Dying Experts: Efficient Algorithms with Optimal Regret Bounds

Hamid Shayestehmanesh, Sajjad Azami, Nishant Mehta

Department of Computer Science
University of Victoria
Victoria, BC, Canada

33rd Conference on Neural Information Processing Systems (NeurIPS 2019)
Vancouver, Canada

Table of Contents

Table of Contents

Decision-theoretic Online Learning (DTOL)

For round $t = 1, 2, \ldots$

1. *Nature presents a sample* x_t
2. *Learner plays a probability vector* $\boldsymbol{p}_t$ *over* K *experts*
3. *Nature reveals a loss vector* $\boldsymbol{\ell}_t$
4. *Learner suffers* $\hat{\ell}_t = \boldsymbol{p}_t \cdot \boldsymbol{\ell}_t = \sum_{i=1}^{K} p_{i,t} \ell_{i,t}$

Decision-theoretic Online Learning (DTOL)

For round $t = 1, 2, \ldots$

1. *Nature presents a sample* x_t
2. *Learner plays a probability vector* $\boldsymbol{p}_t$ *over* K *experts*
3. *Nature reveals a loss vector* $\boldsymbol{\ell}_t$
4. *Learner suffers* $\hat{\ell}_t = \boldsymbol{p}_t \cdot \boldsymbol{\ell}_t = \sum_{i=1}^{K} p_{i,t}\ell_{i,t}$

The classic notion of regret:

$$R_E(1, T) = \sum_{t=1}^{T} \hat{\ell}_t - \min_{i \in [K]} \sum_{t=1}^{T} \ell_{i,t}$$

Sleeping Experts

- What if the set of experts is changing?

Sleeping Experts

- What if the set of experts is changing?
- "Specialists" framework proposed by Blum (1997) and followed by Freund et al. (1997)

Sleeping Experts

- What if the set of experts is changing?
- "Specialists" framework proposed by Blum (1997) and followed by Freund et al. (1997)

Sleeping Experts Framework:

For round $t = 1, 2, \ldots$

1. *Nature presents a sample* x_t *and availability set* E_a^t
2. *Learner plays a probability vector* $\boldsymbol{p}_t$ *over* K *experts*
3. *Nature reveals a loss vector* $\boldsymbol{\ell}_t$
4. *Learner suffers* $\hat{\ell}_t = \boldsymbol{p}_t \cdot \boldsymbol{\ell}_t = \sum_{i=1}^{K} p_{i,t}\ell_{i,t}$

Regret Notion

- Observe: using classic notion of regret (R_E) is not reasonable anymore
- Some possible notions:

- *(Per-action Regret)* Sum only over the rounds in which the best action is available

Regret Notion

- Observe: using classic notion of regret (R_E) is not reasonable anymore
- Some possible notions:

- *(Per-action Regret)* Sum only over the rounds in which the best action is available
- *(Policy Regret)* Compete with the best policy

Regret Notion

- Observe: using classic notion of regret (R_E) is not reasonable anymore
- Some possible notions:

- *(Per-action Regret)* Sum only over the rounds in which the best action is available
- *(Policy Regret)* Compete with the best policy

- (Ranking Regret) Compete with the best ranking of experts

Ranking Regret

Define π to be an ordering over the set of initial experts E

Ranking Regret

Define π to be an ordering over the set of initial experts E

For example: $\pi_i = (e_3, e_1, e_2)$

Ranking Regret

Define π to be an ordering over the set of initial experts E

For example: $\pi_i = (e_3, e_1, e_2)$

Let Π be the set of all possible orderings of E

Ranking Regret

Define π to be an ordering over the set of initial experts E

For example: $\pi_i = (e_3, e_1, e_2)$

Let Π be the set of all possible orderings of E

Denote by $\sigma^t(\pi)$ the first alive expert of ordering π in round t

Ranking Regret

Define π to be an ordering over the set of initial experts E

For example: $\pi_i = (e_3, e_1, e_2)$

Let Π be the set of all possible orderings of E

Denote by $\sigma^t(\pi)$ the first alive expert of ordering π in round t

$$R_\Pi(1, T) = \sum_{t=1}^{T} \hat{\ell}_t - \min_{\pi \in \Pi} \sum_{t=1}^{T} \ell_{\sigma^t(\pi),t} \ .$$

