{t} E[\ell_{j,i}]}{t}$) unless $S_i = [t]$. However if $S_{t,i} = [t]$ then we have no information on the other "arms."

Needed: a method of obtaining a **simultaneous** unbiased estimate for **all** arms!

Importance Weighting – 1

Suppose X is random variable over $\Re$ with a mean μ. By definition $E[X] = \mu$ and $\hat{\theta}_1 = X$ is an unbiased estimator of the mean. Define the biased coin Z_p with outcome H with probability p and T with probability $(1 - p)$ then define estimator $\hat{\theta}_p$ as equal to X/p if $Z_p = H$ as equal to 0 if $Z_p = T$. Now observe that,

$$E[\hat{\theta}_p] = \mathbb{P}(Z_p = H)(X/p) + \mathbb{P}(Z_p = T)0 = (p)X/p + (1 - p)0 = X$$

and is thus unbiased.

Importance Weighting – 2

We now generalise to obtain an unbiased estimator of ℓ_t in the bandit setting. Given $\mathbf{v}_t \in \Delta_n$ by the notation $\hat{y}_t \sim \mathbf{v}_t$ we mean sample $\hat{y}_t$ from a discrete distribution over $[n]$ where $\mathbb{P}(i) := v_{t,i}$.

Definition (Hallucinated Loss Vector)

We define the unbiased estimator ℓ_t^h of ℓ_t with respect to $\mathbf{v}_t$ as

$$(\ell_{t,i}^h := \frac{\ell_{t,i}}{v_{t,i}}[i = \hat{y}_t])_{i \in [n]},$$

where we have sampled $\hat{y}_t \sim \mathbf{v}_t$.

I.e, ℓ_t^h looks like, $\ell_t^h := (0, 0, \ldots, \frac{\ell_{t,\hat{y}_t}}{v_{t,\hat{y}_t}}, 0, \ldots, 0)$. Observe that ℓ_t^h is unbiased as for all $i \in [n]$ we have that $E[\ell_{t,i}^h] = \ell_{t,i}$, since

$$E_{\hat{y}_t \sim \mathbf{v}_t}[\ell_{t,i}^h] = \sum_{j=1}^{n} v_{t,j} \frac{\ell_{t,i}}{v_{t,i}}[i = j] = \ell_{t,i}$$

- Amazingly! we have an unbiased estimator for all arms by only observing a single arm.

The Next Steps

Idea

By sampling a single arm we can obtain an unbiased estimate for ℓ_t. Since we already have an algorithm for the full information setting for arbitrary loss functions (HEDGE) our proposed algorithm is to simply apply HEDGE to the hallucinated loss vectors.

Todo

1. **Problem:** ℓ_t^h is potentially unbounded and HEDGE requires bounded loss vectors. **Fix:** Use a more careful analysis of HEDGE.
2. **Problem:** We will be giving an *expected* regret bound and there will be some subtleties in the sources of randomness. **Fix:** we will clarify the adversarial model that generates $\ell_1, \ldots, \ell_m$.

EXP3 Exponential-weight algorithm for Exploration and Exploitation [ACFS02]

Exp3

Parameter $\eta \in (0, \infty)$

Set $\mathbf{v}_1 = (\frac{1}{n}, \ldots, \frac{1}{n})$

For $t = 1$ To m do

Sample $\hat{y}_t \sim \mathbf{v}_t$ % i.e., treat $\boldsymbol{v}_t$ as a distribution over $[n]$

Observe loss $\ell_{t,\hat{y}_t} \in [0, 1]$

Construct hallucinated loss vector

$$\boldsymbol{\ell}_t^h := (\ell_{t,i}^h := \frac{\ell_{t,i}}{v_{t,i}}[i = \hat{y}_t])_{i \in [n]}$$

Update

$$v_{t+1,i} = v_{t,i} \exp(-\eta \ell_{t,i}^h)/Z_t \text{ for } i \in [n]$$

where $Z_t = \sum_{i=1}^n v_{t,i} \exp(-\eta \ell_{t,i}^h)$

Footnote: the original EXP3 algorithm was slightly different but it incorporated the key idea of using HEDGE with *hallucinated* loss vectors.

EXP3 – Initial Analysis

Lemma

For any sequence of loss vectors

$$\ell_1, \ldots, \ell_m \in [0,1]^n$$

we have the following inequality for EXP3,

$$\sum_{t=1}^{m} \mathbf{v}_t \cdot \ell_t^h - \sum_{t=1}^{m} \mathbf{u} \cdot \ell_t^h \leq \frac{\ln n}{\eta} + \frac{\eta}{2} \sum_{t=1}^{m} \sum_{i=1}^{n} v_{t,i} (\ell_{t,i}^h)^2 \text{ for all } \mathbf{u} \in \Delta_n \,. \tag{1}$$

Proof. The Lemma follows the fact EXP3 is "just" HEDGE with ℓ_t mapped to ℓ_t^h and we proved the above inequality for HEDGE on slide "HEDGE Theorem - Proof (2)."

To finish our analysis:

1. We need to perform expectations so we may replace hallucinated losses ℓ_t^h with the true losses ℓ_t.
2. In order perform expectations we need understand the sources of randomness in our model, in particular we need a model for how the "adversary" generates $\ell_1, \ldots, \ell_m$
3. Finally we need to bound the term $\sum_{t=1}^{m} \sum_{i=1}^{n} v_{t,i} (\ell_{t,i}^h)^2$ and tune η.

Model : Deterministic Oblivious Adversary

Motivation: Our predictions/actions may influence the environment.

Suppose we are using a bandit algorithm to set prices for our products. This may influence the way competitors set prices. If we have an app to display curated news to a user the prior display of new items may effect the user's choices for the next news items. More generally the learners actions may influence the adversary/nature in the future.

Deterministic Oblivious Adversary Model

In this model simply put

$$\ell_1, \ldots, \ell_m$$

are determined before the run the algorithm.

However, the process/adversary setting them is assumed to have complete prior knowledge of the learner's algorithm and may set the loss vectors using this knowledge, i.e., if the adversary knows the learner will do (or is likely to) predict something on trial t it may set ℓ_t accordingly. The limitation of this near-omniscient adversary is that it is non-adaptive i.e., if the learner will be taking random actions although the adversary may take this into account in setting $\ell_1, \ldots, \ell_m$ it cannot however change them "on-the-fly." Observe that this adversary may simulate the *stochastic* model, by simple repeatedly sampling $\mathcal{D}_1, \ldots, \mathcal{D}_m$ in advance.

EXP3 - Theorem

Notation: $L_A(S) := \sum_{t=1}^{m} \ell_{t,\hat{y}_t}$ and $L_i(S) := \sum_{t=1}^{m} \ell_{t,i}$

Theorem EXP3 **(Bound)** **[ACFS02]**

For any sequence of loss vectors

$$S = \ell_1, \ldots, \ell_m \in [0,1]^n$$

the regret of EXP3 with $\eta = \sqrt{2 \ln n / mn}$ is

$$E[L_A(S)] - \min_i L_i \leq \sqrt{2mn \ln n}\,.$$

Comparing regrets : HEDGE: $\sqrt{2m \ln n}$ and EXP3: $\sqrt{2mn \ln n}$.

EXP3 Theorem - Proof (1)

Proof – 1

Observe that the only sources of randomness are the samples $\hat{y}_t \sim \mathbf{v}_t$. As previously argued note that $E[\ell^h_{t,i}] = \ell_{t,i}$ and thus

$$E[\mathbf{v}_t \cdot \ell^h_t] = \sum_{i=1}^{n} E[v_{t,i}\ell^h_{t,i}] = \sum_{i=1}^{n} v_{t,i} E[\ell^h_{t,i}] = \sum_{i=1}^{n} v_{t,i}\ell_{t,i} = E[\ell_{t,\hat{y}_t}] \tag{2}$$

and we also have,

$$E[(\ell^h_{t,i})^2] = \sum_{j=1}^{n} v_{t,j} \left(\frac{\ell_{t,i}}{v_{t,i}}\right)^2 [j = i]^2 = v_{t,i} \left(\frac{\ell_{t,i}}{v_{t,i}}\right)^2 = \frac{\ell^2_{t,i}}{v_{t,i}} \tag{3}$$

which implies

$$E\left[\sum_{i=1}^{n} v_{t,i}(\ell^h_{t,i})^2\right] = \sum_{i=1}^{n} v_{t,i}\frac{\ell^2_{t,i}}{v_{t,i}} = \sum_{i=1}^{n} \ell^2_{t,i} \leq n\,. \tag{4}$$

Proof – 2

Recalling (1) and taking expectations,

$$E\left[\sum_{t=1}^{m}\mathbf{v}_t\cdot\ell_t^h-\sum_{t=1}^{m}\mathbf{u}\cdot\ell_t^h\right]\leq E\left[\frac{\ln n}{\eta}+\frac{\eta}{2}\sum_{t=1}^{m}\sum_{i=1}^{n}v_{t,i}(\ell_{t,i}^h)^2\right](\forall\mathbf{u}\in\Delta_n)$$

thus,

$$E\left[\sum_{t=1}^{m}\mathbf{v}_t\cdot\ell_t^h\right]-\min_i E\left[\sum_{t=1}^{m}\ell_{t,i}^h\right]\leq\frac{\ln n}{\eta}+\frac{\eta}{2}E\left[\sum_{t=1}^{m}\sum_{i=1}^{n}v_{t,i}(\ell_{t,i}^h)^2\right]$$

applying (2) to the first term, lower bounding the second term by observing that $\mathbf{u}$ can be any coordinate vector and that $E[\ell_{t,i}^h]=\ell_{t,i}$ then using (4) for the final term gives

$$E\left[L_A(S)\right]-\min_i L_i(S)\leq\frac{\ln n}{\eta}+\frac{\eta}{2}mn$$

now substituting $\eta=\sqrt{2\ln n/mn}$ proves the theorem. □

Part II
Online Multitask Learning with Long-Term Memory

Eternal Return

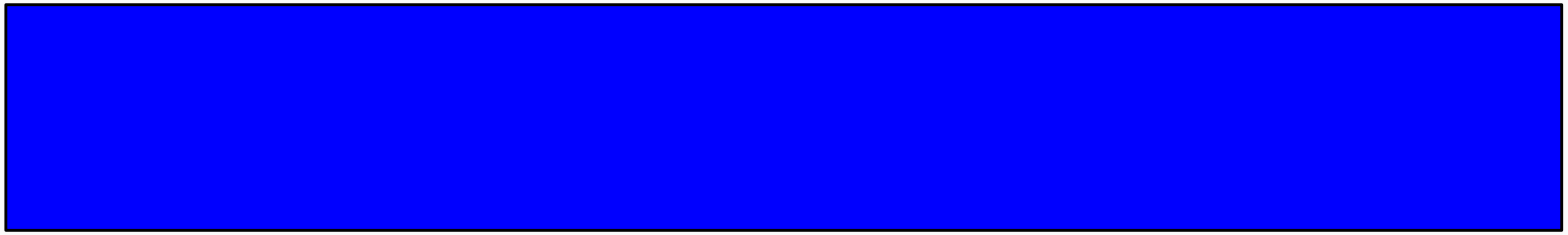
What has been will be again,
what has been done will be done again;
there is nothing new under the sun.

– **Ecclesiastes 1:9**

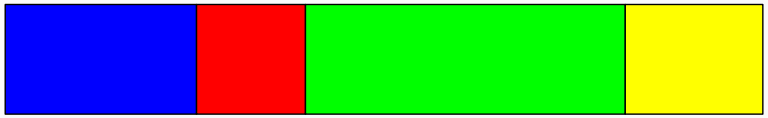
Must not what ever can already have passed this way before? Must not what ever can happen, already have happened, been done, passed by before? And if everything has already been here before, what do you think of this moment, dwarf? Must this gateway too not already – have been here? And are not all things firmly knotted together in such a way that this moment draws after it all things to come? Therefore – itself as well? For, whatever can run, even in this long lane outward-must run it once more! – And this slow spider that creeps in the moonlight, and this moonlight itself, and I and you in the gateway whispering together, whispering of eternal things – must not all of us have been here before? – And return and run in that other lane, outward, before us, in this long, eerie lane – must we not return eternally?

