Memory Bounds for the Expert Problem

Vaidehi Srinivas
Ziyu (Neil) Xu
David Woodruff
Samson Zhou

Northwestern and Carnegie Mellon Universities
Experts Problem

a problem of sequential prediction

<table>
<thead>
<tr>
<th>Day</th>
<th>You</th>
<th>Actual outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>?</td>
<td>Sun</td>
</tr>
<tr>
<td>2</td>
<td>?</td>
<td>Rain</td>
</tr>
<tr>
<td>3</td>
<td>?</td>
<td>Sun</td>
</tr>
<tr>
<td>4</td>
<td>?</td>
<td>Sun</td>
</tr>
</tbody>
</table>
Performance

Make no distributional assumptions
We judge our algorithm based on **regret**.

**Definition (Regret)**

\[
\frac{\text{# of mistakes our algorithm makes more than the best expert}}{\text{# of days}}
\]
Prediction with Expert Advice

<table>
<thead>
<tr>
<th>Day</th>
<th>Day You</th>
<th>Actual outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><img src="image" alt="Sun" /> <img src="image" alt="Cloud" /> <img src="image" alt="Sun" /> <img src="image" alt="Sun" /></td>
<td><img src="image" alt="Sun" /></td>
</tr>
<tr>
<td>2</td>
<td><img src="image" alt="Rain" /> <img src="image" alt="Cloud" /> <img src="image" alt="Sun" /> <img src="image" alt="Rain" /></td>
<td><img src="image" alt="Rain" /></td>
</tr>
<tr>
<td>3</td>
<td><img src="image" alt="Sun" /> <img src="image" alt="Cloud" /> <img src="image" alt="Sun" /> <img src="image" alt="Sun" /></td>
<td><img src="image" alt="Rain" /></td>
</tr>
<tr>
<td>4</td>
<td><img src="image" alt="Sun" /> <img src="image" alt="Cloud" /> <img src="image" alt="Sun" /> <img src="image" alt="Rain" /></td>
<td><img src="image" alt="Sun" /></td>
</tr>
</tbody>
</table>

a fundamental problem of **sequential prediction**
Prediction with Expert Advice

a problem of **sequential prediction**

<table>
<thead>
<tr>
<th>Day</th>
<th>You</th>
<th>Actual outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>🌞UGC</td>
<td>☁️UGC ☀️UGC ☀️UGC</td>
</tr>
<tr>
<td>2</td>
<td>☁️UGC ☁️UGC ☀️UGC</td>
<td>☁️UGC ☁️UGC ☁️UGC</td>
</tr>
<tr>
<td>3</td>
<td>☀️UGC ☁️UGC ☀️UGC ☀️UGC</td>
<td>☀️UGC ☀️UGC ☁️UGC</td>
</tr>
<tr>
<td>4</td>
<td>☀️UGC ☁️UGC ☀️UGC</td>
<td>☁️UGC ☁️UGC ☀️UGC</td>
</tr>
</tbody>
</table>

1/4 Regret

Algorithm makes 2 mistakes
Best expert makes 1 mistake
The Online Learning with Experts Problem

- $n$ experts who decide either $\{0,1\}$ on each of $T$ days
- Algorithm sees expert predictions and predicts either $\{0,1\}$ on each day
- Algorithm sees the outcome, which is in $\{0,1\}$, of each day and can use this information on future days
- The cost of the algorithm is the number of incorrect predictions
- Regret is $(\# \text{ of mistakes we make} - M)/T$, where $M$ is the number of mistakes of best expert
Applications of the Experts Problem

• Ensemble learning, e.g., AdaBoost

• Forecast and portfolio optimization

• Online convex optimization
Weighted Majority (Littlestone, Warmuth 89)

<table>
<thead>
<tr>
<th>Day</th>
<th>Algorithm</th>
<th>Actual outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>weights</td>
<td>1 1 1 1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1-(\varepsilon))</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1-(\varepsilon))</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1-(\varepsilon))</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(1-(\varepsilon))</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1-(\varepsilon))</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1 - (\varepsilon))^2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(1-(\varepsilon))</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1 - (\varepsilon))^2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1 - (\varepsilon))^2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1 - (\varepsilon))^2</td>
<td></td>
</tr>
</tbody>
</table>
Theorem (Deterministic Weighted Majority)

\[ \text{# } m \text{ of mistakes by deterministic weighted majority} \leq (2+\varepsilon)M + \frac{2}{\varepsilon} \ln n \]

where \( M \) is the # of mistakes the best expert makes, \( n \) is # of experts.

\[ (1 - \varepsilon)^M \leq \text{sum of the weights} \leq \left(1 - \frac{\varepsilon}{2}\right)^m n \]
Theorem (Deterministic Weighted Majority)

\[ \# m \text{ of mistakes by deterministic weighted majority} \leq (2+\varepsilon)M + \frac{2}{\varepsilon} \ln n \]

where \( M \) is the \# of mistakes the best expert makes, \( n \) is \# of experts.