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic
- As usual, losses can be adversarial or stochastic

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic
- As usual, losses can be adversarial or stochastic
- We are interested in Fully-Adversarial setting

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic
- As usual, losses can be adversarial or stochastic
- We are interested in Fully-Adversarial setting
- Recall that regret of Hedge is $\mathcal{O}(\sqrt{T \log K})$

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic
- As usual, losses can be adversarial or stochastic
- We are interested in Fully-Adversarial setting
- Recall that regret of Hedge is $\mathcal{O}(\sqrt{T \log K})$
- Applying Kleinberg et al. (2010), we get $\mathcal{O}(\sqrt{KT \log K})$ with respect to ranking regret (create all $K!$ orderings)

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic
- As usual, losses can be adversarial or stochastic
- We are interested in Fully-Adversarial setting
- Recall that regret of Hedge is $\mathcal{O}(\sqrt{T \log K})$
- Applying Kleinberg et al. (2010), we get $\mathcal{O}(\sqrt{KT \log K})$ with respect to ranking regret (create all $K!$ orderings)
- They also prove $\Omega(\sqrt{KT \log K})$

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic
- As usual, losses can be adversarial or stochastic
- We are interested in Fully-Adversarial setting
- Recall that regret of Hedge is $\mathcal{O}(\sqrt{T \log K})$
- Applying Kleinberg et al. (2010), we get $\mathcal{O}(\sqrt{KT \log K})$ with respect to ranking regret (create all $K!$ orderings)
- They also prove $\Omega(\sqrt{KT \log K})$
- Hardness results

Previous Results on Sleeping Experts

- Availability set (E_a^t) can be chosen adversarial or stochastic
- As usual, losses can be adversarial or stochastic
- We are interested in Fully-Adversarial setting
- Recall that regret of Hedge is $\mathcal{O}(\sqrt{T \log K})$
- Applying Kleinberg et al. (2010), we get $\mathcal{O}(\sqrt{KT \log K})$ with respect to ranking regret (create all $K!$ orderings)
- They also prove $\Omega(\sqrt{KT \log K})$
- Hardness results

Can we get better results (regret/computation) in the easier case we are interested in?

Dying Expert

- We are interested in a more restricted version of sleeping experts

Dying Expert

- We are interested in a more restricted version of sleeping experts
- Motivated by disqualification or expiration of experts (e.g. fairness)

Dying Expert

- We are interested in a more restricted version of sleeping experts
- Motivated by disqualification or expiration of experts (e.g. fairness)
- The experts can only go to sleep (never wake up)

Dying Expert

- We are interested in a more restricted version of sleeping experts
- Motivated by disqualification or expiration of experts (e.g. fairness)
- The experts can only go to sleep (never wake up)
- Observe: Not all the $K!$ orderings are needed anymore

Dying Expert

- We are interested in a more restricted version of sleeping experts
- Motivated by disqualification or expiration of experts (e.g. fairness)
- The experts can only go to sleep (never wake up)
- Observe: Not all the $K!$ orderings are needed anymore

$\pi_1 = (e_1, e_2, e_3)$	$\pi_2 = (e_2, e_1, e_3)$	$\pi_3 = (e_2, e_3, e_1)$
$\pi_4 = (e_1, e_3, e_2)$	$\pi_5 = (e_3, e_1, e_2)$	$\pi_6 = (e_3, e_2, e_1)$

Table: After e_2 dies

Dying Expert

- We are interested in a more restricted version of sleeping experts
- Motivated by disqualification or expiration of experts (e.g. fairness)
- The experts can only go to sleep (never wake up)
- Observe: Not all the $K!$ orderings are needed anymore

$\pi_1 = (e_1, e_2, e_3)$	$\pi_2 = (e_2, e_1, e_3)$	$\pi_3 = (e_2, e_3, e_1)$
$\pi_4 = (e_1, e_3, e_2)$	$\pi_5 = (e_3, e_1, e_2)$	$\pi_6 = (e_3, e_2, e_1)$

Table: *After* e_2 dies

number of *effective* orderings reduced from 6 to 4

Summary of Our Results

Can we take advantage of this pattern to get better results?