– **Friedrich Nietzsche, Thus Spoke Zarathustra**

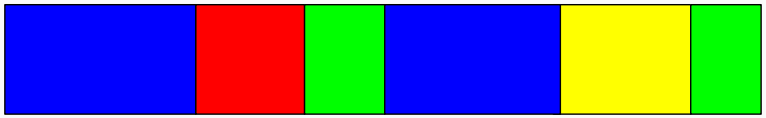
The model



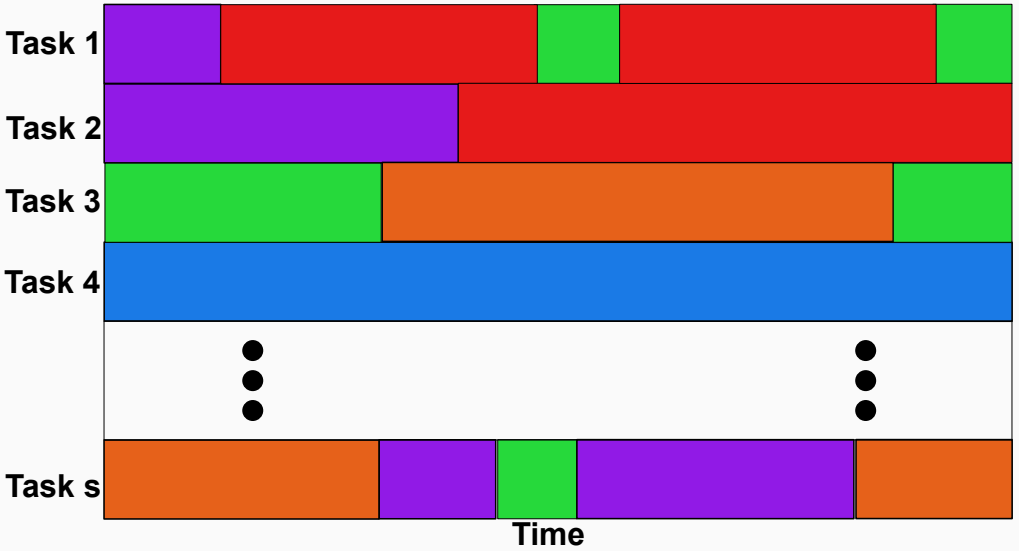
One Mode ("hypothesis") – One Task



Multiple Modes – One Task



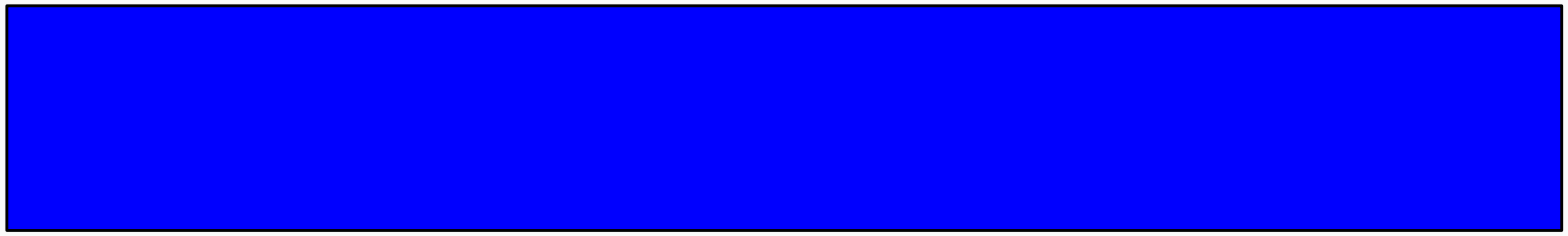
Multiple Repeated Modes – One Task



Multiple Modes – Multiple Tasks

Outline

- Single Task Model
  - Finite Hypotheses Classes (“Experts”) : $\mathcal{H}_n$
  - Linear Interpolants with Kernels (RKHS) : $\mathring{\mathcal{H}}_K$
- Single Task with Switching
- The Share algorithm
  - Switches and Modes
  - Inefficient Algorithms (exponential time)
- Multitask Model
  - Model
  - Algorithm and Bounds (Finite Hypotheses Classes) % Not our focus in this lecture.
  - Algorithm and Bounds (RKHS)
- Online Switching Multitask Learning in an RKHS
  - On-Line (Inductive) Binary Matrix Completion
  - Complexity Measures for Matrices
  - MEG-IMCSI Algorithm and Bound
  - Latent Block Models
  - Side-information
  - Bounding the Quasi-Dimension
  - Reducing Online Switching Multitask Learning to Matrix Completion with Side-information
  - A Simplified Online Multitask Model



One Mode – One Task

On-Line Learning Model

Protocol

For $t = 1$ to T do

Receive pattern	$x_t \in \mathcal{X}$
Predict/Act	$\hat{y}_t \in \mathcal{Y}$
Receive label	$y_t \in \mathcal{Y}$
Incur loss	$L_A = L_A + \ell(y_t, \hat{y}_t)$

Evaluation metric: The loss of an Algorithm A on $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$

$$L_A := \sum_{t=1}^{T} \ell(y_t, \hat{y}_t)$$

Our Goal: Design master algorithms with "small loss".

How is this possible without distributional assumptions?

Regret Bounds

Idea

We do not give an absolute bound but rather a relative bound to a hypothesis class $\mathcal{H}$ of predictors.

Cumulative loss definitions

$$L_A := \sum_{t=1}^{T} \ell(y_t, \hat{y}_t) \qquad L_h := \sum_{t=1}^{T} \ell(y_t, h(\mathbf{x}_t))$$

Non-uniform regret bound

$$L_A - L_h \leq \text{regret}(\mathcal{H}, h, T) \qquad \forall h \in \mathcal{H}$$

Example Hypotheses Classes

1. $\mathcal{H}_n := \{h^1, \ldots, h^n\} \subseteq \{-1, 1\}^{\mathcal{X}}$ "Finite"
2. $\mathcal{H}_K := \overline{\text{span}}(\{K(x, \cdot)\}_{\forall x \in \mathcal{X}})$ $\mathcal{H}_K$ is RKHS def. $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

$0-1$ Loss Regret Bounds

- Regret Bounds HEDGE/WM/AGGA and Online Gradient Descent ("OGD").
- $\mathcal{L}_{01}(y, \hat{y}) := [y \neq \hat{y}]$, "mistake counting"

Theorem "Hedge/WM/AggA" (Bound) [V90,LW94,FS97]

For any finite hypothesis class $\mathcal{H}_n \subset \{-1,1\}^{\mathcal{X}}$ and any $(x_1, y_1), \ldots, (x_T, y_T)$

$$\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \mathcal{L}_{01}(y_t, h(x_t)) \leq \sqrt{2 \log(n) T} \qquad (\forall h \in \mathcal{H}_n)\,. \tag{5}$$

Theorem "OGD" (Bound) [CLW96,S11]

For any kernel K and any $(x_1, y_1), \ldots, (x_T, y_T)$

$$\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \mathcal{L}_{01}(y_t, h(x_t)) \leq \frac{1}{2}\sqrt{\|h\|_K^2 X^2 T} \qquad (\forall h \in \mathring{\mathcal{H}}_K) \tag{6}$$

with $X^2 := \max_{t \in [T]} K(x_t, x_t)$ and interpolating hypotheses

$$\mathring{\mathcal{H}}_K := \{h \in \mathcal{H}_K : h(x_t) \in \{-1,1\}\,, \forall t \in [T]\}\,.$$

- Bounds depend on tuning of parameters for algorithms.
- **Note:** difference between linear *classification* and *interpolation*.
- No known eff. algs. achieve regret bounds for linear classification.

Hedge/WM/AggA

Parameters: Learning rate $\gamma \in [0, \infty)$; finite hypothesis set $\{h^1, \ldots, h^n\} = \mathcal{H}_n \subset \{-1, 1\}^{\mathcal{X}}$

Initialization: Initialize $\boldsymbol{v}_1 = \frac{1}{n}\mathbf{1}$

For $t = 1, \ldots, T$

- Receive instance $\boldsymbol{x}_t \in \mathcal{X}$.
- Set $\mathbf{h}_t = (h^1(\boldsymbol{x}_t) \ldots h^n(\boldsymbol{x}_t)) \in \{-1, 1\}^n$.
- Predict

$$i_t \sim v_t \,;\; \hat{y}_t \leftarrow h_t^{i_t} \,.$$

- Receive label $y_t \in \{-1, 1\}$.
- Update

$$\ell_t \leftarrow \frac{1}{2}|\mathbf{h}_t - \hat{y}_t\mathbf{1}|$$

$$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t \odot \exp\left(-\gamma \ell_t\right)$$

$$\boldsymbol{v}_{t+1} \leftarrow \frac{\boldsymbol{w}_{t+1}}{\sum_{i=1}^n w_{t+1,i}}$$

Algorithm: HEDGE/WM

Randomised Kernel OGD

Parameters: Learning rate $\gamma \in [0, \infty)$, SPD Kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

Initialization: Initialize $\boldsymbol{w}_1 = \mathbf{0}$

For $t = 1, \ldots, T$

- Receive $x_t \in \mathcal{X}$.
- Predict

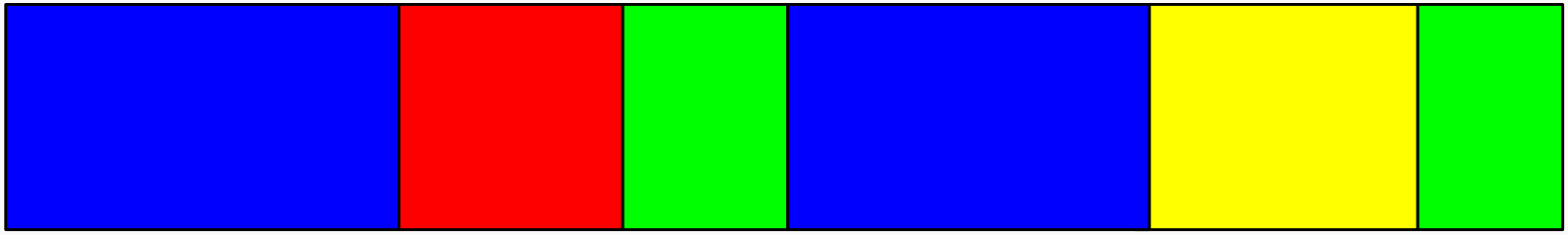
$$Y_t \sim \textsc{Uniform}(-1, 1)\,;\ \bar{y}_t \leftarrow \boldsymbol{w}_t(x_t);\ \hat{y}_t \leftarrow \text{sign}(\bar{y}_t - Y_t)\,.$$

- Receive label $y_t \in \{-1, 1\}$.
- If $\bar{y}_t y_t \leq 1$ then

$$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \gamma y_t K(x_t, \cdot)$$

- Else $\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t$

Algorithm: Randomized Online Kernel Gradient Descent (OGD)



Switching – Single Task

Share Algorithm

Learning with sequences of experts

On-line Learning (Review)

time t	1	2	3	4	$\cdots$	t
expert 1	.5	3	2	1	$\cdots$	$x_{t,1}$
expert 2	.75	-1.5	-1	-2	$\cdots$	$x_{t,2}$
expert 3	-1.5	3	2	-4	$\cdots$	$x_{t,3}$
expert 4	.75	-1.3	1.5	1.5	$\cdots$	$x_{t,4}$
alg. preds	0	1.5	1.5	.75	$\cdots$	$\hat{y}_t$
label	.75	-1.5	2	1	$\cdots$	y_t
alg. loss	0.56	9	.25	.06	$\cdots$	$(\hat{y}_t - y_t)^2$

For a sequence of examples $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}$

$$\text{Loss}_A(S) = \sum_{t=1}^{T} (\hat{y}_t - y_t)^2$$

Aim: to bound $\text{Loss}_A(S)$ in terms of the loss of the best expert.

$$\text{Loss}_i(S) = \sum_{t=1}^{T} (x_{t,i} - y_t)^2$$

Static Relative Loss Bounds

For all sequences of examples S

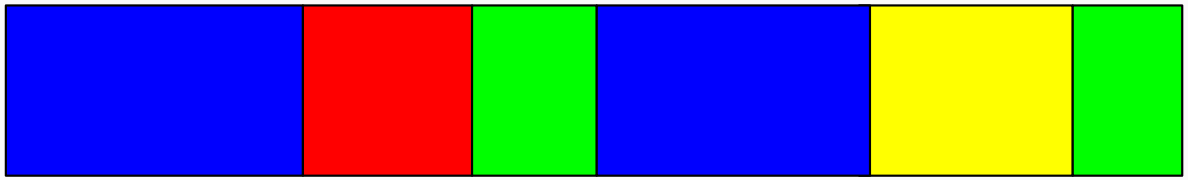
$$\text{Loss}_A(S) \leq \text{Loss}_i(S) + c \ln n \quad [V90, LW94, HKW98]$$

- The loss of the algorithm is bounded in terms of the best single expert.

