Scalable Deletion-Robust Submodular Maximization: Data Summarization with Privacy and Fairness Constraints

Reproduced By: Rohit Musti (on my own)

December 5, 2019
Motivation

- We now have huge amounts of data generated by users
- Stakeholders analyze data for insights
- Insights incur the cost of user privacy
- Algorithms amplify bias in data
Motivation

- Not only an ethical obligation, a legal one as well
  - GDPR outlines the right to be forgotten
  - Title VII of the Civil Rights Act (USA) bans discriminative hiring
- Challenges:
  - Expensive to train bespoke models for protected groups
  - We might not know sensitive features!
- What if we could remove sensitive features?
  - Geo-location data
  - Skin colour
  - Gender
  - Age
- Without compromising on accuracy!
- Submodular Optimization:
  - Formalizes the idea of diminishing returns

\[
A \text{ function is submodular if } \quad \text{Given } A \in B \in V, j \not\in A \not\in B, j \in V \\
\quad f(A \cup j) - f(A) \geq f(B \cup j) - f(B)
\]

- Problems like ML, web search, social network, crowd sourcing, user modeling

Nemhauser et al (1978) demonstrated a simple greedy algorithm that starts with an empty set and simply adds elements with the highest marginal utility provides a \(1 - \frac{1}{e}\) approximation guarantee.

Kraus et al (2008) introduced classic cardinality constrained submodular maximization for the first time, however returned a set that was logarithmically larger than \(k\), the cardinality.
Related Work

Orlin et al (2016) could output a set of size $k$ in polynomial time however it was only robust to $o(\sqrt{k})$ elements.

Mirzasoleiman et al (2017) developed a streaming algorithm that was robust to any $d$ elements, however it required massive amounts of memory for $k$ and $d$. 
To identify an \((\alpha, d)\) robust randomized core set for a set \(V\)

A randomized robust core set is a random set \(A \subseteq V\) such that for any \(D \subseteq V\) of size \(|D| \leq d\), there exists a \(B \subseteq A \setminus D, |B| \leq k\) such that

\[
\mathbb{E}[f(B)] \geq \alpha \times \max f(\{S \mid S \subseteq V \setminus D, |S| \leq k\})
\]
An Intuitive Figure Showing WHY Claim

\[ A - B = D, \quad f(B) \geq \alpha \cdot f(S) \]
Proposed Solution

Robust-CoreSet-Centralized

Robust-CoreSet-Streaming

Robust-Distributed
Implementation: Robust-CoreSet-Centralized

1. Select the d + 1 largest element in V and set aside the d+1 largest elements in V into Vt.
2. Set T to the set of (1+e_i) such that (1 + sigma)^i is less than the change in utility of d and greater than the change in utility of d divided by (2 (1 + sigma)k)
3. Set A_t and B_t to the null set for all t in T
4. For all t in T,
   1. while the size of B is greater than d/sigma, add a random element to At from B
      • defining Bt as all e in V such that the value gained by adding e to At is less than (1+e)t but greater than t
5. Set aside all the elements in V not in Bt or At
6. Union Bt with Vt and return it along with A_t as the core set
Implementation: Robust-CoreSet-Streaming

1. Create two sets, $A_t$ and $B_t$
2. All of the elements in $A_t$ having greater than $t$ marginal gains
3. Good enough elements are in $B_t$, which only accepts elements within a certain range of utility. When $B_t$ exceeds a certain size and becomes too big, we pick a random element and add it to $A_t$
   1. This guarantees that the elements being added to $A$ have a similar gain
4. We must then re-compute the marginal gain of the elements in $B_t$
5. This continues until we have $k$ elements in $A$ or until the data stream ends
6. There are at most $d$ elements with marginal gains within the range acceptable to $B_t$
7. The core set is the union of $B_t$ and $A_t$
Implementation: Robust-Distributed

1. First, randomly distribute data onto $m$ machines
2. Each machine runs Robust-Coreset-Centralized as described earlier on its local data
3. After the deletion of set $D$, the central machine runs $m$ instances of Robust-Centralized to find the Solutions $S_i$
4. It also runs the classic greedy on the union of the sets from all the machines to find a solution $T$
5. The best answer is contained in the sets $S$ and $T$

6. BONUS: you can run Robust-CoreSet-Centralized on the output of Robust-Distributed to get an ultra-compact set
Data Summary

- Location Data from publicly available data sets
  - The goal is to find k representative samples from manhattan latitude longitude data
- The Adult Income Dataset
  - Used to test feature deletion for submodular feature selection
- Census1990
  - Used as a large dataset to understand Robust-Distributed performance
Experimental Results

- For experiment 1, the Manhattan location representation experiment, the proposed set of algorithms came up with better representational values and used less memory.