Theorem (Randomized Weighted Majority, i.e, Multiplicative Weights)

For \( \varepsilon > 0 \), can construct algorithm \( A \) such that

\[ \mathbb{E}[\# \text{ of mistakes by } A] \leq (1 + \varepsilon)M + O\left(\frac{\ln n}{\varepsilon}\right) \]
Previous Work

- Weighted majority algorithm down-weights each expert that is incorrect on each day and selects the weighted majority as the output.

- Weighted majority algorithm gets $(2+\varepsilon)M + \frac{2}{\varepsilon} \ln n$ total mistakes.

- Randomized weighted majority algorithm randomly follows an expert on a day with probability proportional to the weight of the expert.

- Randomized weighted majority algorithm achieves regret $O \left( \sqrt{\frac{\log T}{T}} \right)$. 
Memory Bounds for the Expert Problem

• These algorithms require $\Omega(n)$ memory to maintain a weight for each expert – but what if $n$ is very large and we want sublinear space?

• Can use no memory and just randomly guess each day – still good if the best expert makes a lot of mistakes but bad if the best expert makes very few mistakes

• What are the space/accuracy tradeoffs for the experts problem?
The Streaming Model

wake up with no memory

except a note from your past self (at most $s$ bits)

see expert predictions for today

make a prediction

see outcome

write a note to your future self (at most $s$ bits)

fall asleep and forget everything

repeat
The Streaming Model

The complete sequence of $T$ days is the data stream.

$$(\text{prediction}_1, \text{outcome}_1), \ldots, (\text{prediction}_T, \text{outcome}_T)$$

**Definition (Arbitrary Order Model)**

An adversary chooses a worst-case set of outcomes and orderings of the days in the stream beforehand

**Definition (Random Order Model)**

An adversary chooses a worst-case set of outcomes, then the order of days is randomly shuffled
Natural Ideas

• What if we can just identify the best expert?

• This requires $\Omega(n)$ space
Set Disjointness Communication Problem

• **Set disjointness communication problem**: Alice has a set $X \in \{0,1\}^n$ and Bob has a set $Y \in \{0,1\}^n$ and the promise is that either $|X \cap Y| = 0$ or $|X \cap Y| = 1$

• Set disjointness requires total (randomized) communication $\Omega(n)$
Reduction

• Holds even for 2 days (can copy each day T/2 times if desired)

• Alice creates a stream $S$ so that each element of $X$ is an expert that is correct on day 1

• Bob creates a stream $S'$ so that each element of $Y$ is an expert that is correct on day 2
Reduction

- Alice runs streaming algorithm $A$ on the stream $S$ and passes the state of $A$ to Bob, who continues the algorithm on the stream $S'$.

- At the end, $A$ will output an expert $i \in [n]$, and then Alice and Bob will check whether $X \cap Y = i$.

- Solves set disjointness* so $A$ must use $\Omega(n)$ space.

- Not end of story: low-regret algorithm need not find best expert!
Our Results (I)

• Any algorithm that achieves $\delta < \frac{1}{2}$ regret with probability at least $\frac{3}{4}$ must use $\Omega \left( \frac{n}{\delta^2 T} \right)$ space

• Lower bound holds for arbitrary-order, random-order, and i.i.d. streams
Our Results (II)

• There exists an algorithm that uses $O \left( \frac{n}{\delta^2 T} \log^2 n \log \frac{1}{\delta} \right)$ space and achieves expected regret $\delta > \sqrt{\frac{8 \log n}{T}}$ in the random-order model.

• The algorithm is almost-tight with the space lower bounds and oblivious to $M$, the number of mistakes made by the best expert.

• Can achieve regret almost matching randomized weighted majority.