So far the comparator is "static"

Recall: 1) That c depends on the loss function for example log ($c = 1$) and square ($c = 2$). In the allocation model the results will still apply but we will gain an additional term that depend $\Theta(\sqrt{\log nT})$.

Switching Model



Sequence of hypotheses (switches $k(h^*) = 5$, modes $m(h^*) = 4$)

Let $h^* := (h_1, h_2, \ldots, h_T)$ denote a sequence of hypotheses.

Loss for a sequence of hypotheses

$$\mathsf{L}_{h^*} = \sum_{t=1}^{T} \ell(y_t, h_t(x_t))$$

Measures of complexity for a sequence of hypotheses

$$\underbrace{k(h^*) := \sum_{t=1}^{T} [h_t \neq h_{t+1}]}_{\text{switches}} \quad ; \quad \underbrace{m(h^*) := \bigcup_{t=1}^{T} \{h_t\}}_{\text{modes (distinct hypotheses)}}$$

Note: Switch times and constituent hypotheses are **unknown** to the learner.

Multiple Modes

Abbreviation: Set $k := \text{size}(\{h_1, h_2, \ldots, h_T\})$.

Proposition

The FIXED-SHARE algorithm achieves,

1. $\mathcal{X} = [0,1]^n$
2. $\mathcal{Y} = [0,1]$
3. $\ell(y_t, \hat{y}_t) = y \log \frac{y}{\hat{y}} + (1-y) \log \frac{1-y}{1-\hat{y}}$
4. $\mathcal{H}_e := \{h_i : i = 1, \ldots, n\}$ with $h_i(\mathbf{x}) := x_i$
5.

$$L_A - L_{\{h_1, h_2, \ldots, h_T\}} \leq k \log \frac{T-1}{k} + k \log(n-1) + \log n + k$$

$$\forall \{h_1, h_2, \ldots, h_T\} \subset \mathcal{H}_e$$

Asymptotically we pay $\log \frac{T}{k} + \ln(n-1)$ per mode.

Fixed Share Algorithm

- **Parameters:** $0 \leq \eta$ and $0 \leq \alpha \leq 1$.
- **Initialization:** Set the weights to $w^s_{1,1} = \ldots = w^s_{1,n} = \frac{1}{n}$.
- **Prediction:** Let $v_{t,i} = \frac{w^s_{t,i}}{W_t}$, where $W_t = \sum_{i=1}^{n} w^s_{t,i}$. Predict with $\hat{y}_t = \mathbf{v}_t \cdot \mathbf{x}_t$
- **Loss Update:** After receiving the tth outcome y_t,

$$\forall i : 1, \ldots, n : w^m_{t,i} = w^s_{t,i} e^{-\eta \ell(y_t, x_{t,i})}$$

- **Fixed Share Update:**
  - $\text{pool} = \sum_{i=1}^{n} \alpha w^m_{t,i}$
  - $\forall i : 1, \ldots, n : w^s_{t+1,i} = (1-\alpha) w^m_{t,i} + \frac{1}{n-1} w^s_{t,i} (\text{pool} - \alpha w^m_{t,i})$

Shifting Loss Bounds

Shifting Experts:

- Let

$$k := \text{size}(\{i_1, i_2, \ldots, i_T\}) := \sum_{k=1}^{T-1} [i_k \neq i_{k+1}]$$

- $L^* := \text{Loss}_{\{i_1, i_2, \ldots, i_T\}}(S)$
- Choose $\alpha \in [0, 1]$,

We then have the bound:

$$L_A(S) \leq L^* + c[(T-1)(H(\alpha^*) + D(\alpha^* \| \alpha)) + k \ln(n-1) + \ln n],$$

where $\alpha^* = \frac{k}{T-1}$, $H(p) = \frac{1}{p} \ln \frac{1}{p} + \frac{1}{1-p} \ln \frac{1}{1-p}$ and
$D(p^* \| p) = p^* \ln \frac{p^*}{p} + (1-p^*) \ln \frac{1-p^*}{1-p}$

When $\alpha = \frac{k}{T-1}$, then the above is upper bounded by

$$L_A(S) \leq L^* + c[k \ln \frac{T-1}{k} + k \ln(n-1) + \ln n + k],$$

Shifting Loss Bounds (Proof Sketch – 1)

Proof Sketch – 1

1. Observe that the bound $\text{Loss}_A(S) \leq \text{Loss}_i(S) + c \ln n$ for the expert setting trivially generalises to

$$\text{Loss}_A(S) \leq \text{Loss}_i(S) + c \ln \frac{1}{w_{1,i}} \quad (i \in [n])$$

   i.e., rather than assign all experts uniform probability initially we can choose an arbitrary prior weights (probability).

2. Consider the following set of n^T initial weights:

$$w_{1,\{i_1,i_2,\ldots,i_T\}} := \frac{1}{n}(1-\alpha)^{T-1-k}\left(\frac{\alpha}{n-1}\right)^k \quad (\{i_1, i_2, \ldots, i_T\} \in [n]^T)$$

3. Observe that the sum of these weights is 1.
4. These are just “meta-experts” that describe all possible expert sequences over T trials.

Shifting Loss Bounds (Proof Sketch – 2)

Proof Sketch – 2

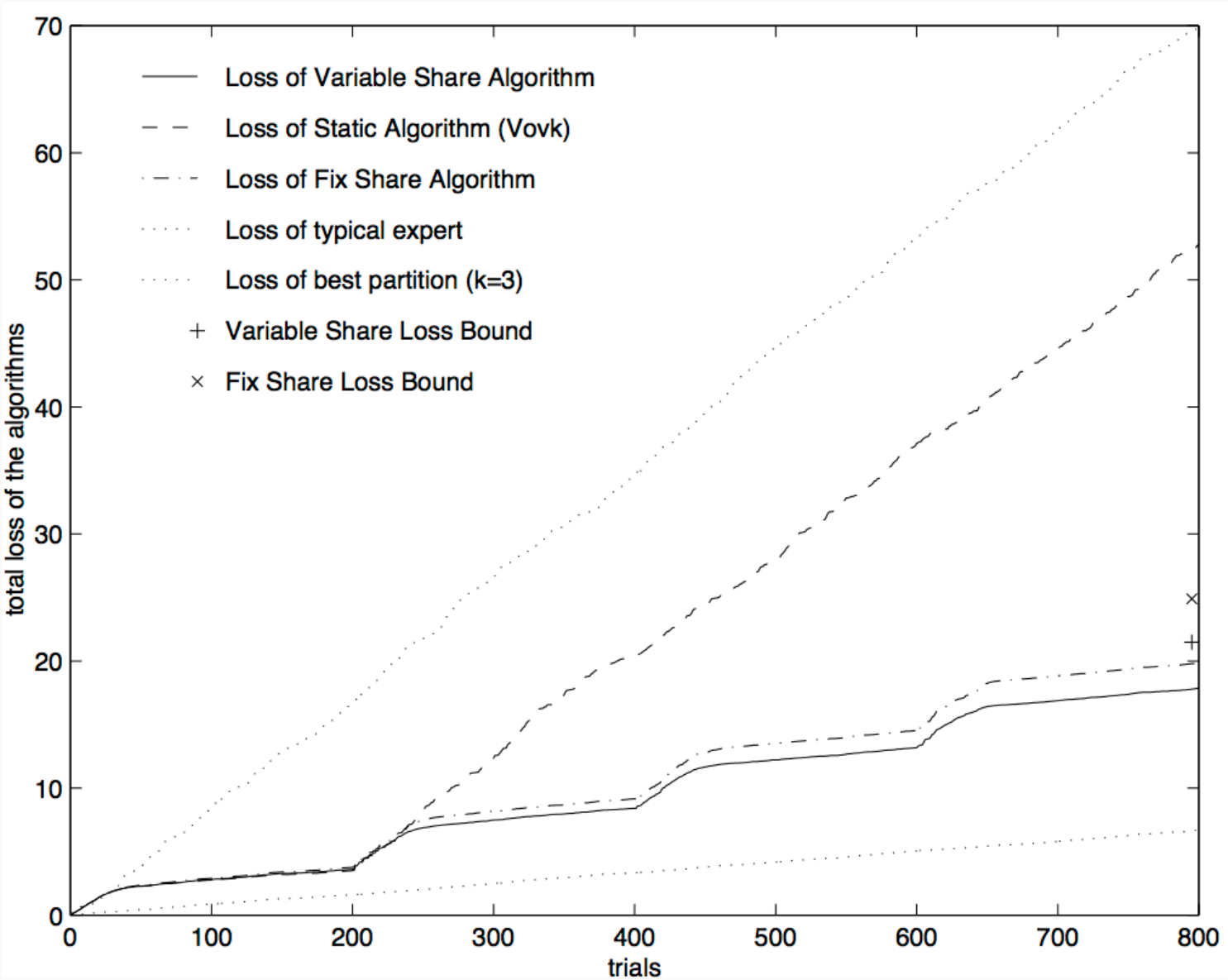
1. Observe if we apply the WA algorithm with these meta-experts then the previous bound follows (with some algebra) by just simplifying $\ln \frac{1}{w_{1,\{i_1,i_2,\ldots,i_T\}}}$.
2. With a more detailed analysis one can then show FIXED-SHARE algorithm perfectly simulates the prediction of the algorithm with T^n meta-experts in $O(n)$ time.

□

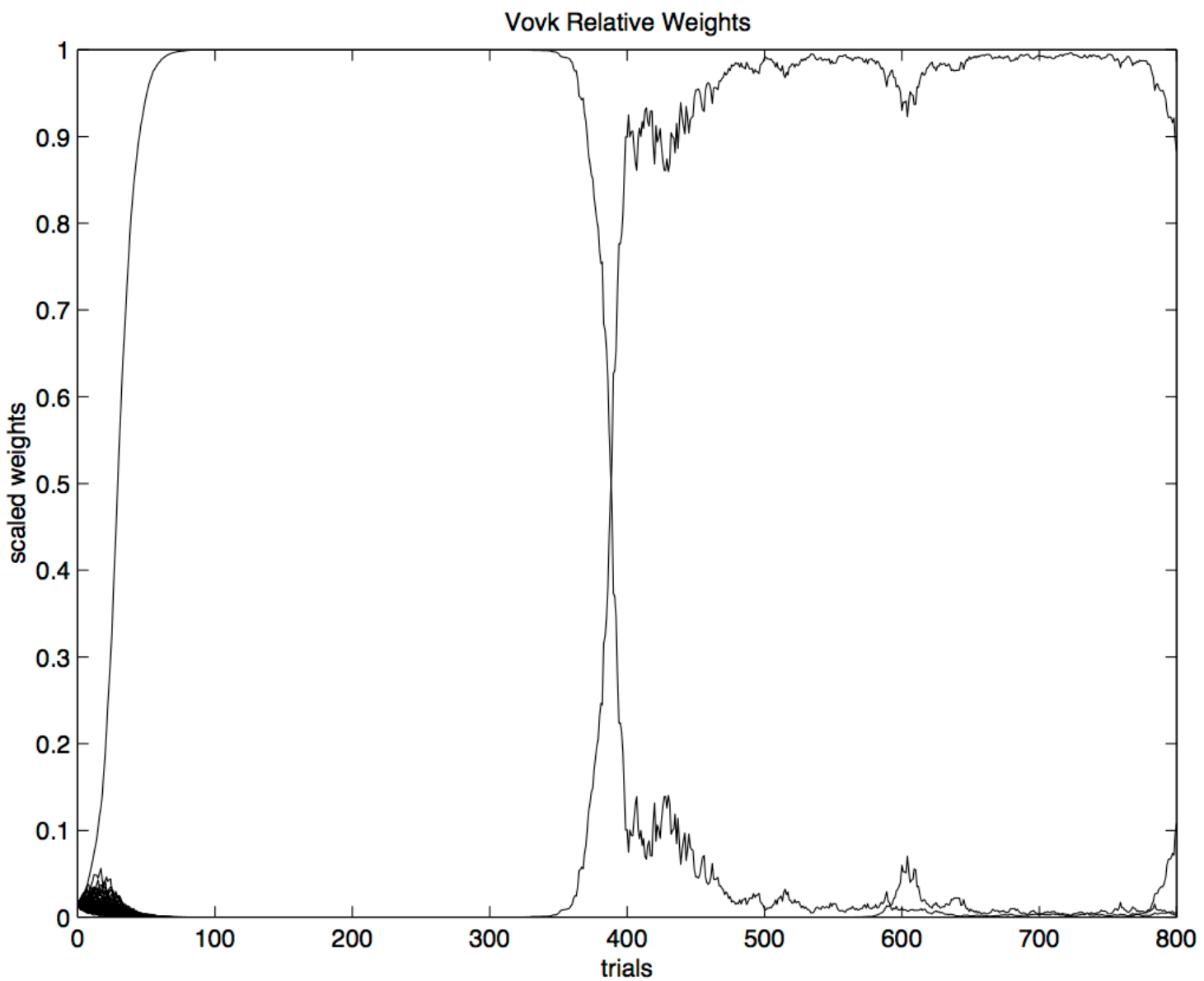
Experiment (simulation)

