Differential Privacy via a Flask REST API

Jason C. Nucciarone
The Pennsylvania State University
College of Information Sciences and Technology
DS 300: Privacy and Security
Instructor: Dr. Sarah Rajtmajer

As the coronavirus pandemic has been raging for almost two years now, we are seeing its ramifications globally with supply chain disruptions. The perfect storm of a worker shortage, port disruptions caused by the coronavirus, and a surge in demand so large that it has been compared to 50 million Americans joining the market, shortages are now a common occurrence across the country [1]. Experts believe that there is currently no end in sight unless consumer spending is cut back, even with the government's current efforts. The Biden administration recently announced that the Port of Los Angeles would be open 24/7 to help increase the throughput of goods entering the country, however, experts believe that this does not adequately address the full issue [2]. To address the whole issue, these experts believe that two things need to happen:

1. Everyone needs to "get on the same page" and work together.
2. The onus to solve the supply chain issue needs to be on private logistics companies, not the government. The government is unable to do anything about the issue in the short-term unless they start regulating and penalizing logistics companies.

How do we get all these companies to work together? A good first step could be spinning up public databases. Logistics companies and data scientists could access this information over the internet via a public application programming interface and use the insights gained from it to position themselves towards areas where demand is the highest. Unfortunately, public databases are closely followed by privacy risks. A privacy risk highlighted in S. Chawla et. al's "Toward Privacy in Public Databases" is that specific records in our database can be a part of an adversaries' auxiliary information. If the adversary has access to this public database, they could successfully identify individuals and entities with the right type of query. A proposed solution to this issue is through database sanitization, but a perfect non-trivial sanitizer does not exist [3].

Instead of trying to sanitize the database - remove all vulnerable records - another approach that could be applied is using differential privacy. Using mechanisms such as the Laplacian mechanism or Gaussian mechanism, we can add some noise to the results of the logistic companies' queries, but we need to pay attention to how much noise is added. As the complexity of the queries go up, so does the amount of noise added [4]. Also, we need to track how many queries each logistics company is sending in. To preserve privacy, we need to limit the amount of times a single entity can query the public database. One such way that has been proposed to preserve privacy is to limit the amount of queries is to implement a continual observational counter; however, a potential shortcoming of this approach is that data science is an iterative process [5]. By exposing our database, without granting direct access to it, we can expect many duplicate queries as the requestor is trying to fine-tune their queries to gain the insights they are looking for. Therefore, we must find a healthy balance where we still preserve the privacy of individuals in the dataset while still allowing the logistics companies and data scientists to tune their queries.

Considering the approaches above, my goal is to spin up a public database that can be accessed by an outside third-party that is protected through differential privacy. The dataset that I am going to be protecting is a Paycheck Protection Program (PPP) dataset provided by the United States Department of the Treasury [6]. I chose this dataset because it provides valuable insights into economic activity of the course of the coronavirus pandemic, and it can be used to continue the successes and provide insights to address shortcomings identified in the program [7, 8]. The dataset itself is rather large, having 53 attributes and 968,557 records (the CSV file is a little less than .5GBs). This larger dataset should be efficiently protected by differential privacy since the noisy results will be more representative of the original result.

The way I am protecting this dataset, while still making it publicly available over a network, is by building a REST application that wraps around the dataset to function as the middle-man between the dataset and the querier [9]. I am using Python and the external libraries *Flask*, *sqlite3*, *numpy*, and *pandas* to serve as the back-end for this application. To launch the web server, all you need to do is have a working installation of Python (greater than 3.5; need support for f-strings) and pip, install the external libraries using the packaged requirements.txt file, and run the python script *dfserver.py*. After launching the server, you can now send Structured Query Language queries over HTTP to the server. When the server receives a query request, here is the process it follows:

1. User submits HTTP GET request containing SQL query.
    a. Example: GET http://localhost:5000/diff/SELECT+COUNT(*)+FROM +PPP
2. Server determines if the user wants differential privacy enabled. Togglable for analysis purposes.
    a. If the URL contains “diff” in the path, activate differential privacy.
    b. If the URL contains “orig” in the path, deactivate differential privacy.
3. Server receives the request and parses the query into correct SQL syntax.
    a. Example: SELECT+COUNT(*)+FROM +PPP -> SELECT COUNT(*) FROM PPP
4. Server calls DBManager class and executes query against SQLite database.
5. Determine if the query returns a single result or crosstable.
6. If differential privacy is active:
    a. Call DFManager class.
    b. Calculate the sensitivity of the SQL query.
    c. Apply Laplacian noise to results.
    d. Increment query count. Max is 100 before the server shuts off access to the SQLite database.
7. Return noisy results to the user as a JSON packet.