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No
Matching lower bound?	Yes

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No
Matching lower bound?	Yes
Ok, how about efficiency?	Yes

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No
Matching lower bound?	Yes
Ok, how about efficiency?	Yes

What information can help improving regret?

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No
Matching lower bound?	Yes
Ok, how about efficiency?	Yes

What information can help improving regret? *Order of dying*

Question (Known order of dying)	Our Response
Can we improve the upper bound?	Yes: $\mathcal{O}(\sqrt{TK})$

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No
Matching lower bound?	Yes
Ok, how about efficiency?	Yes

What information can help improving regret? *Order of dying*

Question (Known order of dying)	Our Response
Can we improve the upper bound?	Yes: $\mathcal{O}(\sqrt{TK})$
Matching lower bound?	Yes

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No
Matching lower bound?	Yes
Ok, how about efficiency?	Yes

What information can help improving regret? *Order of dying*

Question (Known order of dying)	Our Response
Can we improve the upper bound?	Yes: $\mathcal{O}(\sqrt{TK})$
Matching lower bound?	Yes
Efficiency?	Yes

Summary of Our Results

Can we take advantage of this pattern to get better results?

Question	Our Response
Can we improve $\mathcal{O}(\sqrt{TK \log K})$?	No
Matching lower bound?	Yes
Ok, how about efficiency?	Yes

What information can help improving regret? *Order of dying*

Question (Known order of dying)	Our Response
Can we improve the upper bound?	Yes: $\mathcal{O}(\sqrt{TK})$
Matching lower bound?	Yes
Efficiency?	Yes

+ some more results in Section 5.3 of the paper (will not be discussed in this presentation)

Table of Contents

Lower bound (Unknown Order)

Let m be the number of experts that are going to die

Theorem (Unknown Order Lower Bound)

When the order of dying is unknown, the regret of any algorithm is $\Omega(\sqrt{mT \log K})$.

Lower bound (Unknown Order) Proof

Proof Sketch

- partition the T rounds into $m+1$ days of equal length

Lower bound (Unknown Order) Proof

Proof Sketch

- partition the T rounds into $m+1$ days of equal length
- each day is a game decoupled from the previous ones (goal: no prior info for alg)

Lower bound (Unknown Order) Proof

Proof Sketch

- partition the T rounds into $m + 1$ days of equal length
- each day is a game decoupled from the previous ones (goal: no prior info for alg)
- The days are split into two halves

Lower bound (Unknown Order) Proof

Proof Sketch

- partition the T rounds into $m+1$ days of equal length
- each day is a game decoupled from the previous ones (goal: no prior info for alg)
- The days are split into two halves
- first half: $\ell_{i,t} \sim Bernouli(1/2)$

Lower bound (Unknown Order) Proof

Proof Sketch

- partition the T rounds into $m+1$ days of equal length
- each day is a game decoupled from the previous ones (goal: no prior info for alg)
- The days are split into two halves
- first half: $\ell_{i,t} \sim Bernouli(1/2)$
- best expert of the first half suffers no loss on the second half, the others will suffer $1 - \ell_{j,t}$

Lower bound (Unknown Order) Proof

Proof Sketch

- partition the T rounds into $m+1$ days of equal length
- each day is a game decoupled from the previous ones (goal: no prior info for alg)
- The days are split into two halves
- first half: $\ell_{i,t} \sim \mathit{Bernouli}(1/2)$
- best expert of the first half suffers no loss on the second half, the others will suffer $1 - \ell_{j,t}$
- implies $R_{\Pi}(1, T) = \sum_{s=1}^{m+1} R_{E_a(s)}(\tau_s)$

Lower bound (Unknown Order) Proof

Proof Sketch

- partition the T rounds into $m+1$ days of equal length
- each day is a game decoupled from the previous ones (goal: no prior info for alg)
- The days are split into two halves
- first half: $\ell_{i,t} \sim Bernouli(1/2)$
- best expert of the first half suffers no loss on the second half, the others will suffer $1 - \ell_{j,t}$
- implies $R_\Pi(1, T) = \sum_{s=1}^{m+1} R_{E_a(s)}(\tau_s)$
- using DTOL minimax regret, we get:

$$R_\Pi(1, T) = \sum_{s=1}^{m+1} \sqrt{T/2(m+1)\log(K-s)} = \Omega\left(\sqrt{Tm\log K}\right) \square$$

Upper bound (Unknown Order)

Resetting-Hedge: run Hedge over the set of initial experts E and, after each night, reset the algorithm

Upper bound (Unknown Order)

Resetting-Hedge: run Hedge over the set of initial experts E and, after each night, reset the algorithm

Theorem (Unknown Order Upper Bound)

Resetting-Hedge strategy enjoys a regret of $R_\Pi(1, T) = \mathcal{O}(\sqrt{mT \log K})$.