• Result extends to general costs in $[0, \rho]$ with expected regret $\rho \delta$. 
Our Results (III)

• For $M = O\left(\frac{\delta^2 T}{\log^2 n}\right)$ and $\delta > \sqrt{\frac{128 \log^2 n}{T}}$, there exists an algorithm that uses $\tilde{O}\left(\frac{n}{\delta T}\right)$ space and achieves regret $\delta$ with high probability.

• The algorithm beats the lower bounds, showing that the hardness comes from the best expert making a lot of mistakes.

• Can achieve regret almost matching randomized weighted majority.

• The algorithm is oblivious to $M$, the number of mistakes made by the best expert.
Format

- Part 1: Background
- Part 2: Lower Bound
- Part 3: Arbitrary Model
- Part 4: Random-Order Model
Lower Bound

• Any algorithm that achieves $\delta < \frac{1}{2}$ regret with probability at least $\frac{3}{4}$ must use $\Omega \left( \frac{n}{\delta^2 T} \right)$ space

• Lower bound holds for arbitrary-order, random-order, and i.i.d. streams
Communication Problem for Lower Bound

- Distributed detection problem
- $\varepsilon$-DIFFDIST problem: $T$ players each hold $n$ bits and must distinguish between two cases.
  - **Case 1**: (NO) Every bit of every player is drawn i.i.d. from a fair coin, i.e., a Bernoulli distribution with parameter $\frac{1}{2}$
  - **Case 2**: (YES) An index $L \in [n]$ is selected arbitrarily. The $L$-th bit of each player is chosen i.i.d. from a Bernoulli distribution with parameter $\frac{1}{2} + \varepsilon$ and all the other bits are chosen i.i.d. from a fair coin
- Blackboard communication model
<table>
<thead>
<tr>
<th>NO</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>H T H T H</td>
<td>H T H T H</td>
</tr>
<tr>
<td>T H H H H</td>
<td>T H H H H</td>
</tr>
<tr>
<td>H H T H T</td>
<td>H H T H T</td>
</tr>
<tr>
<td>H T H T T</td>
<td>H H H H H</td>
</tr>
<tr>
<td>H T H T T</td>
<td>H H H H H</td>
</tr>
</tbody>
</table>

**Communication Problem for Lower Bound**
\( \varepsilon \)-DIFFDIST Problem

- \( \varepsilon \)-DIFFDIST problem: \( T \) players each hold \( n \) bits and must distinguish between two cases.

- Protocol: Randomly choose \( \tilde{O} \left( \frac{1}{\varepsilon^2} \right) \) players and send all bits of those players, see whether some bit has bias at least \( \frac{\varepsilon}{2} \)
Communication Problem for Lower Bound

<table>
<thead>
<tr>
<th></th>
<th>YES</th>
<th>NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>H T H T T H</td>
<td>H T H T H</td>
</tr>
<tr>
<td>Bob</td>
<td>T H H H H H</td>
<td>T H H H H</td>
</tr>
<tr>
<td>Charlie</td>
<td>H H T H T</td>
<td>H H T H T</td>
</tr>
<tr>
<td>Donald</td>
<td>H T H T T</td>
<td>H H T T T</td>
</tr>
</tbody>
</table>
ε-DIFFDIST Problem

- ε-DIFFDIST problem: \( T \) players each hold \( n \) bits and must distinguish between two cases.
- Protocol: Randomly choose \( \tilde{O} \left( \frac{1}{\varepsilon^2} \right) \) players and send all bits of those players, see whether some bit has bias at least \( \frac{\varepsilon}{2} \)
- Communication of protocol: \( \tilde{O} \left( \frac{n}{\varepsilon^2} \right) \)
- Theorem: \( \Omega \left( \frac{n}{\varepsilon^2} \right) \) communication is necessary
**ɛ-DIFFDIST Problem**

- **Theorem:** \( \Omega \left( \frac{n}{\varepsilon^2} \right) \) communication is necessary

- **Fact:** \( \Omega \left( \frac{1}{\varepsilon^2} \right) \) samples are necessary to distinguish between a fair coin, i.e., a Bernoulli distribution with parameter \( \frac{1}{2} \) and a coin with bias \( \varepsilon \)

- **Intuition:** players roughly need to solve the single coin problem on each of the \( n \) coins (actually just need the OR of \( n \) instances)
ε-DIFFDIST Problem

• Formally, all the coins are independent in the NO distribution

• Can use a direct sum theorem for OR [BJKS04], so reduces to showing high information cost under NO distribution on a single coin

• \( \Omega \left( \frac{1}{\varepsilon^2} \right) \) information necessary to distinguish between a single fair coin, i.e., a Bernoulli distribution with parameter \( \frac{1}{2} \) and a coin with bias \( \varepsilon \), even when information is measured under the NO distribution

  • Uses strong data processing inequality [DJWZ13, GMN14, BGM+16]
ε-DIFFDIST Summary

• **ε-DIFFDIST problem**: $T$ players each hold $n$ bits and must distinguish between two cases.