1. Trials (T) : 800
2. Shifts(k): 3
3. Number of Experts (n) : 64
4. Loss function (L): $L(y, \hat{y}) = (y - \hat{y})^2$ ($c = 1/2, \eta = 2$)
5. Share Parameter(α) : 0.024
6. Typical expert expected Loss/Trial : $1/12$
7. “Best” expert expected Loss/Trial : $1/120$

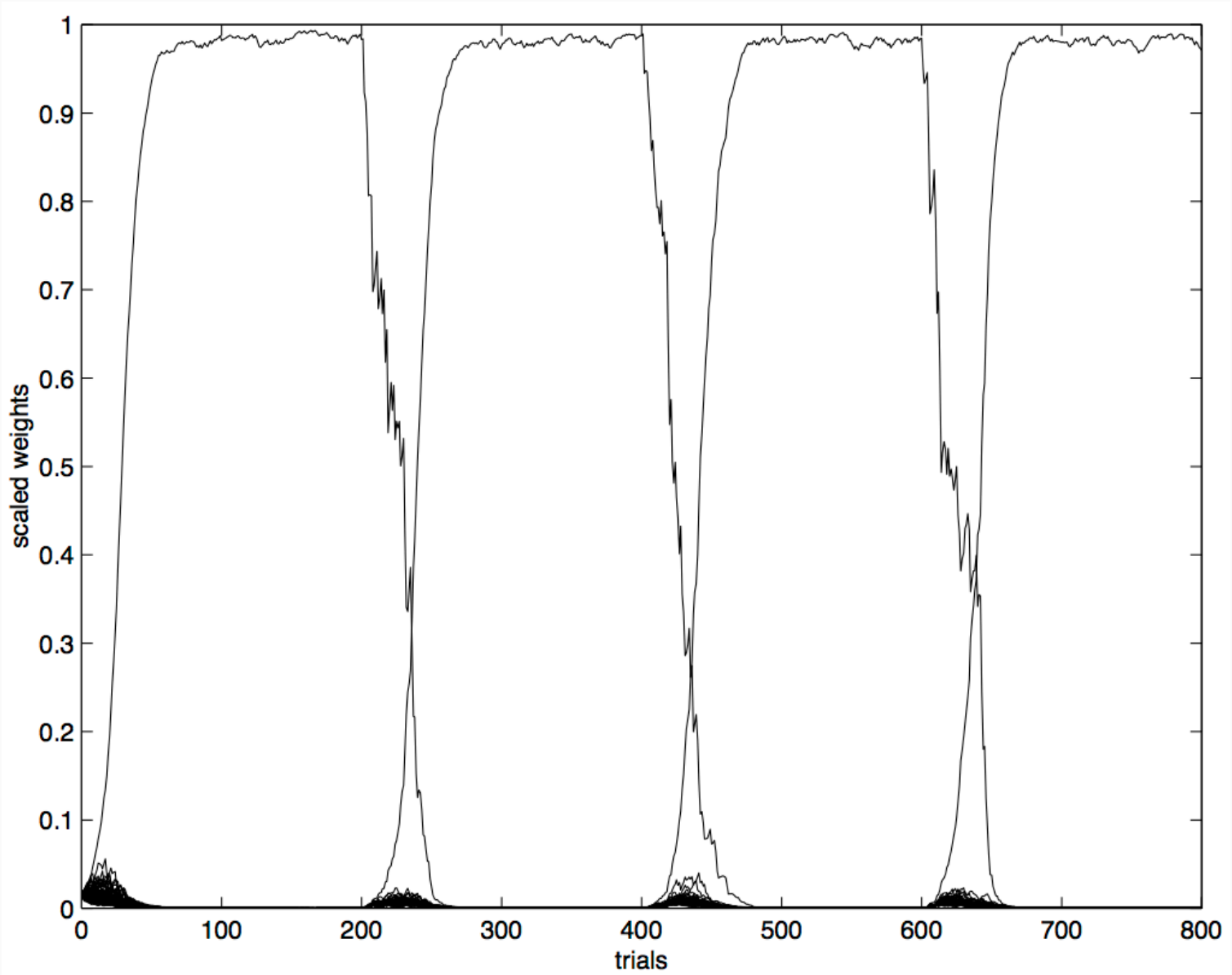
Share Algorithms – Performance



Expert Algorithm's Weights



Variable Share Weights



Some Applications

1. Predicting disk idle times
2. Online load balancing (process migration determination)
3. Predicting TCP packet inter-arrival times
4. Financial prediction by combining portfolios
5. Combining language for domain and topic adaptation

Introducing Memory

Learning a sequences of hypotheses paying less when a hypotheses “returns.”

Reasoning about upper bounds for switching for $\mathcal{H}_n$ (MEMORY)

1. Reduce switching regret via creating "meta"-hypotheses $\bar{h}$ from $\mathcal{H}_n$

$$\mathcal{H}_n^{T,k,m} := \{\bar{h} \in \mathcal{H}^T : k(\bar{h}) = k, |m(\bar{h})| = m\}$$

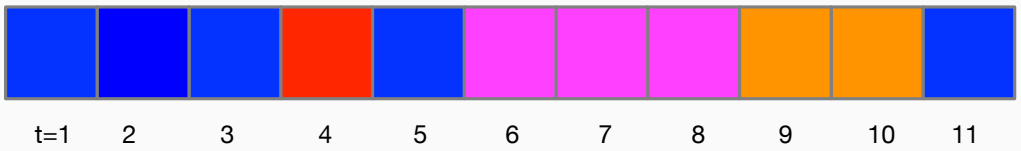
2. A meta-hypothesis "colors" each trial t with a hypothesis from $\mathcal{H}_n$.
3. Bounding $|\mathcal{H}_n^{T,k,m}| \leq \binom{n}{m}\binom{T-1}{k} m^{k+1}$;

$$\log |\mathcal{H}_n^{T,k,m}| \leq m \log \frac{n}{m} + k \log \frac{T-1}{k} + (k+1) \log m + k + m$$

4. Recalling the regret bound of HEDGE,

$$\mathbb{E}[L_{\text{HEDGE}}] - L_h \leq \sqrt{2T \log(n)} \qquad (\forall h \in \mathcal{H}_n)\,.$$

5. Bounds immediately follow but algorithms are exponential in k & m.
6. An $\mathcal{O}(n)$ algorithm for $\mathcal{H}_n^{T,k,m}$ was posed as an open problem in [F00] solved by [BW03] and improved in [KAW12].



Meta-hypotheses $\bar{h}$ with $k = 5$ and $m = 4$.

Reasoning about upper bounds for switching for $\mathcal{H}_K$ - 1

1. Previous approach fails since $|\mathcal{H}_K|$ is typically infinite.
2. Meta-hypotheses are now meta-"learners" $\hat{h}$.
3. A meta-learner $\hat{h}$ consists of m instances of OGD.
4. The "color" of a trial is the associated OGD instance.
5. An OGD instance only runs only its colored trials with state saved/restored between trials.

Start OGD-1	Cont. OGD-1	Cont. OGD-1	Start OGD-2	Cont. OGD-1	Start OGD-3	Cont. OGD-3	Cont. OGD-3	Start OGD-4	Cont. OGD-4	Cont. OGD-1
t=1	2	3	4	5	6	7	8	9	10	11

A meta-learner $\hat{h}$ with $k = 5$ and $m = 4$.

Reasoning about upper bounds for switching for $\mathcal{H}_K$ - 2

1. Thus the loss of a single meta-learner $\hat{h}$ is

$$L_{\hat{h}} = \sum_{i=1}^{m} L_{\text{OGD-}i}$$

OGD-i receives a subsequence from a partition of trial sequence (POTS),

$$(x_1(i), y_1(i)), \ldots, (x_{T_i}(i), y_{T_i}(i)) \text{ with } T = \sum_{i=1}^{m} T_i.$$

2. Using OGD bounding via (6)

$$R = \sum_{i=1}^{m} \sum_{t=1}^{T_i} \mathbb{E}[\mathcal{L}_{01}(y_t(i), \hat{y}_t(i))] - \mathcal{L}_{01}(y_t, h_i(x_t(i))) \leq \frac{1}{2} \sum_{i=1}^{m} \sqrt{\|h_i\|_K^2 X^2 T_i}$$

for a fixed POTS with any $h_1, \ldots, h_m \in \mathcal{H}_K^\circ$

3. Then using HEDGE to mix over all $\hat{h}$ bounding via (5)

$$R \leq \mathcal{O}\left(\sqrt{\left(\sum_{i=1}^{m} \|h_i\|_K^2 X^2 + k \log m + k \log \frac{T}{k}\right) T}\right)$$

for any POTS with any $h_1, \ldots, h_m \in \mathcal{H}_K^\circ$.

Considerations

For finite hypotheses classes we obtained,

$$R \leq \mathcal{O}\left(\sqrt{\left(m \log \frac{n}{m} + k \log m + k \log \frac{T}{k}\right) T}\right) \quad (\forall \bar{h} \in \mathcal{H}_n^{T,k,m}) \quad (7)$$

and for RKHS hypotheses classes we obtained,

$$R \leq \mathcal{O}\left(\sqrt{\left(\sum_{i=1}^{m} \|h_i\|_K^2 X^2 + k \log m + k \log \frac{T}{k}\right) T}\right) \quad (\forall \bar{h} \in \mathring{\mathcal{H}}_K^{T,k,m}) \quad (8)$$

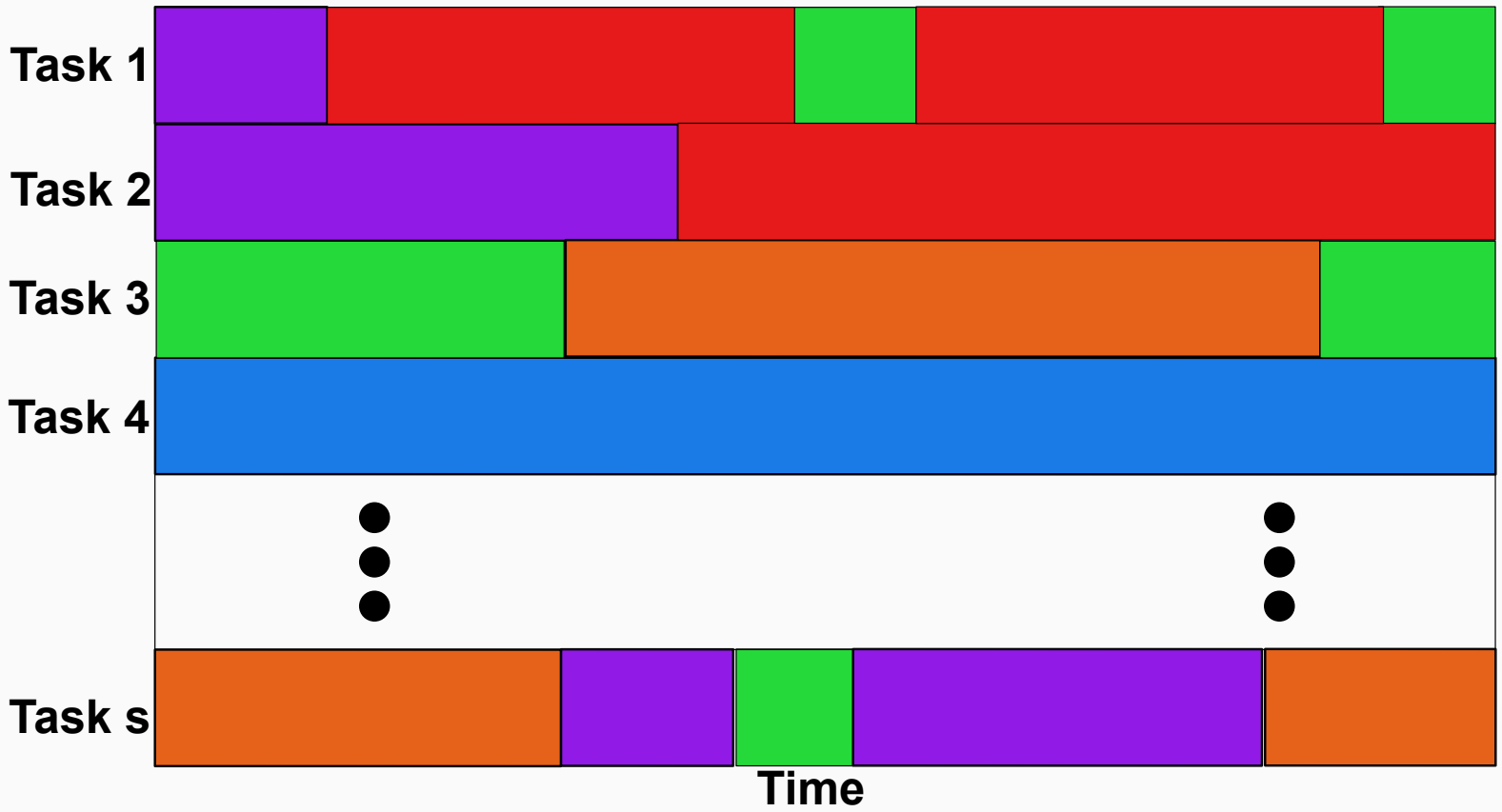
- We view the blue terms as “learner complexities” and the remaining terms as the price for their sequence locations.