Upper bound (Unknown Order)

Resetting-Hedge: run Hedge over the set of initial experts E and, after each night, reset the algorithm

Theorem (Unknown Order Upper Bound)

Resetting-Hedge strategy enjoys a regret of $R_\Pi(1, T) = \mathcal{O}(\sqrt{mT \log K})$.

- Resetting can be wasteful in practice

Upper bound (Unknown Order)

Resetting-Hedge: run Hedge over the set of initial experts E and, after each night, reset the algorithm

Theorem (Unknown Order Upper Bound)

Resetting-Hedge strategy enjoys a regret of $R_{\Pi}(1, T) = \mathcal{O}(\sqrt{mT \log K})$.

- Resetting can be wasteful in practice
- Running on $K!$ orderings on the other hand is inefficient

Upper bound (Unknown Order)

Resetting-Hedge: run Hedge over the set of initial experts E and, after each night, reset the algorithm

Theorem (Unknown Order Upper Bound)

Resetting-Hedge strategy enjoys a regret of $R_{\Pi}(1, T) = \mathcal{O}(\sqrt{mT \log K})$.

- Resetting can be wasteful in practice
- Running on $K!$ orderings on the other hand is inefficient
- We will propose an algorithm to implement it efficiently (to be discussed later)

Table of Contents

Lower bound (Known Order)

Theorem (Known Order Lower Bound)

When Learner knows the order of dying, the minimax regret is $\Omega(\sqrt{mT})$.

Lower bound (Known Order)

Theorem (Known Order Lower Bound)

When Learner knows the order of dying, the minimax regret is $\Omega(\sqrt{mT})$.

Proof Sketch

- partition all the rounds to $m/2$ days of equal length

Lower bound (Known Order)

Theorem (Known Order Lower Bound)

When Learner knows the order of dying, the minimax regret is $\Omega(\sqrt{mT})$.

Proof Sketch

- partition all the rounds to $m/2$ days of equal length
- On day s: all experts suffer full loss except for e_{2s-1} and e_{2s}, who will suffer i.i.d. $Bernouli(1/2)$

Lower bound (Known Order)

Theorem (Known Order Lower Bound)

When Learner knows the order of dying, the minimax regret is $\Omega(\sqrt{mT})$.

Proof Sketch

- partition all the rounds to $m/2$ days of equal length
- On day s: all experts suffer full loss except for e_{2s-1} and e_{2s}, who will suffer i.i.d. $Bernouli(1/2)$
- e_{2s-1} and e_{2s} die at the end of day s

Lower bound (Known Order)

Theorem (Known Order Lower Bound)

When Learner knows the order of dying, the minimax regret is $\Omega(\sqrt{mT})$.

Proof Sketch

- partition all the rounds to $m/2$ days of equal length
- On day s: all experts suffer full loss except for e_{2s-1} and e_{2s}, who will suffer i.i.d. *Bernouli*$(1/2)$
- e_{2s-1} and e_{2s} die at the end of day s
- similar to the unknown case's proof, we have

$$R_\Pi(1,T) \geq \sum_{s=1}^{m/2} \frac{1}{L} \min\{\sqrt{T'/2\log 2}, T'\} = \sum_{s=1}^{m/2} \sqrt{T/m} = \Omega\left(\sqrt{mT}\right)$$

□

Upper bound (Known Order)

Strategy: create only the effective experts and run Hedge on them.

Note: we only have $2^m(K - m)$ experts (orderings) instead of $K!$

Upper bound (Known Order)

Strategy: create only the effective experts and run Hedge on them.

Note: we only have $2^m(K - m)$ experts (orderings) instead of $K!$

Theorem (Known Order Upper Bound)

For the case of known order of dying, the strategy as described above achieves a regret of $\mathcal{O}\big(\sqrt{T(m + \log K)}\big)$.

