pyPANTERA: A Python PAckage for Natural language obfuscaTion Enforcing pRivacy & Anonymization

Francesco Luigi De Faveri*
University of Padova
Padova, Italy
francescoluigi.defaveri@phd.unipd.it

Guglielmo Faggioli
University of Padova
Padova, Italy
guglielmo.faggioli@unipd.it

Nicola Ferro
University of Padova
Padova, Italy
ferro@dei.unipd.it

Abstract

Privacy is critical when dealing with user-generated text, as common in Natural Language Processing (NLP) and Information Retrieval (IR) tasks. Documents, queries, posts, and reviews might pose a risk of inadvertently disclosing sensitive information. Such exposure of private data is a significant threat to user privacy, as it may reveal information that users prefer to keep confidential. The leading framework to protect user privacy when handling textual information is represented by the ε-Differential Privacy (DP). However, the research community lacks a unified framework for comparing different DP mechanisms. This study introduces pyPANTERA, an open-source Python package developed for text obfuscation. The package is designed to incorporate State-of-the-Art DP mechanisms within a unified framework for obfuscating data. pyPANTERA is not only designed as a modular and extensible library for enriching DP techniques, thereby enabling the integration of new DP mechanisms in future research, but also to allow reproducible comparison of the current State-of-the-Art mechanisms. Through extensive evaluation, we demonstrate the effectiveness of pyPANTERA, making it an essential resource for privacy researchers and practitioners. The source code of the library and for the experiments is available at: https://github.com/Kekkodf/pypantera.

CCS Concepts

• **Security and privacy → Privacy protections**; *Usability in security and privacy*; Social aspects of security and privacy.

Keywords

Privacy Preserving Mechanisms, Differential Privacy, NLP, Information Retrieval, Security

ACM Reference Format:
Francesco Luigi De Faveri, Guglielmo Faggioli, and Nicola Ferro. 2024. pyPANTERA: A Python PAckage for Natural language obfuscaTion Enforcing pRivacy & Anonymization. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3627673.3679173

[*] Corresponding Author.

This work is licensed under a Creative Commons Attribution International 4.0 License.

CIKM '24, October 21–25, 2024, Boise, ID, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679173

1 Introduction

Natural Language Processing (NLP) and Information Retrieval (IR) systems are commonly trained on and applied to pieces of text (e.g., queries, documents, reviews, posts) that contain sensitive and personal user information and pose privacy risks. Consider, for example, the queries a user submits to a search engine or the textual content they post on a social network. Such pieces of text might contain personally identifiable information (e.g., the name or address of the searcher) or details about the user's personal sphere (e.g., political views, sexual orientation) that might expose them to blackmailing and cyberbullying [8] or even endanger their safety in illiberal countries [21, 23]. Therefore, privacy literature [4, 11, 14, 15, 37] has stressed the importance of privacy for textual data analysis proposing new methods for text obfuscation. In such a context, Differential Privacy (DP) represents the leading framework to provide user-level privacy guarantees. The DP framework was designed to provide the "Plausible Deniability" property, i.e., the outcome of any analysis is statistically indistinguishable within a given privacy budget. A current state-of-the-art limitation is that such approaches have been tested on different tasks and datasets, and never organized in a unified framework for text obfuscation in NLP and IR. For this reason, privacy practitioners can benefit from a unified, modular and flexible framework to enable a rapid design of new DP approaches and allow a uniform and fast evaluation with the state-of-the-art on multiple tasks.

In this work, we present pyPANTERA, an open-source unified, flexible and user-friendly framework for DP mechanisms implementation and comparison. Moreover, we bring together state-of-the-art mechanisms [5, 7, 12, 32, 33, 35] used for NLP and IR tasks. pyPANTERA is organized into modules that implement different families of obfuscation mechanisms, namely sampling and embedding perturbation approaches, and a module for evaluation. The former provides different interfaces for families of mechanisms and ensures consistent implementation of new algorithms alongside existing ones. On the other hand, the latter allows the practitioner to evaluate the privacy of the resulting obfuscated text, measuring the similarity between original and obfuscated sentences.