GOAL: to obtain these bounds with *efficient* algorithms.

A Novel Model

Multiple Tasks

Multiple Tasks with Switching and Memory



Multiple Tasks with Switching and Memory

Multiple Task Learning Model

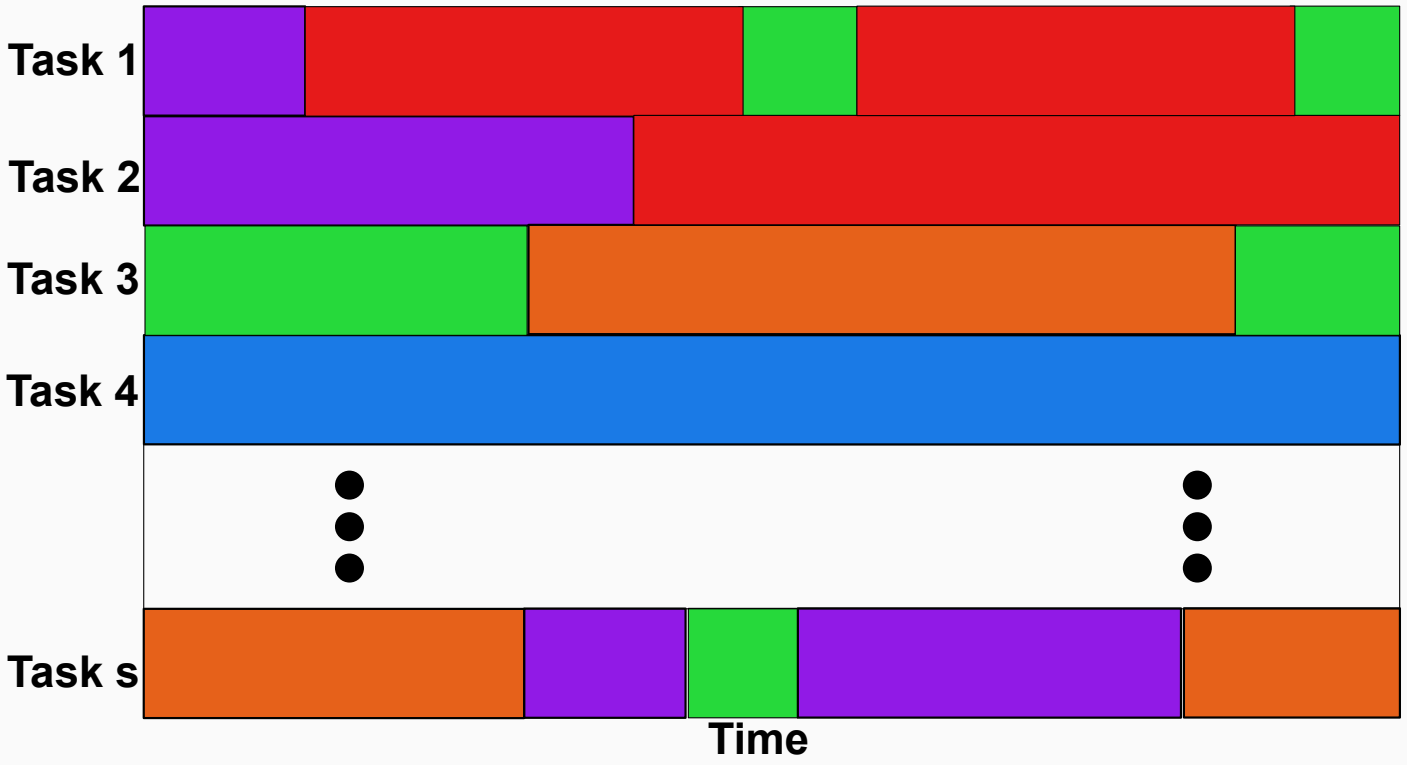
Protocol

For $\tau = 1$ to T do

Receive task $i \in [s]$; set t to "local time" in task i	
Receive pattern	$x_t^i \in \mathcal{X}$
Predict/Act	$\hat{y}_t^i \in \mathcal{Y}$
Receive label	$y_t^i \in \mathcal{Y}$
Incur loss	$L_A = L_A + \ell(y_t^i, \hat{y}_t^i)$

Notation: Global time is τ and local time is $\frac{i}{t}$ wrt task i.

Measures of Complexity



$$k(h^*) := \text{switches} = \sum_{i=1}^{s} \sum_{t=1}^{T-1} [h_t^i \neq h_{t+1}^i] \qquad m(h^*) := \text{modes} = \bigcup_{i,t}^{s,T} \{h_t^i\}$$

- Definitions are direct extension of single task case.
- Combinatorically tasks *reduce* to s additional switches.
- Algorithmically tasks *do not reduce* to switches.

Theorem (Finite Hypotheses Classes)

Theorem [HPT20b] (Finite Hypotheses Classes)

There exists an algorithm with parameters $\mathcal{H}_n \subseteq \{-1, 1\}^{\mathcal{X}}$; $s, m, k, T \in \mathbb{N}$ and

$$C \leq m \log\left(\frac{n}{m}\right) + s(\log(m)+1) + k\left(\log(m-1) + 2\log\left(\frac{T-s}{k}\right) + 2\right).$$

with expected regret bounded above by

$$\sum_{i=1}^{s}\sum_{t=1}^{T^i} \mathbb{E}[\mathcal{L}_{01}(y_t^i, \hat{y}_t^i)] - \mathcal{L}_{01}(y_t^i, h_t^i(x_t^i)) \leq \sqrt{2CT} \tag{9}$$

for any $\mathbf{h}^* \in \mathcal{H}_n^T$ such that $k = k(\mathbf{h}^*)$, $m \geq |m(\mathbf{h}^*)|$, $m > 1$.

- The algorithm and analyses is directly based on the "Markov circadian specialist" method of [KAW12].
- We match (7) up to constant factors in the single (multi-) task case.

Predicting $\mathcal{H}_n$ in a switching multitask setting.

Parameters: $\mathcal{H}_n \subseteq \{-1,1\}^{\mathcal{X}}$; $s, m, k, T \in \mathbb{N}$

Initialization: $n := |\mathcal{H}_n|$; $\boldsymbol{\pi}^1 \leftarrow \frac{\mathbf{1}}{n}$; $\mu := \frac{1}{m}$; $\boldsymbol{w}_1^1 = \cdots = \boldsymbol{w}_1^s \leftarrow \mu\mathbf{1}$; $\theta := 1 - \frac{k}{T-s}$; $\phi := \frac{k}{(m-1)(T-s)}$ and

$$\eta := \sqrt{\left(m \log\left(\frac{n}{m}\right) + smH\left(\frac{1}{m}\right) + (T-s)H\left(\frac{k}{T-s}\right) + (m-1)(T-s)H\left(\frac{k}{(m-1)(T-s)}\right)\right)\frac{2}{T}}$$

For $\tau = 1, \ldots, T$

- Receive task $\ell^\tau \in [s]$.
- Receive $x^\tau \in \mathcal{X}$.
- Set $i \leftarrow \ell^\tau$; $t \leftarrow \boldsymbol{\sigma}(\tau)$. % Convert "global time" τ to "local time" $\frac{i}{t}$.
- Predict

$$\boldsymbol{v}^\tau \leftarrow \frac{\boldsymbol{\pi}^\tau \odot \boldsymbol{w}_t^i}{\boldsymbol{\pi}^\tau \cdot \boldsymbol{w}_t^i}, \qquad \hat{h}^\tau \sim \boldsymbol{v}^\tau, \qquad \hat{y}^\tau \leftarrow \hat{h}^\tau(x^\tau).$$

- Receive $y^\tau \in \{-1,1\}$.
- Update:

i) $\forall h \in \mathcal{H}_n,\ c_h^\tau = \mathcal{L}_{01}(h(x^\tau), y^\tau)$ *ii)* $\boldsymbol{\delta} \leftarrow \boldsymbol{w}_t^i \odot \exp(-\gamma \boldsymbol{c}^\tau)$

iii) $\beta \leftarrow (\boldsymbol{\pi}^\tau \cdot \boldsymbol{w}_t^i)/(\boldsymbol{\pi}^\tau \cdot \boldsymbol{\delta})$ *iv)* $\boldsymbol{\epsilon} \leftarrow \mathbf{1} - \boldsymbol{w}_t^i + \beta\boldsymbol{\delta}$

v) $\boldsymbol{\pi}^{\tau+1} \leftarrow \boldsymbol{\pi}^\tau \odot \boldsymbol{\epsilon}$ *vi)* $\boldsymbol{w}_{t+1}^i \leftarrow (\phi(\mathbf{1} - \boldsymbol{w}_t^i) + \theta\beta\boldsymbol{\delta}) \odot \boldsymbol{\epsilon}^{-1}$

Algorithm: Predicting $\mathcal{H}_n$ in a switching multitask setting.

- Requires $\mathcal{O}(n)$ time per trial to predict.
- Maintains a global weight vector $\boldsymbol{\pi} \in \Delta_n$ and task vectors $\boldsymbol{w}^1, \ldots, \boldsymbol{w}^s \in [0,1]^n$.

Theorem (RKHS Hypotheses Classes)

Theorem [HPT20b] (RKHS Hypotheses Classes)

There exists an algorithm with upper parameter estimates, $k \geq k(\mathbf{h}^*)$, $m \geq |m(\mathbf{h}^*)|$,

$$\hat{C} \geq C(\mathbf{h}^*) := \left(\sum_{h \in m(\mathbf{h}^*)} \|h\|_K^2 X_K^2 + 2(s+k-1)m\lceil \log_2 T \rceil^2 + 2m^2 \right),$$

$\hat{X}_K^2 \geq \max_{\tau \in [T]} K(x^\tau, x^\tau)$, and learning rate $\gamma = \sqrt{\frac{\hat{C}\log(2T)}{2Tm}}$ is bounded by

$$\sum_{i=1}^{s} \sum_{t=1}^{T^i} \mathbb{E}[\mathcal{L}_{01}(y_t^i, \hat{y}_t^i)] - \mathcal{L}_{01}(y_t^i, h_t^i(x_t^i)) \leq 4\sqrt{2\hat{C}\, T \log(2T)} \qquad (10)$$

for any $\mathbf{h}^* \in \mathring{\mathcal{H}}_K^T$.

- Next section we provide a few intuitions about the algorithm.