Table of Contents

Number of Effective Experts

Assumption (for simplicity): the experts die in order, e_1 dies first, e_2 second, ...

Number of Effective Experts

Assumption (for simplicity): the experts die in order, e_1 dies first, e_2 second, ...

Behavior of π is a sequence of predictions $(\sigma^1(\pi), \sigma^2(\pi), \ldots, \sigma^T(\pi))$

π and π' behave the same if they use the same initial experts in *every* round.

Number of Effective Experts

Assumption (for simplicity): the experts die in order, e_1 dies first, e_2 second, ...

Behavior of π is a sequence of predictions $(\sigma^1(\pi), \sigma^2(\pi), \ldots, \sigma^T(\pi))$

π and π' behave the same if they use the same initial experts in *every* round.

Set of effective orderings $\mathcal{E} \subseteq \Pi$: for each unique behavior of orderings, there only exists one ordering in $\mathcal{E}$.

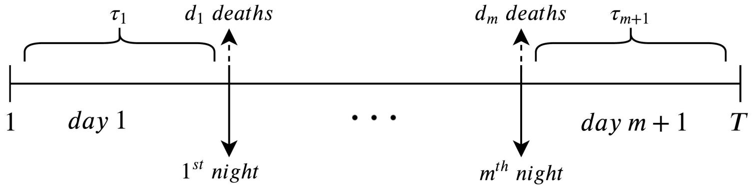
Number of Effective Experts

Theorem (Number of Effective Experts)

In the dying experts setting, for K initial experts and m nights, the number of effective orderings in Π is $f(\{d_1, d_2, \dots d_m\}, A) = A \cdot \prod_{s=1}^{m}(d_s + 1)$.

- d_i is the number of experts that die on i^{th} night
- If no expert dies, i.e. $f(\{\}, A) = A$
- The maximum number of effective experts is $2^m(K - m)$

Illustration



Number of Effective Experts - Proof

- Induction on number of nights, m
- Base: $f(\{\}, A) = A$
- Hypothesis: $f(\{d_2, \ldots d_i\}, A) = A \prod_{s=2}^{i} (d_s + 1)$, Denote this set of effective permutations by $\mathcal{E}_{i-1}$
- Step: Any effective permutation π where $\sigma_1(\pi) = e_i$, one of the experts that dies at first night, will look like (e_i, π') where $\pi' \in \mathcal{E}_{i-1}$ This will create d_1 set of effective permutation of size $\mathcal{E}_{i-1}$. Summing these d_1 new sets with $\mathcal{E}_{i-1}$ give us $\mathcal{E}_i$ of size $(d_1 + 1)|\mathcal{E}_{i-1}|$

Example

$\pi_1 = (e_4, e_2, e_3)$	$\pi_2 = (e_2, e_4, e_3)$	$\pi_3 = (e_2, e_3, e_4)$
$\pi_4 = (e_4, e_3, e_2)$	$\pi_5 = (e_3, e_4, e_2)$	$\pi_6 = (e_3, e_2, e_4)$

Table: All permutations with e_2, e_3, e_4, red permutations are effective

$\pi_1 = (e_1, e_4, e_2, e_3)$	$\pi_2 = (e_1, e_2, e_4, e_3)$	$\pi_3 = (e_1, e_2, e_3, e_4)$
$\pi_4 = (e_1, e_4, e_3, e_2)$	$\pi_5 = (e_1, e_3, e_4, e_2)$	$\pi_6 = (e_1, e_3, e_2, e_4)$

Table: All permutations with e_1, e_2, e_3, e_4 that start with e_1, red permutations are effective

Table of Contents

Review: Hedge

1. Set some $\eta > 0$ and $w_{j,0} = 1$ for $j = 1, 2, \ldots, K$

   For $t = 1, 2, \ldots, T$

2. Set:
$$p_{j,t} = \frac{w_{j,t-1}}{\sum_{j=1}^{K} w_{j,t-1}}$$
   for $j \in [1, K]$

3. Observe loss vector $\boldsymbol{\ell_t}$
4. Suffer loss $\boldsymbol{p}_t \cdot \boldsymbol{\ell}_t$
5. Set $w_{j,t} = w_{j,t-1} e^{-\eta \ell_{j,t}}$ for $j \in [1, K]$