Finally, to showcase the library's potential, we extensively compare the implemented mechanisms, enforcing their use in real NLP and IR tasks, showing that the results achieved are comparable to those found in the original mechanism studies. This underscores the capability of pyPANTERA as an essential tool for privacy practitioners to implement future obfuscation techniques and replicate previous study results effectively. The code is open source under the GNU GPL 3.0 license.

The study is structured as follows: Section 2 describes other related works related to pyPANTERA; moreover, Section 3 describes the design of the Python package, providing technical information about the resource, and finally Section 4 describes the results obtained from the tasks performed to evaluate the overall framework.

2 Related Works

Background and DP approaches. The gold-standard definition of formal privacy is represented by the notion of ε-DP [10]. A DP mechanism $\mathcal{M}$, i.e., an algorithm that takes an input and produces a noisy output, is designed to ensure sensitive data privacy depending on a privacy budget ε. Formally, a mechanism $\mathcal{M}$ satisfies ε-DP if, for any pair of neighbouring datasets D, D', i.e., datasets that differ at most for only one record, and a privacy budget $\varepsilon \in \mathbb{R}^+$, the condition $\Pr\{\mathcal{M}(D) \in \mathcal{S}\} \leq e^{\varepsilon}\Pr\{\mathcal{M}(D') \in \mathcal{S}\}$, $\forall \mathcal{S} \subseteq \text{Image}(\mathcal{M})$ is verified. This provides the property of "plausible deniability" for a user: the adversary cannot determine with absolute certainty which input, i.e., the user's original data, corresponds to a given output. Metric-DP, introduced in [6], represents a DP relaxation of the initial definition applied to metric spaces. Metric-DP ensures that a randomized mechanism $\mathcal{M} : \mathbb{R}^p \to \mathbb{R}^p$ defined over a geometric space with distance function $d : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}^+$ respects the definition of DP, iff, for any three points $w, w', \hat{w} \in \mathbb{R}^p$, the inequality $\Pr\{\mathcal{M}(w) = \hat{w}\} \leq e^{\varepsilon d(w,w')}\Pr\{\mathcal{M}(w') = \hat{w}\}$ is respected.

In recent years [20, 37], obfuscation mechanisms for natural language texts have received significant attention, particularly through adopting ε-DP [10] as a formal framework for designing these mechanisms. Specifically for these kinds of obfuscation mechanisms, it is possible to differentiate the mechanisms based on the type of obfuscation perturbation that is performed on the texts: on the one hand, mechanisms proposed in [12, 32, 33], obfuscates the embeddings of the terms in the sentence by adding statistical noise based on the privacy budget ε. On the other hand, the mechanisms introduced in [5, 7, 35] are based on the initial computation of a score between word embeddings to rank similar terms, followed by the use of ε to adjust the probability of sampling the new obfuscated words.

DP tools. Although unified libraries for obfuscating texts do not exist currently, several endeavours are available for structured tabular data. Such libraries primarily facilitate the implementation of private statistical interrogation and private Machine Learning pipeline, e.g., computing the Differentially Private Stochastic Gradient Descent [1, 3]. Accordingly to the evaluation proposed in [24, 36], examples of such libraries are represented by the IBM Diffprivlib [17], the Meta PyTorch Opacus [34], and Google TensorFlow DP [29] toolkit. Moreover, built as a forked project of the Google TensorFlow DP and OpenDP [13] projects, OpenMined released PyDP [26], a wrapper version library in Python, that can be used for aggregating sensitive statistics over numeric datasets.

Text sanitization and anonymization are another important privacy aspect in NLP. Microsoft Presidio [22] is designed upon the SpaCy [18] library and comprises an Analyzer and an Anonymizer to identify and mask personally identifiable information within a given sentence. The analyzer employs regex rules and Named Entity Recognition Machine Learning models provided by SpaCy to identify sensitive terms within the context provided. Afterwards, Presidio utilizes the anonymization module to obfuscate by redacting, hashing, or replacing the identified sensitive information, producing an obfuscated rendition of the original text. Despite both Presidio and pyPANTERA work on textual data, they consider different privacy aspects, i.e., data sanitization and obfuscation. Therefore, the two resources are complementary. As a future direction, we plan to combine Presidio and pyPANTERA: the intention is to integrate Presidio data sanitization with the semantic obfuscation capabilities of pyPANTERA in the sentence obfuscation pipeline.