The utility that I am trying to get from the differential privacy server is that I can still extract meaningful information from the database. For example, without exposing a single record, I want to know which states received the most PPP loans, or I could want more fine-tuned information and see how many businesses in Pennsylvania received PPP loans. This would be insightful because it could allude to which states are faring better than others, and might give rise to weaknesses in the supply chain in that state. I also might want to see how each city is fairing, where some cities might have more borrowers than others, and see if we could make shipping costs more affordable for that area. My full analysis for each of the queries can be found in my *analysis.ipynb* Jupyter Notebook, but I saw promising results for COUNT and SUM queries, but I did not have good results for average queries. Going forward, for my server to provide more accurate results to counting queries, I believe I will need to implement a better algorithm for calculating the sensitivity and best value of epsilon for an SQL query. The current algorithm that I have for calculating sensitivity does the job, but the SQL queries cannot be too complicated (i.e. subqueries are currently not supported) The way I have the server setup now, epsilon is set to the static value of 0.1, which might not always be the most optimal value. Overall though, I am pleased at how the server is able to communicate over a network, preserve the utility of COUNT and SUM queries, and be able to correctly parse SQL queries from an HTTP message.

Bibliography

[1] G. Kay, "Why the supply chain is in crisis, spurring an 'everything shortage,'" *Business Insider*. https://www.businessinsider.com/why-store-shelves-are-empty-supply-chain-crisis-shortages-2021-10?op=1 (accessed Dec. 05, 2021).

[2] G. Kay, "Why Biden's plan for 24/7 California port operations won't solve the supply-chain crisis," *Business Insider*. https://www.businessinsider.com/why-bidens-plan-us-ports-wont-solve-supply-chain-crisis-2021-10 (accessed Dec. 06, 2021).

[3] S. Chawla, C. Dwork, F. McSherry, A. Smith, H. Wee, "Toward Privacy in Public Databases." In: Proceedings of the 2nd Theory of Cryptography Conference. 2005.

[4] C. Dwork, "Differential Privacy: A Survey of Results." In: Agrawal M., Du D., Duan Z., Li A. (eds) Theory and Applications of Models of Computation. TAMC 2008. Lecture Notes in Computer Science, vol 4978. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-79228-4_1. 2008.

[5] C. Dwork, M. Naor, T. Pitassi, G. N. Rothblum. "Differential privacy under continual observation." In Proceedings of the forty-second ACM symposium on Theory of computing (STOC '10). *Association for Computing Machinery, New York, NY, USA*, 715–724. DOI:https://doi.org/10.1145/1806689.1806787. 2010.

[6] "Paycheck Protection Program | U.S. Department of the Treasury," *home.treasury.gov*. https://home.treasury.gov/policy-issues/coronavirus/assistance-for-small-businesses/paycheck-protection-program.

[7] P. Erickson, "PPP: What Worked, What Didn't, and What Needs Tweaking," *www.industryweek.com*, Jul. 13, 2020. https://www.industryweek.com/supply-chain/supply-chain-initiative/article/21136517/ppp-what-worked-what-didnt-and-what-needs-tweaking (accessed Dec. 05, 2021).

[8] S. Agarwal, B. W. Ambrose, L. Lopez, X. Xiao, "Did the Paycheck Protection Program Help Small Businesses? Evidence from Commercial Mortgage-backed Securities." Available at SSRN: https://ssrn.com/abstract=3674960 or http://dx.doi.org/10.2139/ssrn.3674960. 2021.

[9] R. Fielding, “Architectural styles and the design of network-based software architectures,” *University of California, Irvine*. Chapter 5. 2000.