Considerations

In the single task case ($s = 1$) for $\mathcal{H}_n$ in (9) we obtained,

$$R \leq \mathcal{O}\left(\sqrt{\left(m \log \frac{n}{m} + k \log m + k \log \frac{T}{k}\right) T}\right)$$

which matches (7) up to constant factors.
For $\mathring{\mathcal{H}}_K$ we obtain,

$$R \leq \mathcal{O}\left(\sqrt{\left(\sum_{i=1}^{m} \|h_i\|_K^2 X^2 + k \log m + k \log \frac{T}{k}\right) T}\right) \quad \text{[Exponential time in } m\text{]}$$

Whereas with an efficient alg. (Inductive Matrix Completion) we obtain

$$R \leq \mathcal{O}\left(\sqrt{\left(\sum_{i=1}^{m} \|h_i\|_K^2 X^2 + k m (\log T)^2\right) T \log T}\right) \text{ [}\mathcal{O}(T^3)\text{ time per trial]}$$

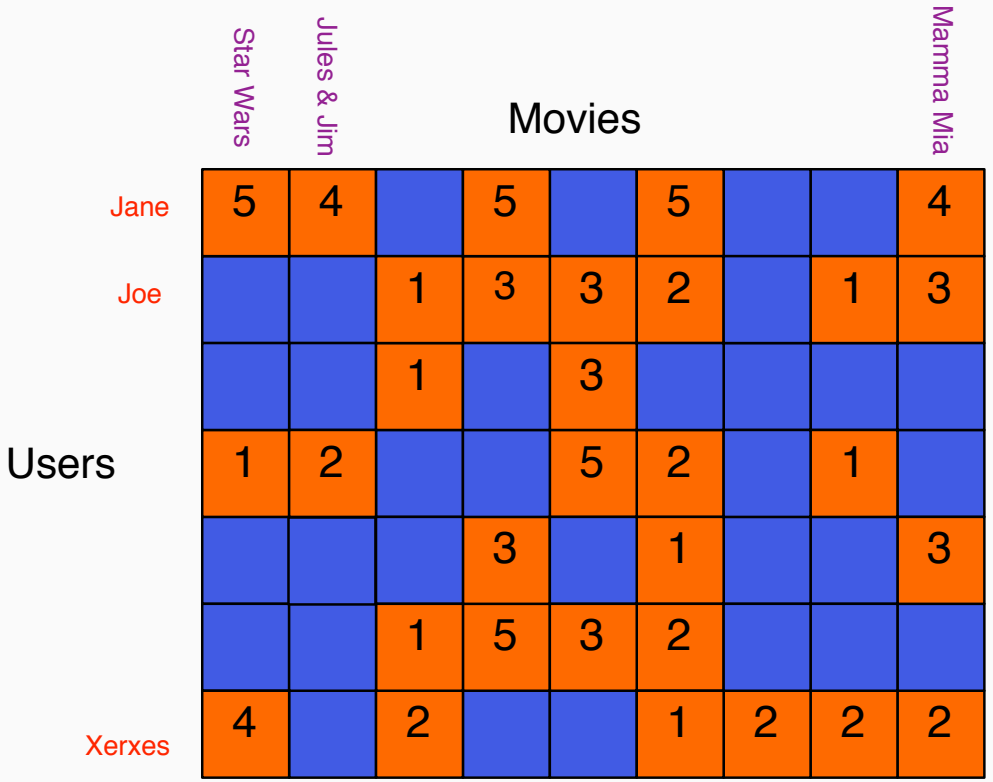
- Thus our efficient algorithm gains poly-log factors in T and we now scale with m rather than $\log m$.
- In many cases the dominant term is the "learner complexity" in red.

Methods

Online Switching Multitask Learning in an RKHS – Overview

- **Idea:** Reduce Online Multitask Learning to Matrix Completion.
  1. Inductive Matrix Completion with Side-Infomation
  2. MEG-IMCSI Algorithm

Matrix Completion



- **Matrix completion:** fill in the missing values.
- For example, the rows may represent users and the columns movies.
- Netflix had a $1,000,000 challenge with a matrix consisted of 480,189 users $\times$ 17,770 users with 100,480,507 rating – won in 2009.

On-Line (Inductive) Binary Matrix Completion

Protocol

For $t = 1$ to T do

Receive indices	$(i_t, j_t) \in \mathcal{I} \times \mathcal{J}$
Predict/Act	$\hat{y}_t \in \{-1, 1\}$
Receive label	$y_t \in \{-1, 1\}$
Incur loss	$L_A = L_A + \mathcal{L}_{01}(y_t, \hat{y}_t)$

- **Side Information:** Kernels $\mathcal{M}^+ : \mathcal{I} \times \mathcal{I} \to \Re$, $\mathcal{N}^+ : \mathcal{J} \times \mathcal{J} \to \Re$.
- **Aim:** Regret bounds of the form,

$$\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \mathcal{L}_{01}(y_t, U_{i_t, j_t}) \leq \text{regret}(\boldsymbol{U}, \mathcal{M}^+, \mathcal{N}^+, T) \quad (\forall \boldsymbol{U} \in \{-1, 1\}^{\mathcal{I} \times \mathcal{J}}$$

- **Note:** The index sets $\mathcal{I}$ and $\mathcal{J}$ may potentially be infinite.
- **Example:** Consider the case of users and movies where associated with each is a feature vector. In the inductive model the side information allows for non-trivial predictions for a user or movie without any observations for that particular movie or user.

Matrix Complexity

Rank Complexity

A natural notion of complexity for matrices is that of a low rank decomposition. A matrix $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ has a rank-k decomposition if there exists $\boldsymbol{P} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{Q} \in \mathbb{R}^{n \times k}$ such that $\boldsymbol{U} = \boldsymbol{P}\boldsymbol{Q}^\top$. Observe that a rank-$k$ decomposition is specified by $k(m+n)$ parameters.

Factor Models

Consider the Netflix problem. A very simple model is that for each user i we associate a factor p_i which is how positive they are about movies and likewise for movies for each movie j we associate a factor q_j which is popular the movie and thus for any particular (user i,movie j) the score associated is just $U_{ij} = p_i \times q_j$ this is a rank-1 decomposition $\boldsymbol{p} \in \mathbb{R}^n$ and $\boldsymbol{q} \in \mathbb{R}^m$. This is an overly simple model ... slightly more complex we might consider there are a k factors and each user and movie has a score with respect to these factor (e.g., comedy, romance, age, education) and total score is the sum of the scores i.e, $U_{ij} = \sum_{s=1}^{k} p_i^{(s)} q_j^{(s)}$, i.e., a rank-$k$ decomposition.

Complexity Measures for Matrices

Max-norm of $\boldsymbol{U} \in \mathbb{R}^{m \times n}$

$$\|\boldsymbol{U}\|_{\max} := \min_{\boldsymbol{P}\boldsymbol{Q}^\top = \boldsymbol{U}} \left\{ \max_{1 \le i \le m} \|\boldsymbol{P}_i\| \times \max_{1 \le j \le n} \|\boldsymbol{Q}_j\| \right\},$$

where the minimum is over all matrices $\boldsymbol{P} \in \Re^{m \times d}$, $\boldsymbol{Q} \in \Re^{n \times d}$ and every integer d. **Note:** $1 \le \|\boldsymbol{U}\|_{\max} \le \min\left(\sqrt{m}, \sqrt{n}\right)$

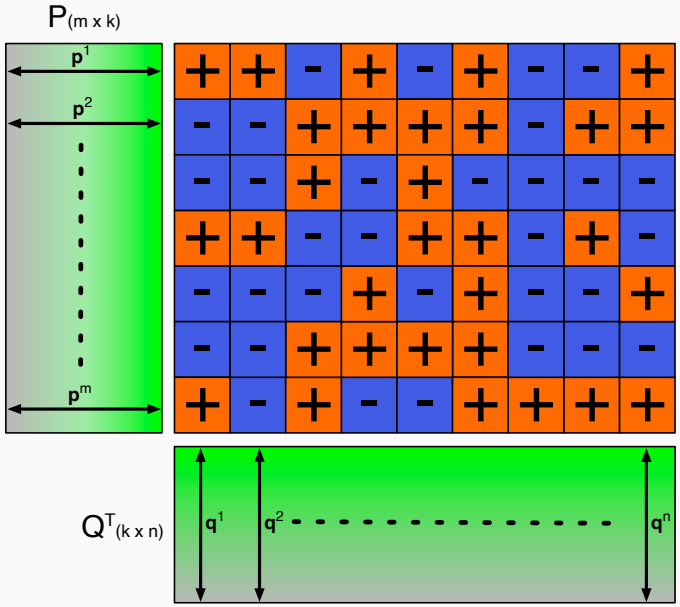
Quasi-dimension of $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ wrt $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$, $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ at margin γ

$$\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \gamma \boldsymbol{U}} \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{M} \hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top \boldsymbol{N} \hat{\boldsymbol{Q}}\right),$$

where the infimum is over all row-normalized matrices $\hat{\boldsymbol{P}} \in \mathcal{N}^{m,d}$ and $\hat{\boldsymbol{Q}} \in \mathcal{N}^{n,d}$ and every integer d. If the infimum does not exist then $\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := +\infty$. If $\boldsymbol{M}, \boldsymbol{N}$ are identity matrices then $\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) = m + n$

Where $\boldsymbol{S}^m_{++}$ is the set of strictly positive definite matrices $m \times m$ matrices, $\mathcal{R}_{\boldsymbol{M}} := \max_{i \in [m]} M^+_{ii}$, $\mathcal{N}^{m,d} := \{\hat{\boldsymbol{P}} \subset \Re^{m \times d} : \|\hat{\boldsymbol{P}}_i\| = 1, i \in [m]\}$ is the set of $m \times d$ *row-normalized* matrices, and $\boldsymbol{P}_i$ is the ith row of $\boldsymbol{P}$.

Max Norm – Illustrated



Definition:

$$\|\boldsymbol{U}\|_{\max} := \min_{\substack{\boldsymbol{P}\in\mathbb{R}^{m\times k}\\ \boldsymbol{Q}\in\mathbb{R}^{n\times k}\\ k\in\mathbb{N}\\ \forall r,s:(\boldsymbol{P}\boldsymbol{Q}^\top)_{rs}=U_{rs}}} \max_{\substack{1\le i\le m\\ 1\le j\le n}} \|\boldsymbol{p}_i\|_2 \|\boldsymbol{q}_j\|_2$$

Expected Regret MEG-IMCSI Algorithm

Theorem [HPT20a]

The expected regret of the MEG-IMCSI *algorithm with parameters* $\gamma \in (0,1]$, $\widehat{\mathcal{D}} \geq \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$, $\gamma = \sqrt{\frac{\widehat{\mathcal{D}}\log(m+n)}{2T}}$, *p.d. matrices* $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$ *and* $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ *is bounded by*

$$\sum_{t\in[T]} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \mathcal{L}_{01}(y_t, U_{i_t j_t}) \leq 4\sqrt{2\frac{\widehat{\mathcal{D}}}{\gamma^2}\log(m+n)T}$$

for all $\boldsymbol{U} \in \{-1,1\}^{m\times n}$ *with* $\|\boldsymbol{U}\|_{max} \leq 1/\gamma$.

Note: Proof builds on techniques from [TRW05, SSSHZ15, GHP13].

Bounds for online matrix completion with general loss functions and with out side-information were first given in [HKS12].

MEG-IMCSI **Algorithm [HPT20a]**

Parameters: Learning rate: $0 < \gamma$ quasi-dimension estimate: $1 \leq \lambda$, margin estimate: $0 < \gamma \leq 1$ and side-information kernels $\mathcal{M}^+ : \mathcal{I} \times \mathcal{I} \to \Re$, $\mathcal{N}^+ : \mathcal{J} \times \mathcal{J} \to \Re$, with $\mathcal{R}_{\mathcal{M}} := \max_{i \in \mathcal{I}} \mathcal{M}^+(i,i)$, $\mathcal{R}_{\mathcal{N}} := \max_{j \in \mathcal{J}} \mathcal{N}^+(j,j)$.

Initialization: $\mathbb{M} \leftarrow \emptyset\,, \mathbb{U} \leftarrow \emptyset\,, \mathcal{I}^1 \leftarrow \emptyset\,,\ \mathcal{J}^1 \leftarrow \emptyset\ .$

For $t = 1, \ldots, T$

- Receive pair $(i_t, j_t) \in \mathcal{I} \times \mathcal{J}$.
- Define

$$(\boldsymbol{M}^t)^+ := (\mathcal{M}^+(i_r, i_s))_{r,s \in \mathcal{I}^t \cup \{i_t\}}\,; \quad (\boldsymbol{N}^t)^+ := (\mathcal{N}^+(j_r, j_s))_{r,s \in \mathcal{J}^t \cup \{j_t\}}\,,$$

$$\tilde{\boldsymbol{X}}^t(s) := \left[\frac{(\sqrt{(\boldsymbol{M}^t)^+})\mathbf{e}^{i_s}}{\sqrt{2\mathcal{R}_{\mathcal{M}}}}; \frac{(\sqrt{(\boldsymbol{N}^t)^+})\mathbf{e}^{j_s}}{\sqrt{2\mathcal{R}_{\mathcal{N}}}}\right]\left[\frac{(\sqrt{(\boldsymbol{M}^t)^+})\mathbf{e}^{i_s}}{\sqrt{2\mathcal{R}_{\mathcal{M}}}}; \frac{(\sqrt{(\boldsymbol{N}^t)^+})\mathbf{e}^{j_s}}{\sqrt{2\mathcal{R}_{\mathcal{N}}}}\right]^\top,$$

$$\tilde{\boldsymbol{W}}^t \leftarrow \exp\left(\log\left(\frac{\lambda}{m+n}\right)\boldsymbol{I}^{|\mathcal{I}^t|+|\mathcal{J}^t|+2} + \sum_{s \in \mathbb{U}} \gamma y_s \tilde{\boldsymbol{X}}^t(s)\right)\,.$$

- Predict

$$Y_t \sim \textsc{Uniform}(-\gamma, \gamma)\,; \bar{y}_t \leftarrow \mathrm{tr}\left(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t\right) - 1\,; \quad \hat{y}_t \leftarrow \mathrm{sign}(\bar{y}_t - Y_t)\,.$$

- Receive label $y_t \in \{-1, 1\}$.
- If $y_t \bar{y}_t \leq \gamma$ then

$$\mathbb{U} \leftarrow \mathbb{U} \cup \{t\}\,,\ \mathcal{I}^{t+1} \leftarrow \mathcal{I}^t \cup \{i_t\},\ \text{and}\ \mathcal{J}^{t+1} \leftarrow \mathcal{J}^t \cup \{j_t\}\,.$$

- Else $\mathcal{I}^{t+1} \leftarrow \mathcal{I}^t$ and $\mathcal{J}^{t+1} \leftarrow \mathcal{J}^t$.