3 pyPANTERA

3.1 Pipeline

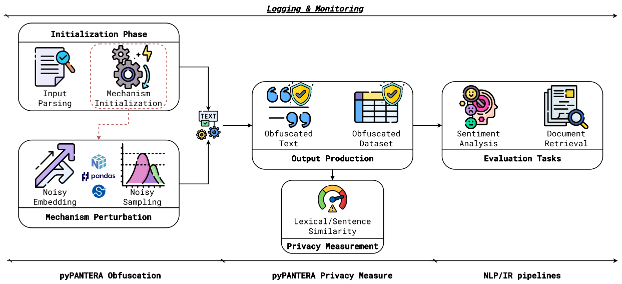

Figure 1: Pipeline of the pyPANTERA library.

Figure 1 reports the pipeline of how the text obfuscation is performed in pyPANTERA. Initially, the input data are tokenized and parsed to remove punctuations. Upon receiving the necessary parameters from the practitioner, the mechanism is initialized, and thus, it incorporates methods for obfuscation based on the family of DP methods. This defines the techniques involving noisy perturbation of the embeddings or noisy sampling of the obfuscated terms. The tokenized text is finally obfuscated, generating the required number of obfuscation variants within a single text or a suitable data frame. With these new sentences, the NLP and IR tasks can be performed. In addition, pyPANTERA provides a module to perform privacy measurements to assess the level of privacy obtained.

3.2 Development Workflow

pyPANTERA is written in Python (version 3.10) and requires Python ≥ 3.7 to run. The Python language was selected due to its accessibility, fast prototyping and active user community. Furthermore, as the tasks for which the obfuscation mechanisms are implemented rely on deep learning methods, ensuring rapid interoperability between obfuscation and the overall pipeline significantly enhances the efficiency of conducting experiments. The library can be installed and used in two ways: by cloning the repository of the resource available in GitHub[1], the README provides detailed instructions for setting up the virtual environment for conducting obfuscation and analysis. Otherwise, pyPANTERA can be installed using `pip` to download the package from PyPI[2], using `pip install pypantera`.

A fundamental characteristic of pyPANTERA is that it is accessible to privacy practitioners of all expertise. To achieve this, pyPANTERA constructs upon popular data science libraries, i.e., Numpy [16], Pandas [31], and SciPy [30]. In addition, to optimize

[1] https://github.com/Kekkodf/pypantera
[2] https://pypi.org/

large amounts of text obfuscation, the library supports parallel computing with the Python library `multiprocessing`[3].

Mechanisms Overview. The flexibility for new mechanisms implementation relies on the abstract classes offered in pyPANTERA. The library implements one parent abstract DP mechanism class that handles the initialization of new mechanisms. Furthermore, each specific obfuscation process is defined by each child abstract class of the embedding and sampling perturbation. We make the UML diagram available on the project repository. Here we report a list of the state-of-the-art mechanisms that have been implemented. For detailed information regarding the obfuscation of each mechanism, we recommend referring to the original studies.

- Cumulative Multivariate Perturbation Mechanism (CMP) [12]: Noise sampled from an n - dimensional Laplace distribution.
- Mahalanobis (Mhl) [32]: Noise sampled from an n - dimensional Normal distribution proportional to the λ regularized Mahalanobis norm of the term embedding.
- Vickrey CMP Mechanism (VickreyCMP) and Vickrey Mahalanobis Mechanism (VickreyMhl) [33]: The noise is sampled as defined by the parent method (CMP or Mhl) and the obfuscation term is selected based on a free parameter t.
- Custumized Text Mechansim (CusText) [7]: Sampling new terms is bounded to k possible terms selected using the scores computed using the distances among word embeddings.
- Sanitization Text Mechanism (SanText) [35]: Sampling of the new term is computed with a score based on the distances among embeddings, with terms closer to the word being obfuscated having a higher probability of selection.
- Truncated Exponential Mechanism (TEM) [5]: Noise sampled from an n - dimensional Gumbel distribution is added to the score computed based on the distances between the vector embeddings and the final obfuscation term is sampled accordingly to the maximum noisy score. The truncation is computed using the free parameter β provided.

The mechanisms have been categorized into *Embedding* (CMP, Mhl, VickreyCMP, VickreyMhl) and *Sampling* (CusText, SanText, TEM) perturbation groups to delineate the type of obfuscation process they perform. More details can be found in the repository.