• **Case 1**: (NO) Every index for every player is drawn i.i.d. from a fair coin, i.e., a Bernoulli distribution with parameter $\frac{1}{2}$

• **Case 2**: (YES) An index $L \in [n]$ is selected arbitrarily. The $L$-th bit of each player is chosen i.i.d. from a Bernoulli distribution with parameter $\frac{1}{2} + \varepsilon$ and all the other bits are chosen i.i.d. from a fair coin

• **Fact**: $\Omega \left( \frac{n}{\varepsilon^2} \right)$ communication is necessary to solve the problem
Reduction Intuition

• Each player in the \(\epsilon\)-DIFFDIST Problem corresponds to a different day

• Each bit in the \(\epsilon\)-DIFFDIST Problem corresponds to a different expert

• Reduction: distinguishing whether there exists a slightly biased random bit corresponds to distinguishing whether there exists a slightly “better” expert
# Reduction Challenge

<table>
<thead>
<tr>
<th>Day</th>
<th>You</th>
<th>Actual outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><img src="sun.png" alt="sun" /> <img src="cloud.png" alt="cloud" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /></td>
<td><img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /></td>
</tr>
<tr>
<td>2</td>
<td><img src="cloud.png" alt="cloud" /> <img src="cloud.png" alt="cloud" /> <img src="sun.png" alt="sun" /> <img src="cloud.png" alt="cloud" /></td>
<td><img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /></td>
</tr>
<tr>
<td>3</td>
<td><img src="sun.png" alt="sun" /> <img src="cloud.png" alt="cloud" /> <img src="cloud.png" alt="cloud" /> <img src="sun.png" alt="sun" /></td>
<td><img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /></td>
</tr>
<tr>
<td>4</td>
<td><img src="sun.png" alt="sun" /> <img src="cloud.png" alt="cloud" /> <img src="sun.png" alt="sun" /> <img src="cloud.png" alt="cloud" /></td>
<td><img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /> <img src="sun.png" alt="sun" /></td>
</tr>
</tbody>
</table>
Reduction

• We would like to use an online learning with experts algorithm for solving \( \varepsilon \text{-DIFFDIST} \) Problem for \( \varepsilon = O(\delta) \)

• However, an algorithm with bad guarantees can still have good cost by just outputting 1 every day

• Use masking argument – outcome of each day is masked by an independent fair coin flip on each day (expert advice also flipped)
Reduction Challenge

<table>
<thead>
<tr>
<th>Day</th>
<th>Actual outcome</th>
<th>You</th>
<th>Actual outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
</tr>
<tr>
<td>2</td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
</tr>
<tr>
<td>3</td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
</tr>
<tr>
<td>4</td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
<td><img src="image" alt="Icon" /></td>
</tr>
</tbody>
</table>

MASK=1

MASK=0

MASK=1

MASK=1
Reduction

• For $\delta < \frac{1}{2}$, if there is no biased coin, no expert and no algorithm will do better than $\frac{1}{2} + \frac{\delta}{3}$ with probability at least $\frac{1}{4}$.

• For $\delta < \frac{1}{2}$, if there is a biased coin, an expert will do better than $\frac{1}{2} + \frac{2\delta}{3}$ with probability at least $\frac{1}{4}$.
Reduction Summary

• The online learning with experts algorithm with regret $\delta$ will be able to solve the $\epsilon$-DIFFDIST Problem with probability at least $\frac{3}{4}$ for $\epsilon = O(\delta)$. Must use $\Omega\left(\frac{n}{\delta^2}\right)$ total communication.