- For experiment 2, predicting adult income data with missing features, the SVM classifier with greedy selected features, had an accuracy of 83%, after deleting race and class sensitive features, the accuracy drops to 79%, when trained on the features found by Robust-Centralized and Robust-Streaming, the performance only dropped to 83.3%.

- Robust-Distributed allows for summarizing a data set of almost 2.5 million into just 4,500 points, robust up until deleting 80% of the items.
Experimental Analysis

- These are the results from the original paper, the data used to generate the graphs was not publicly available.

![Graphs showing performance and memory complexity](image)

(a) Uber dataset: we set $d = 5$ and $k = 20$.  
(b) Uber dataset: we set $d = 5$ and $r = 100$.  
(c) Adult Income: we set $d = 3$

*Figure 1.* (a) The effect of deletion on the performance of algorithms with respect to two different deletion strategies; (b) memory complexity of robust algorithms for different cardinality constraints; (c) The effect of deletion on the performance for feature selection.
§Experimental Analysis

- These are the results from the original paper, the data used to generate the graphs was not publicly available

*Table 2. The comparison of Naive Bayes and SVM classifiers for Adult Income dataset. Ten sensitive features are deleted. The number of stored features is reported in parenthesis.*

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Naive Bayes (Acc.)</th>
<th>SVM (Acc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>All features</td>
<td>0.798</td>
<td>0.830</td>
</tr>
<tr>
<td>GREEDY</td>
<td>0.788</td>
<td>0.796</td>
</tr>
<tr>
<td>GREEDY_D</td>
<td>0.781</td>
<td>0.793</td>
</tr>
<tr>
<td>Rob-Cent</td>
<td>0.781 (22)</td>
<td>0.791</td>
</tr>
<tr>
<td>Rob-Stream</td>
<td>0.781 (29)</td>
<td>0.791</td>
</tr>
<tr>
<td>ROBUST</td>
<td>0.779 (39)</td>
<td>0.788</td>
</tr>
<tr>
<td>STAR-T-GREEDY</td>
<td>0.779 (50)</td>
<td>0.787</td>
</tr>
</tbody>
</table>
These are the results from the original paper, the data used to generate the graphs was not publicly available.

**Figure 2.** Census1990 dataset: ROBUST-DISTRIBUTED versus SG-DISTRIBUTED for four different deleting strategies.
Conclusion and Future Work

• Provided the first scalable and memory efficient algorithms for deletion robust submodular maximization
• They showcased how much powerful the algorithms were in real world scenario for preserving privacy
Challenges in Reproducing Results

1. There were many references to variables in the formulas that were not explained, instantiated, or clarified in other parts of the formula.
2. The dataset preprocessing steps loosely described in the paper.
   1. More later: they claimed to produce 101 binary features from the data, unclear how this is actually possible given the data.
3. I personally didn’t have a lot of background and terminology that the general audience for this paper has.
4. While the paper did offer some intuition for how the concepts worked, they weren’t fully flushed out.
5. Being solo, I didn’t have a team to bounce ideas off of.
6. There is very little to no existing open code or data available for related/similar papers for this particular research problem. So I had to do a lot of the work from complete scratch.
I always able to reproduce experiment 2. Experiment 3 required hardware beyond my means and Experiment 1 and 2 tested the effectiveness of the same algorithms, just on an unwieldy + inconvenient dataset.

Experiment 1 also required implementing several other algorithms that have only been written in research papers. This would mean reproducing two other papers so I didn’t pursue that particular experiment.
I implemented the Robust-Coreset-Centralized (by definition also the Robust Coreset) algorithms

- **SVM Result**
  - lazy greedy feature selection: 79.08%
  - Submodular detection feature selection: 83.39%
  - no feature selection: 83.85%

- **Naïve Bayes Results**
  - lazy greedy feature selection: 78.91%
  - Submodular detection feature selection: 78.89%
  - no feature selection: 78.06%
MY RESULTS: Robust Core Stream - Adult Income Dataset

I implemented the Robust-Coreset-Centralized (by definition also the Robust Coreset) algorithms

- SVM Result
  - lazy greedy feature selection: 78.96%
  - fancy research feature selection: 81.79%
  - no feature selection: 83.92%

- Naïve Bayes Results
  - lazy greedy feature selection: 78.91%
  - Submodular detection feature selection: 77.86%
  - no feature selection: 78.23%
Acknowledgements

Amin Karbasi was supported by a DARPA Young Faculty Award (D16AP00046) and a AFOSR Young Investigator Award (FA9550-18-1-0160). Ehsan Kazemi was supported by the Swiss National Science Foundation (Early Postdoc.Mobility) under grant number 168574.

References


References


References