HPU

Algorithm 1: Hedge-Perm-Unknown (HPU)

$c_{i,1} := 1,\ h_{i,1} := (K-1)!,\ E_a := \{e_1, e_2, \ldots e_K\}$
for $t = 1, 2, \ldots T$ **do**
  play $p_{i,t} := 1\left[e_i \in E_a\right]\left(\frac{h_{i,t} \cdot c_{i,t}}{\sum_{j=1}^{k} h_{j,t} \cdot c_{j,t}}\right)$
  receive $(\ell_{1,t}, \ldots, \ell_{K,t})$
  for $e_i \in E_a$ **do**
    $c_{i,t+1} := c_{i,t} \cdot e^{-\eta \ell_{i,t}}$
    $h_{i,t+1} := h_{i,t}$
  if *expert j dies* **then**
    $E_a := E_a \setminus \{e_j\}$
    for $e_i \in E_a$ **do**
      $h_{i,t+1} := h_{i,t+1} \cdot c_{i,t+1} + (h_{j,t+1} \cdot c_{j,t+1})/|E_a|$
      $c_{i,t+1} := 1$

Why HPU works?

Theorem (Hedge Perm Same as Hedge)

At every round, HPK simulates running Hedge on the set of experts $\mathcal{E}$.

1. When $\sigma^t(\pi) = \sigma^t(\pi')$
2. No need to know w_π^t and $w_{\pi'}^t$

Why HPU works?

Theorem (Hedge Perm Same as Hedge)

At every round, HPK simulates running Hedge on the set of experts $\mathcal{E}$.

1. When $\sigma^t(\pi) = \sigma^t(\pi')$
2. No need to know w_π^t and $w_{\pi'}^t$
3. Use $w_\pi^t + w_{\pi'}^t$ instead for the weight of that prediction

Why HPU works?

Theorem (Hedge Perm Same as Hedge)

At every round, HPK simulates running Hedge on the set of experts $\mathcal{E}$.

1. When $\sigma^t(\pi) = \sigma^t(\pi')$
2. No need to know w_π^t and $w_{\pi'}^t$
3. Use $w_\pi^t + w_{\pi'}^t$ instead for the weight of that prediction
4. Let η be the learning rate and L_π^t be the cumulative loss of ordering π up until round t
5. The algorithm maintains $W(\Pi_j^t) = \sum_{\pi \in \Pi_j^t} e^{-\eta L_\pi^{t-1}}$

Proof Sketch

- Group the orderings with similar predictions in one group

$\pi_1 = (e_3, e_1, e_2)$	$\pi_2 = (e_1, e_3, e_2)$	$\pi_3 = (e_1, e_2, e_3)$
$\pi_4 = (e_3, e_2, e_1)$	$\pi_5 = (e_2, e_3, e_1)$	$\pi_6 = (e_2, e_1, e_3)$

Table: All permutations with e_1, e_2, e_3, all alive

$\pi_1 = (e_3, e_1, e_2)$	$\pi_2 = (e_1, e_3, e_2)$	$\pi_3 = (e_1, e_2, e_3)$
$\pi_4 = (e_3, e_2, e_1)$	$\pi_5 = (e_2, e_3, e_1)$	$\pi_6 = (e_2, e_1, e_3)$

Table: All permutations with e_1, e_2, e_3, after e_1 dies

Proof Sketch

- Group the orderings with similar predictions in one group

$\pi_1 = (e_3, e_1, e_2)$	$\pi_2 = (e_1, e_3, e_2)$	$\pi_3 = (e_1, e_2, e_3)$
$\pi_4 = (e_3, e_2, e_1)$	$\pi_5 = (e_2, e_3, e_1)$	$\pi_6 = (e_2, e_1, e_3)$

Table: All permutations with e_1, e_2, e_3, all alive

$\pi_1 = (e_3, e_1, e_2)$	$\pi_2 = (e_1, e_3, e_2)$	$\pi_3 = (e_1, e_2, e_3)$
$\pi_4 = (e_3, e_2, e_1)$	$\pi_5 = (e_2, e_3, e_1)$	$\pi_6 = (e_2, e_1, e_3)$