• Any algorithm that achieves $\delta < \frac{1}{2}$ regret with probability at least $\frac{3}{4}$ must use $\Omega\left(\frac{n}{\delta^2 T}\right)$ space.
Format

❖ Part 1: Background
❖ Part 2: Lower Bound
❖ Part 3: Arbitrary Model
❖ Part 4: Random-Order Model
No Mistake Regime

• For \( M = O\left(\frac{\delta^2 T}{\log^2 n}\right) \) and \( \delta > \sqrt{\frac{128 \log^2 n}{T}} \), there exists an algorithm that uses \( \tilde{O}\left(\frac{n}{\delta T}\right) \) space and achieves regret \( \delta \) with high probability

• We know there is a really accurate expert. What if we iteratively pick “pools” of experts and delete them if they run “poorly”? 
Reduction Problem

<table>
<thead>
<tr>
<th>Day</th>
<th>You</th>
<th>Actual outcome</th>
</tr>
</thead>
</table>
| 1   | ![Sun](#) ![Cloud](#) ![Sun](#) ![Sun](#) | ![Sun](#) ![Rain](#) |}
| 2   | ![Cloud](#) ![Cloud](#) ![Sun](#) ![Cloud](#) | ![Cloud](#) ![Sun](#) |}
| 3   | ![Sun](#) ![Cloud](#) ![Cloud](#) ![Sun](#) | ![Sun](#) ![Rain](#) |}
| 4   | ![Sun](#) ![Cloud](#) ![Sun](#) ![Cloud](#) | ![Sun](#) ![Rain](#) |}
No Mistake Regime

- If iteratively pick pool of next $k$ experts (“rounds”) and output the majority vote of the pool while deleting any incorrect expert, each pool will have at most $O(\log k)$ errors.

- If best expert makes no mistakes, use $\frac{n}{k}$ pools to achieve regret $\delta T$, means setting $k = O\left(\frac{n \log n}{\delta T}\right)$.
No Mistake Regime Summary

• **Algorithm**: Iteratively pick pool of $k = \tilde{O} \left( \frac{n}{\delta T} \right)$ experts (“rounds”) and output the majority vote of the pool while deleting any incorrect expert.

• If the number of rounds is small, the pools must have done well so the overall regret is small.

• The number of rounds cannot be large because at some point the best expert would have been chosen and retained.
“Low-Mistake” Regime

- **Algorithm**: Iteratively pick pool of next $k = \tilde{O}\left(\frac{n}{\delta T}\right)$ experts and output the majority vote of the pool while deleting any incorrect expert.

- If best expert makes $M$ mistakes, use $\frac{nM}{k}$ pools to achieve regret $\delta T$, means setting $k = \tilde{O}\left(\frac{nM}{\delta T}\right)$, but this is too large!
Randomly Sampling Pools

- **Fix**: Randomly sample pools of experts instead of iteratively picking pools

- **Problem**: Cannot guarantee that the best expert will be retained
“Low-Mistake” Regime

- **Algorithm**: Repeatedly sample a pool of $k = \tilde{O}\left(\frac{n}{\delta T}\right)$ experts and output the majority vote of the pool while deleting any expert with lower than $1 - \frac{\delta}{8 \log n}$ accuracy since it was sampled.

WANT TO SHOW

- If the number of rounds is small, the pools must have done well so the overall regret is small.
- The number of rounds cannot be large because at some point the best expert would have been sampled and retained.
“Low-Mistake” Regime: First Property

• **Algorithm**: Repeatedly sample a pool of \( k = \tilde{O}\left(\frac{n}{\delta T}\right) \) experts and output the majority vote of the pool while deleting any expert with lower than \( 1 - \frac{\delta}{8 \log n} \) accuracy since it was sampled.

• **Lemma**: A pool used for \( t \) days can only make \( \frac{t\delta}{2} + 4 \log n \) mistakes.

• For the algorithm to make \( T\delta \) mistakes, need at least \( \frac{T\delta}{8 \log n} \) rounds.
“Low-Mistake” Regime: Second Property

• For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8 \log n}$ rounds

• “BAD” day: the best expert is deleted by the pool if it is sampled on that day

• $|\text{BAD}| \leq \frac{8M \log n}{\delta}$
“Low-Mistake” Regime: Second Property

• For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8 \log n}$ rounds.

• Using that $|BAD| \leq \frac{8M \log n}{\delta}$ and $M = O\left(\frac{\delta^2 T}{\log^2 n}\right)$, then at least $\frac{T\delta}{16 \log n}$ rounds starting on good days.