Functionalities. pyPANTERA implements several utility functions to help the practitioner get a comprehensive view of all the pipeline steps. pyPANTERA offers an appropriate class to speed up the initialization of the embedding vocabulary that uses parallelization to read the embeddings from the provided file. Moreover, using the `logging` python library[4] the method creates a folder containing a logger to report all the information of the steps performed in the pipeline. Finally, to evaluate the similarities between the original and obfuscated texts, pyPANTERA implements the Jaccard Index to compute the overlapping terms (the lexical similarity) and a cosine similarity among the contextual embeddings of the sentences (the sentence similarity). Even in this case, pyPANTERA offers the possibility of designing and implementing new metric functions to evaluate the similarity between original and privatized texts.

4 Experimental Evaluation

To assess the pyPANTERA effectiveness, we enforced the tasks employed in the original studies of the mechanisms, i.e., sentiment analysis classification, and document retrieval. Finally, we used the metrics methods implemented in the library to estimate the levels of privacy provided by the pyPANTERA library mechanisms.

4.1 Dataset and Experimental Setup

For the experiments performed to assess the correctness and the effectiveness of the pyPANTERA library, we propose similar NLP tasks performed in the original state-of-the-art mechanisms papers, i.e., sentiment analysis. Moreover, as the study in [11] suggested, we also tested the robustness of the library in implementing the obfuscation pipeline for an IR task while protecting user privacy. As datasets, we tested the sentiment analysis on the Kaggle Twitter sentiment analysis[5] test set, and the document retrieval from the TREC Deep Learning (DL'19) [9], based on the MSMARCO [25] passage corpus. In conclusion, we measure the privacy levels of the former query collection using the metrics module in pyPANTERA. To avoid encumbering, the details about the systems used to perform the sentiment analysis and the document retrieval are provided in the respective sections, and the parameters configuration of the mechanisms used during the obfuscation process is reported in the GitHub repository of the project. As per the embeddings used for all the tasks, we adopted the pre-trained vector embeddings[6] of GloVe [27], from Wikipedia 2014 and Gigaword 5.

4.2 Natural Language Processing

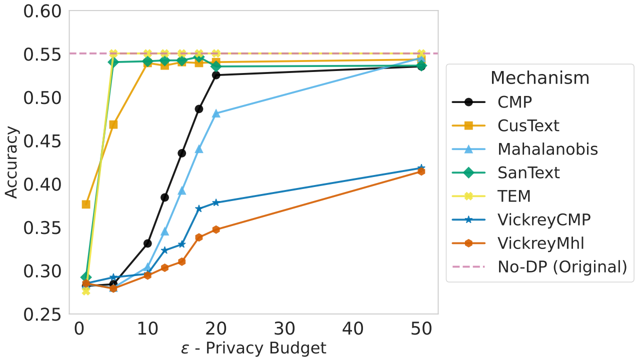

Figure 2: Mean Accuracy of the Sentiment Analysis task using the TweetNLP model [2], varying the privacy budget ε for the different mechanisms implemented in pyPANTERA.

To showcase the potentiality of pyPANTERA, we performed a traditional NLP task, i.e., sentiment analysis on posts obtained from Twitter. For the analysis of tweet sentiment, we employed the Twitter-roBERTa-base for sentiment analysis, also known as TweetNLP [2], as the model to extract sentiment from the parsed version of tweets in the dataset. As a performance measure, we calculated the accuracy in identifying the correct labels of the tweets,

[3] https://docs.python.org/3/library/multiprocessing.html
[4] https://docs.python.org/3/library/logging.html
[5] https://www.kaggle.com/datasets/jp797498e/twitter-entity-sentiment-analysis/data
[6] https://nlp.stanford.edu/projects/glove/

Table 1: Average Recall and nDCG@10 on the MSMARCO dl'19 collection [9] using the obfuscated queries for the searching process, and the original version for the reranking. The searching and the reranking process was performed using Contriver [19].