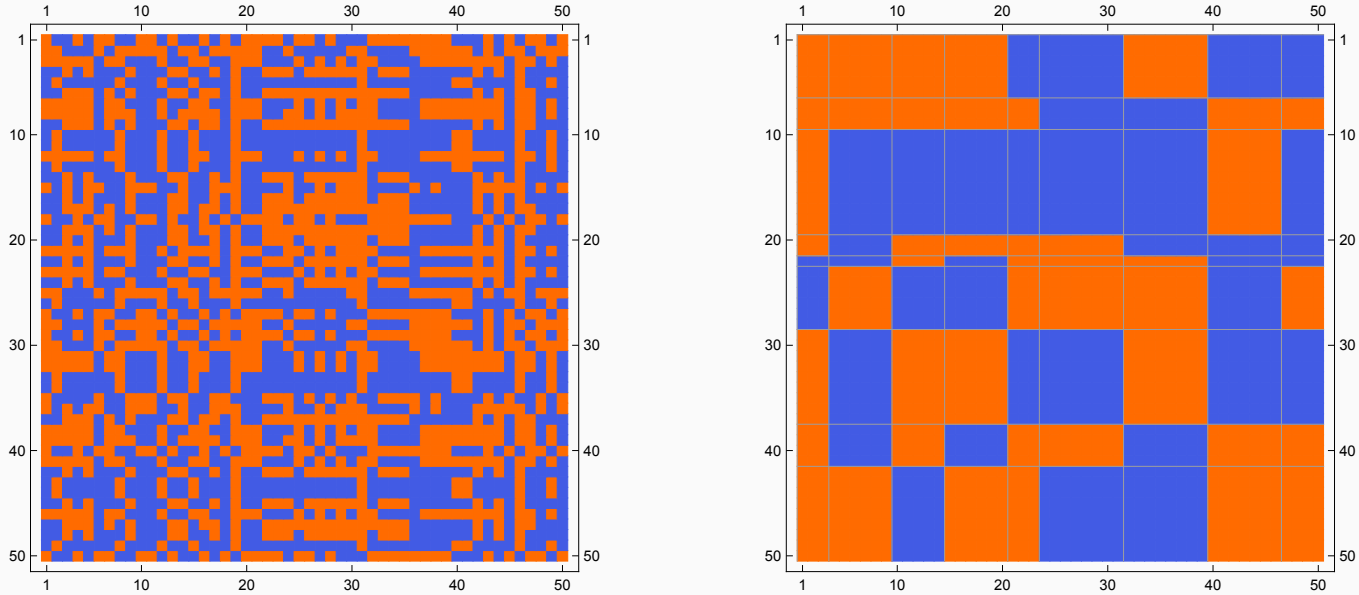
Alg. Matrix Exp. Gradient Inductive Matrix Completion with Side-Infomation (MEG-IMCSI)

MEG-IMCSI **Examples and Bounds**

- Distinct applications induce distinct tunings and bounds of

$$\|\boldsymbol{U}\|_{\max}^2 \mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}^{\gamma}(\boldsymbol{U}) \leq \gamma^{-2}\widehat{\mathcal{D}} \leq \text{“interpretable properties of } \boldsymbol{U} \text{ and side-kernels”}$$

- We give bounds dependent on latent block structure (biclustering)
- A matrix is (k, ℓ)-biclustered if after a permutation of the rows and columns the resultant is a $k \times \ell$ grid of rectangles each labeled as -1 or 1.



A $(9, 9)$-biclustered 50×50 matrix before/after permuting into latent blocks.

- In the following we look at bounding $\|U\|_{\max}$ via latent block structure.

Latent Block Models (definition)

A (k, ℓ)-biclustering is determined by discrete latent k-row (ℓ-column) states. And a $k \times \ell$ interaction matrix $\boldsymbol{U}^*$.

Definition

The set of (k, ℓ)-biclustered matrices is

$$\mathbb{B}_{k,\ell}^{m,n} := \{\boldsymbol{U} \in \{\pm 1\}^{m \times n} : \begin{smallmatrix} \boldsymbol{r} \in [k]^m \\ \boldsymbol{c} \in [\ell]^n \end{smallmatrix}, \boldsymbol{U}^* \in \{\pm 1\}^{k \times \ell}, U_{ij} = U^*_{r_i c_j}, \begin{smallmatrix} i \in [m] \\ j \in [n] \end{smallmatrix}\}.$$

Alternatively, define **block expansion** matrices $\mathcal{B}^{m,d}$,

$$\mathcal{B}^{m,d} := \{\boldsymbol{B} \subset \{0, 1\}^{m \times d} : \|\boldsymbol{B}_i\| = 1, i \in [m], \text{rank}(\boldsymbol{B}) = d\}$$

$$R \; \boxed{} \; C^T = \boxed{}$$

Thus a (k, ℓ)-biclustered matrix $\boldsymbol{U}$ may be decomposed,

$$\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top \text{ for } \boldsymbol{U}^* \in \{\pm 1\}^{k \times \ell}, \boldsymbol{R} \in \mathcal{B}^{m,k} \text{ and } \boldsymbol{C} \in \mathcal{B}^{n,\ell}.$$

Latent models : Low-Rank vs Small Biclustering

Rank Complexity

A matrix $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ has a rank-d decomposition if there exists $\boldsymbol{P} \in \mathbb{R}^{m \times d}$ and $\boldsymbol{Q} \in \mathbb{R}^{n \times d}$ such that $\boldsymbol{U} = \boldsymbol{P}\boldsymbol{Q}^\top$. Observe that a rank-$d$ decomposition is specified by $d(m+n)$ real parameters.

Biclustering Complexity

A matrix $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ has a (k, ℓ)-biclustering if there exists $p \in [k]^m$, $q \in [\ell]^d$, and $U^* \in \mathbb{R}^{k \times \ell}$ such that $U_{ij} = U^*_{p_i, q_j}$. Observe that a (k, ℓ)-biclustering is specified by $k\ell$ real parameters and $m+n$ integers. Define $\mathrm{bcd}(\boldsymbol{U}) := \min(k, \ell)$ where (k, ℓ) is the minimal bicluster of $\boldsymbol{U}$.

Inequalities

$$\|\boldsymbol{U}\|_{\max}^2 \leq \mathrm{rank}(\boldsymbol{U}) \leq \mathrm{bcd}(\boldsymbol{U})$$

Latent Models – Summary

- Low-rank assumption each row/col is associated with latent factors in $P_i, Q_j \in \mathbb{R}^d$ and the matrix entry $U_{ij} = P_i \cdot Q_j$.
- Biclustering assumption each row/col is associated with latent factors in $p(i) \in [k]$, $q(j) \in [\ell]$ and the matrix entry U_{ij} is an arbitrary function of $p(i)$ and $q(j)$.

- Max-norm may be much smaller than rank!

Max-norm of Similarity (Community) Prediction

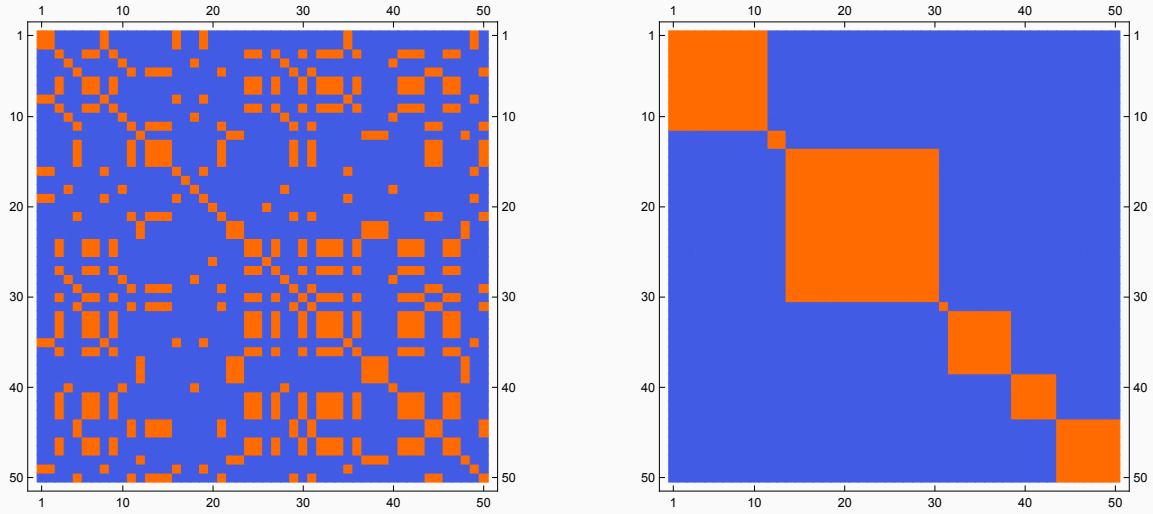
Definition

The class of $m \times m$ **similarity** matrices $\mathcal{S}^m$ is defined as,

$$\mathcal{S}_k^m := \{\boldsymbol{U} = \boldsymbol{R}\boldsymbol{S}_k^*\boldsymbol{C}^\top : \boldsymbol{R} \in \mathcal{B}^{m,k} \text{ and } \boldsymbol{C} \in \mathcal{B}^{m,k}\} \subset \mathbb{B}_{k,k}^{m,m}$$

$$\mathcal{S}^m := \cup_{k=1}^m \mathcal{S}_k^m$$

where $(\boldsymbol{S}_k^*)_{ij} := (2[i == j] - 1)$; "Identity" matrix with '0's $\rightarrow$ '-1's.



Before and After Permutation ($\mathcal{S}_7^{50}$)

- Observe that if $\boldsymbol{U} \in \mathcal{S}_k^m$ then $\mathsf{rank}(\boldsymbol{U}) = k$ but $\|\boldsymbol{U}\|_{\max}^2 = \mathcal{O}(1)$.

Adding Side-Information

Side-Information

- Now assume side information about each row and each column.
- For example with movies we might have “genre” information and for the users we might have “demographics.”
- This side information is expressed via a kernel on the rows $\mathcal{M}^+ : \mathcal{I} \times \mathcal{I} \to \Re$ and a kernel on the columns $\mathcal{N}^+ : \mathcal{J} \times \mathcal{J} \to \Re$.
- The case of no side-information is supplying “identity matrices” for which we have

$$\mathcal{D}^{\gamma}_{\boldsymbol{I}_m, \boldsymbol{I}_n}(\boldsymbol{U}) = m + n\,.$$

Bounding the quasi-dimension $\mathcal{D}$ (Theorem)

Theorem ([A,HPT20a], [B,HPT20b])

[A]: If $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$ then define

$$\mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \begin{cases} 2\,\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + 2\,\mathrm{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} + 2k + 2\ell & (C1) \\ k\,\mathrm{Tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + \ell\,\mathrm{Tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} & (C2) \end{cases}.$$

as the minimum over all decompositions of $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ for $\boldsymbol{R} \in \mathcal{B}^{m,k}$, $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ and $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$ then,

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \le \mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \qquad (\text{if } \|\boldsymbol{U}\|_{\max} \le 1/\gamma)$$

(C1) : $\boldsymbol{M}$ and $\boldsymbol{N}$ are "PDLaplacians"
(C2) : $\boldsymbol{M}$ and $\boldsymbol{N}$ are strictly positive definite.