• $O\left(\frac{n \log^2 n}{\delta T}\right)$ experts sampled in each round $\rightarrow$ low probability don’t sample best expert on a good day.
Analysis

• Define a set of random variables $d_1, d_2, \ldots$ for each round’s day
• Given $d_i$, draw $d_{i+1}$ from the distribution of possible days for the next round based on possible experts sampled in the pool conditioned on entire history
Arbitrary Order Model Summary

• **Algorithm**: Repeatedly sample a pool of \( k = \tilde{O}\left(\frac{n}{\delta T}\right) \) experts and output the majority vote of the pool while deleting any expert with lower than \( 1 - \frac{\delta}{8 \log n} \) accuracy since it was sampled.

• If the number of rounds is small, the pools must have done well so the overall regret is small.

• The number of rounds cannot be large because at some point the best expert would have been sampled and retained.
Format

- Part 1: Background
- Part 2: Lower Bound
- Part 3: Arbitrary Model
- Part 4: Random-Order Model
Random-Order Streams

• **Algorithm**: Repeatedly sample a pool of \( k = \tilde{O}\left(\frac{n}{\delta^2 T}\right) \) experts and run multiplicative weights on pool, resample if the expected cost of the pool over \( t \) time “is bad”

• Can compute this expected cost, so if it doesn’t follow the theory, it means you didn’t sample the best expert

• Main idea: there are no BAD days
  • we will never delete the pool if it contains the best expert
Summary of Results

• Any algorithm achieving $\delta < \frac{1}{2}$ regret with probability $\frac{3}{4}$ uses $\Omega \left( \frac{n}{\delta^2 T} \right)$ space
• There is an algorithm using $O \left( \frac{n}{\delta^2 T} \log^2 n \right)$ space in the random-order model
• For $M < \frac{\delta^2 T}{1280 \log^2 n}$, there exists an algorithm using $\tilde{O} \left( \frac{n}{\delta T} \right)$ space in the arbitrary-order model with regret $\delta$
• If the costs are in $[0, \rho]$, the regret is $\rho \delta$ for both models

• Questions: tight bounds for arbitrary order streams? how general is this framework?
Followup Work

• [Peng, W, Zhang, Zhou] Any deterministic algorithm must use Omega(n) bits of memory to achieve constant regret
  
  • Seems to generalize to a tight Omega(nM/T) bits (still verifying this)

• [Peng, W, Zhang, Zhou] Black box adversarial robustness with constant regret and roughly $n/(\delta T^{0.5})$ memory

• [Peng, Zhang] Let $n << T$. There’s an algorithm with poly(n) $T^{2/(2+\delta)}$ errors and using $n^\delta$ memory for any $\delta \in (0,1)$
Follow the Perturbed Leader

Theorem (Kalai and Vempala 2005): Expected number of mistakes by the algorithm is at most \( \frac{O(\ln n)}{\varepsilon} + (1 + \varepsilon)M \)
Multiplicative Weights Algorithm

\begin{algorithm}
\caption{The multiplicative weights algorithm.}
\textbf{Input:} Number $n$ of experts, number $T$ of rounds, parameter $\varepsilon$
\begin{algorithmic}
\State Initialize $w_i^{(1)} = 1$ for all $i \in [n]$.
\For{$t \in [T]$}
\State $p_i^{(t)} \leftarrow \frac{w_i^{(t)}}{\sum_{i \in [n]} w_i^{(t)}}$
\State Follow the advice of expert $i$ with probability $p_i^{(t)}$.
\State Let $c_i^{(t)}$ be the cost for the decision of expert $i \in [n]$.
\State $w_i^{(t+1)} \leftarrow w_i^{(t)} \left(1 - \varepsilon c_i^{(t)}\right)$
\EndFor
\end{algorithmic}
\end{algorithm}

- **Theorem** (Arora, Hazan, Kale 2012): Expected cost of the algorithm is
  \[ \sum_{t=1}^{T} \sum_{i=1}^{n} c_i^{(t)} p_i^{(t)} \leq \frac{\ln n}{\varepsilon} + (1 + \varepsilon) \sum_{t=1}^{T} c_i^{(t)} \text{ for each } i \in [n] \] (and in particular the best expert), i.e.,
  \[ \leq \frac{\ln n}{\varepsilon} + (1 + \varepsilon)M \]
- $\varepsilon$ is trade-off term between multiplicative and additive error
Follow the Perturbed Leader

Algorithm 2 The follow the perturbed leader algorithm (FPL*) from [KV05], instantiated for the experts problem.