Table: All permutations with e_1, e_2, e_3, after e_1 dies

- Then, if expert e_j dies, every ordering in the group associated with e_j will be moved to another group and the empty group will be deleted

Proof Sketch

- Group the orderings with similar predictions in one group

$\pi_1 = (e_3, e_1, e_2)$	$\pi_2 = (e_1, e_3, e_2)$	$\pi_3 = (e_1, e_2, e_3)$
$\pi_4 = (e_3, e_2, e_1)$	$\pi_5 = (e_2, e_3, e_1)$	$\pi_6 = (e_2, e_1, e_3)$

Table: All permutations with e_1, e_2, e_3, all alive

$\pi_1 = (e_3, e_1, e_2)$	$\pi_2 = (e_1, e_3, e_2)$	$\pi_3 = (e_1, e_2, e_3)$
$\pi_4 = (e_3, e_2, e_1)$	$\pi_5 = (e_2, e_3, e_1)$	$\pi_6 = (e_2, e_1, e_3)$

Table: All permutations with e_1, e_2, e_3, after e_1 dies

- Then, if expert e_j dies, every ordering in the group associated with e_j will be moved to another group and the empty group will be deleted
- Orderings will distribute to other groups symmetrically after a death

HPK

Algorithm 2: Hedge-Perm-Known (HPK)

$c_{i,1} := 1$, $h_{i,1} := \lceil 2^{K-i-1} \rceil$, $E_a := \{e_1, e_2, \ldots e_K\}$

for $t = 1, 2, \ldots T$ **do**

  play $p_{i,t} := 1\left[e_i \in E_a\right]\left(\frac{h_{i,t} \cdot c_{i,t}}{\sum_{j=1}^{k} h_{j,t} \cdot c_{j,t}}\right)$

  receive $(\ell_{1,t}, \ldots, \ell_{K,t})$

  for *each* $e_i \in E_a$ **do**

    $c_{i,t+1} := c_{i,t} \cdot e^{-\eta \ell_{i,t}}$

    $h_{i,t+1} := h_{i,t}$

  if *expert j dies* **then**

    $E_a := E_a \setminus \{e_j\}$

    for *each* $i = j + 1$ *to* K **do**

      $h_{i,t+1} := h_{i,t+1} \cdot c_{i,t+1} + (h_{j,t+1} \cdot c_{j,t+1}) \left(\frac{\lceil 2^{i-2} \rceil}{2^{K-1-j}}\right)$

      $c_{i,t+1} := 1$

Conclusion and Discussion

- We presented matching upper and lower ranking regret bounds for both the cases of known and unknown order of dying

Conclusion and Discussion

- We presented matching upper and lower ranking regret bounds for both the cases of known and unknown order of dying
- Achieving sublinear regret efficiently in the general sleeping experts problem is as hard as PAC learning DNF

Conclusion and Discussion

- We presented matching upper and lower ranking regret bounds for both the cases of known and unknown order of dying
- Achieving sublinear regret efficiently in the general sleeping experts problem is as hard as PAC learning DNF
- we provided efficient algorithms with optimal regret bounds for both cases in dying experts

Conclusion and Discussion

- We presented matching upper and lower ranking regret bounds for both the cases of known and unknown order of dying
- Achieving sublinear regret efficiently in the general sleeping experts problem is as hard as PAC learning DNF
- we provided efficient algorithms with optimal regret bounds for both cases in dying experts
- Open Question: What will happen in the bandits case?

Conclusion and Discussion

- We presented matching upper and lower ranking regret bounds for both the cases of known and unknown order of dying
- Achieving sublinear regret efficiently in the general sleeping experts problem is as hard as PAC learning DNF
- we provided efficient algorithms with optimal regret bounds for both cases in dying experts
- Open Question: What will happen in the bandits case?
- Open Question: Is there any motivated setting in between unknown and known order?

References

Avrim Blum. Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. *Machine Learning*, 26(1):5–23, 1997.

Yoav Freund, Robert E Schapire, Yoram Singer, and Manfred K Warmuth. Using and combining predictors that specialize. In *In Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*. Citeseer, 1997.

Robert Kleinberg, Alexandru Niculescu-Mizil, and Yogeshwer Sharma. Regret bounds for sleeping experts and bandits. *Machine learning*, 80 (2-3):245–272, 2010.