[B]: If $\boldsymbol{U} \in \mathbb{B}^{m,n}_{m,\ell}$, $\gamma = 1/\sqrt{\ell}$, and (C2) holds then define

$$\mathcal{D}^{\star}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \gamma^2\,\mathrm{tr}((\boldsymbol{U}^*)^\top \boldsymbol{M}\boldsymbol{U}^*)\mathcal{R}_{\boldsymbol{M}} + \mathrm{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}}$$

as min. over decomp. of $\boldsymbol{U} = \boldsymbol{U}^*\boldsymbol{C}^\top$ for $\boldsymbol{U}^* \in \{-1,1\}^{m\times\ell}$ and $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ then

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \le \mathcal{D}^{\star}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \qquad (\gamma = 1/\sqrt{\ell})\,.$$

Bounding the quasi-dimension $\mathcal{D}$ (Intuitions)

Intuitions

- We've replaced the row-normalised matrices $\hat{\boldsymbol{P}}^\top$ ($\hat{\boldsymbol{Q}}^\top$) in $\mathcal{D}$ with the "one-hot" row-normalised matrices $\boldsymbol{R}^\top$ ($\boldsymbol{C}^\top$).
- The term $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}}$ closely resembles the mistake bound for the multi-class kernel perceptron. Where the classes are the "latent factors"
- Qualitatively $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}}$ will be small if latent row factors are "well-predicted" via kernel $\mathcal{M}^+$.
- The inequality [A] in its upper bound depends on $\boldsymbol{U}^*$ only through its dimensionality and the fact that $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$.
- Inequality [A] is independent of γ except for requirement $\|\boldsymbol{U}\|_{\max} \leq 1/\gamma$
- Inequality [B] depends on γ to be its "worst-case" $\|\boldsymbol{U}\|_{\max} = 1/\gamma$
- For the results in this talk we need only inequality [B]

The Reduction

Reducing to Inductive Matrix Completion with Side-Information

1. The column space corresponds to *time*.
2. The row space corresponds to the input space of kernel K.
3. The "time" kernel P engenders the bias that adjacent columns should be similar.

The Graph Laplacian

Definition

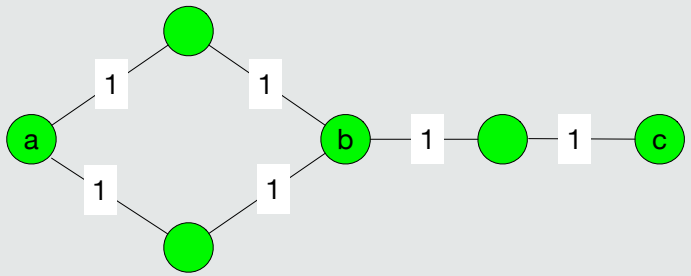
The graph Laplacian is $\boldsymbol{L}(G) := D - W$ where W is the (symmetric non-negatively weighted) adjacency matrix and $\boldsymbol{D} := \text{diag}(d_1, \ldots, d_n)$ is diagonal matrix of (weighted) degrees where $d_v := \sum_{i \neq v} w_{vi}$.

Properties

- The laplacian $\boldsymbol{L}(G)$ is positive semi-definite.
- Observe

$$\mathbf{u}^\top \boldsymbol{L}\mathbf{u} \quad := \quad \sum_{(i,j) \in E(G)} w_{ij}(u_i - u_j)^2$$

- Define $r(i,j)$ to be the effective resistance between vertex i and j if we treat G as a resistive circuit with each edge (resistor) connected with inherent resistance $\frac{1}{w_{ij}}$.



Resistive Network with unit resistors with $r(a,b) = 1$ and $r(b,c) = 2$.

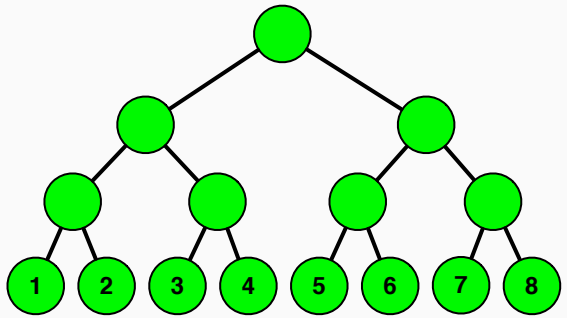
- The diagonal of the "kernel" is bounded by the (effective) resistance diameter

$$\mathcal{R}_L := \max_i L_{ii}^+ \leq \max_{i,j} r(i,j)$$

Path-Tree Kernel

Path-Tree Kernel $P(\cdot,\cdot)$

A *path-tree* kernel $P : [T] \times [T] \to \Re$, is formed via the Laplacian of a fully complete binary *tree* with $N := 2^{\lceil \log_2 T \rceil + 1} - 1$ vertices. The *path* corresponds to the first T leaves of the tree.



Path-Tree Kernel (Length 8)

Lemma (See [HLP09, HPT20b])

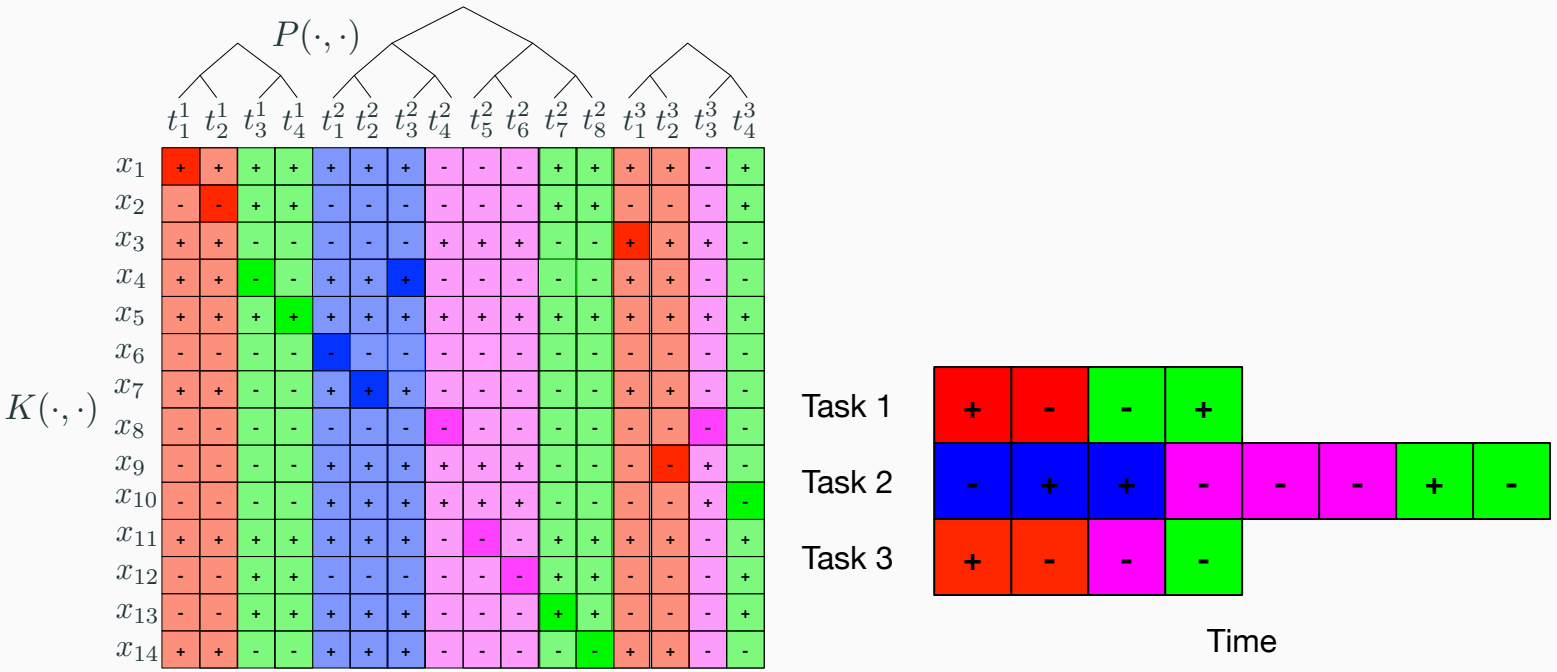
If $f \in \mathcal{H}_P \cap \{0,1\}^T$ then

$$\max_{\tau \in [T]} P(t,t) \leq 2\lceil \log_2 T \rceil$$

$$\|f\|_P^2 \max_{t \in [T]} P(t,t) \leq k(f)\lceil \log_2 T \rceil^2 + 2, ,$$

where $k(f) := \sum_{t=1}^{T-1} [f(t) \neq f(t+1)]$.

Illustration



T=16 trials, s=3 tasks, m=4 modes, k=5 switches

- Emphasised matrix entries correspond to observations.
- Sparse matrix completion at most one observation per column.

Proof Sketch

Theorem

$$R \leq \mathcal{O}\left(\sqrt{\left(\sum_{i=1}^{m} \|h_i\|_K^2 X^2 + km(\log T)^2\right) T \log T}\right)$$

Proof Sketch

1. Recall
$$\sum_{t\in[T]} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \mathcal{L}_{01}(y_t, U_{i_t j_t}) \leq 4\sqrt{2\frac{\widehat{\mathcal{D}}}{\gamma^2}\log(2T)T}$$
for all $\boldsymbol{U} \in \{-1, 1\}^{m\times n}$ with $\|\boldsymbol{U}\|_{\max} \leq 1/\gamma$.
2. Observe that $\boldsymbol{U} \in \mathbb{B}_{T,m}^{T,T}$ (See "Illustration") thus $\|\boldsymbol{U}\|_{\max} \leq \sqrt{m}$.
3. Using
$$\mathcal{D}^{\star}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \gamma^2 \operatorname{tr}((\boldsymbol{U}^*)^\top \boldsymbol{M}\boldsymbol{U}^*)\mathcal{R}_{\boldsymbol{M}} + \operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}}$$
4. Thus,
$$\widehat{\mathcal{D}}\gamma^{-2} = \sum_{i=1}^{m} \|h_i\|_K^2 X^2 + m\operatorname{tr}(\boldsymbol{C}^\top P^{-1}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}}$$
5. By Lemma (assume $\boldsymbol{s} \leq k$) we have $\operatorname{tr}(\boldsymbol{C}^\top P^{-1}\boldsymbol{C}) \in k(\log T)^2$. ■.

Open Problems

1. Improve algorithm efficiency for RKHS $O(T^3)$.
2. Improve RKHS algorithm to replace $m \to \log m$.
3. Lower Bounds (however, see [HPT20b, Prop. 4]).
4. Self-tuning for the parameters.
5. Alternate hypothesis classes (beyond RKHS and finite hypotheses classes).
6. Loss functions beyond $0-1$.
7. Incorporate “drift” into switching model.
8. Algorithms considered are centralized. Distributed algs. with limited communication possible?
9. Experiments!

Thanks

References – 1

[LW94] N. Littlestone and M. Warmuth. *The weighted majority algorithm*, (1994)

[FS97] Y. Freund and R. Schapire. *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*, (1997).

[V90] V. Vovk, *Aggregating strategies*, (1990).

[CLW96] N. Cesa-Bianchi, P. Long, and M. Warmuth. *Worst-case Quadratic Loss Bound for Prediction Using Linear Functions and Gradient Descent,* (1996)

[S11] S. Shalev-Schwartz, *Online Learning and Online Convex Optimization*, (2011)

[HW98] M. Herbster and M. Warmuth. *Tracking the Best Expert*, (1998)

[F00] Y. Freund. *Private communication*, (2000). Also posted on http://www.learning-theory.org

[BW03] O. Bousquet and M.K. Warmuth. Tracking a small set of experts by mixing past posteriors.

[KAW12] W. Koolen, D. Adamskiy, and M. Warmuth. *Putting Bayes to Sleep*, (2012)

[HPT20b] M. Herbster, S. Pasteris and L. Tse. *Online Multitask Learning with Long-Term Memory*, (2020).

References – 2

[HPT20a] M. Herbster, S. Pasteris, and L. Tse. *Online matrix completion with side information*, (2020).

[LMSS07] N. Linial, S. Mendelson, G. Schechtman, and A. Shraibman. *Complexity measures of sign matrices,* (2007).

[TRW05] K. Tsuda, G. Ratsch, and M.K. Warmuth. *Matrix exponentiated gradient updates for on-line learning and bregman projection,* (2005).

[SSSHZ15] S. Sabato, S. Shalev-Shwartz, N. Srebro, Daniel J. Hsu, and T. Zhang. *Learning sparse low- threshold linear classifiers,* (2015).

[GHP13] C. Gentile, M. Herbster, and S. Pasteris. *Online similarity prediction of networked data from known and unknown graphs,* (2013).

[HKS12] E. Hazan, S. Kale, and S. Shalev-Shwartz. *Near-optimal algorithms for online matrix prediction,* (2013).

[HLP09] M. Herbster, G. Lever, and M. Pontil. *Online prediction on large diameter graphs* (2009)