**Input:** Number $n$ of experts, number $T$ of rounds, parameter $\varepsilon$

1. for $t \in [T]$ do
2.     for $i \in [n]$ do
3.         Choose $p_i^{(t)}$ independently, according to $\pm(2r/\varepsilon)$, where $r$ is drawn from a standard exponential distribution
4.     end for
5. end for
6. Follow the expert $i$ for whom the sum of their total cost so far and $p_i^{(t)}$ is the lowest

- **Theorem** (Kalai and Vempala 2005): Expected cost of the algorithm is
  \[
  \frac{O(\ln n)}{\varepsilon} + (1 + \varepsilon) \sum_{t=1}^{T} c_i^{(t)}
  \]
  for each $i \in [n]$ (and in particular the best expert), i.e.,
  \[
  \leq \frac{O(\ln n)}{\varepsilon} + (1 + \varepsilon)M
  \]
- $\varepsilon$ is trade-off term between multiplicative and additive error
Algorithm 2 The follow the perturbed leader algorithm (FPL*) from [KV05], instantiated for the experts problem.

Input: Number $n$ of experts, number $T$ of rounds, parameter $\epsilon$

1: for $t \in [T]$ do
2:   for $i \in [n]$ do
3:     Choose $p_i^{(t)}$ independently, according to $\pm(2r/\epsilon)$, where $r$ is drawn from a standard exponential distribution
4:   end for
5: Follow the expert $i$ for whom the sum of their total cost so far and $p_i^{(t)}$ is the lowest
6: end for
Random-Order Streams: First Property

**Structural lemma:** Let $X_1, ..., X_t$ be independent random variables in $[0,1]$ with expectation $\alpha$ and $X$ be their sum. Then $\Pr[|X - \alpha t| \geq 4\sqrt{t \log T}] \leq \frac{1}{T^2}$

By the guarantee for multiplicative weights for $\varepsilon = \frac{\delta}{2}$, the cost of each pool is at most $\left(1 + \frac{\delta}{2}\right)\left(\alpha t + 4\sqrt{t \log T}\right) + \frac{2\ln n}{\delta}$

For $\delta > \sqrt{\frac{16 \log^2 n}{T}}$, $\delta > \frac{M}{T}$, number of rounds must be at least $\Omega\left(\frac{\delta^2 T}{\log n}\right)$
Random-Order Streams: Second Property

- Number of rounds must be at least $\Omega \left( \frac{\delta^2 T}{\log n} \right)$
- Must avoid sampling the best expert on at least $\Omega \left( \frac{\delta^2 T}{\log n} \right)$ rounds
- $O \left( \frac{n \log^2 n}{\delta^2 T} \right)$ experts sampled in each round $\rightarrow$ low probability
- Must use same “decoupling” argument
- Similar analysis for $\delta \leq \frac{M}{T}$
“Low-Mistake” Regime: Second Property

• For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8 \log n}$ rounds

• “BAD” day: the best expert is deleted by the pool if it is sampled on that day

• $|\text{BAD}| \leq \frac{8M \log n}{\delta}$ and $M < \frac{\delta^2 T}{1280 \log^2 n}$, so the remaining rounds must be sampled on “GOOD” days and avoid the best expert

• Must avoid sampling the best expert on at least $\frac{T\delta}{16 \log n}$ rounds

• $O\left(\frac{n \log^2 n}{\delta T}\right)$ experts sampled in each round $\rightarrow$ low probability
Guarantee for Weighted Majority

**Theorem (Deterministic Weighted Majority)**

\[
\text{# of mistakes by deterministic weighted majority} \leq 2.41 (M + \log_2 n)
\]

where \(M\) is the # of mistakes the best expert makes, \(n\) is # of experts.

- \((\frac{1}{2})^M \leq \text{sum of the weights} \leq (\frac{3}{4})^m \frac{n}{n}

- \(m \leq \frac{M + \log_2 n}{\log_2 \frac{4}{3}}\)
“Low-Mistake” Regime: Second Property

- For the algorithm to make $T\delta$ mistakes, need at least $\frac{T\delta}{8 \log n}$ rounds.
- To fail, algorithm must not sample the best expert on a “GOOD” day.
A Bad Case Study

- Suppose $\delta = \frac{1}{2}$
- Example shows that the pool of $k = 8$ sampled experts can make roughly $T - T/k$ errors