		Recall									**nDCG@10**								
		ε - Privacy Budget									ε - Privacy Budget								
Perturbation	**Mechanism**	1.0	5.0	10.0	12.5	15.0	17.5	20.0	50.0		1.0	5.0	10.0	12.5	15.0	17.5	20.0	50.0	
Embedding	CMP	0.000	0.000	0.028	0.174	0.292	0.403	0.430	0.444		0.000	0.000	0.052	0.277	0.544	0.546	0.535	0.564	
	Mahalanobis	0.000	0.000	0.001	0.077	0.134	0.290	0.368	0.447		0.000	0.000	0.003	0.103	0.262	0.455	0.494	0.565	
	VickreyCMP	0.000	0.000	0.020	0.016	0.048	0.053	0.165	0.235		0.000	0.000	0.031	0.016	0.166	0.159	0.221	0.372	
	VickreyMhl	0.000	0.001	0.002	0.002	0.029	0.042	0.122	0.191		0.000	0.005	0.007	0.004	0.062	0.097	0.158	0.293	
Sampling	CusText	0.053	0.245	0.430	0.442	0.444	0.443	0.443	0.443		0.143	0.439	0.576	0.571	0.569	0.569	0.569	0.569	
	SanText	0.000	0.444	0.447	0.448	0.444	0.450	0.447	0.444		0.000	0.564	0.569	0.570	0.568	0.559	0.568	0.562	
	TEM	0.000	0.498	0.498	0.498	0.498	0.498	0.498	0.498		0.000	0.636	0.636	0.636	0.636	0.636	0.636	0.636	
None	Original	-	-	-	-	-	-	-	-	0.498	-	-	-	-	-	-	-	-	0.636

Table 2: Average Lexical and Sentence similarity between the original and obfuscated queries of the MSMARCO dl'19 collection [9]. Lexical and Semantic similarity are computed using the implemented `metrics` module in pyPANTERA.

		Lexical Similarity *(Jaccard Similarity)*								**Semantic Similarity** *(MiniLM [28])*							
		ε - Privacy Budget								ε - Privacy Budget							
Perturbation	**Mechanism**	1.0	5.0	10.0	12.5	15.0	17.5	20.0	50.0	1.0	5.0	10.0	12.5	15.0	17.5	20.0	50.0
Embedding	CMP	0.000	0.000	0.119	0.274	0.460	0.735	0.785	0.935	0.025	0.037	0.225	0.429	0.628	0.836	0.847	0.902
	Mahalanobis	0.000	0.002	0.047	0.140	0.302	0.457	0.590	0.935	0.016	0.027	0.088	0.242	0.435	0.587	0.730	0.908
	VickreyCMP	0.000	0.000	0.039	0.061	0.180	0.191	0.164	0.212	0.018	0.045	0.103	0.169	0.348	0.382	0.435	0.596
	VickreyMhl	0.000	0.013	0.028	0.038	0.098	0.134	0.117	0.151	0.037	0.030	0.078	0.109	0.202	0.264	0.303	0.498
Sampling	CusText	0.089	0.374	0.816	0.880	0.925	0.929	0.929	0.935	0.357	0.627	0.881	0.900	0.908	0.908	0.909	0.910
	SanText	0.000	0.935	0.935	0.935	0.935	0.935	0.935	0.935	0.031	0.902	0.906	0.910	0.917	0.900	0.902	0.907
	TEM	0.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.037	1.000	1.000	1.000	1.000	1.000	1.000	1.000

replicating the task conducted by other obfuscation mechanisms studies [5, 7, 35]. The objective of the task was to show how the results of the obfuscation obtained by pyPANTERA, once obtained, can be easily used in a simple NLP task to compare how different obfuscation techniques influence the performance of a model. Figure 2 shows the results of the Accuracy *vs.* ε - Privacy Budget for different mechanisms. The results obtained are consistent with those reported in the proposed papers [5, 7, 35]. For instance, the TEM mechanism outperforms the CMP mechanism in sentiment classification, verifying the results obtained by Carvalho et al. [5]. Additionally, CusText demonstrates higher performance than SanText, as reported by Chen et al. [7].

4.3 Information Retrieval and Privacy Analysis

Following the experimental pipeline proposed by Faggioli and Ferro [11], to protect user privacy during document retrieval, we employed the obfuscated MSMARCO DL'19 queries to search for relevant documents within the collection. After that, we re-ranked the search results using the original version. As a retrieval system and re-ranker, we used the Meta Contriever system [19]. Table 1 reports the Recall and nDCG@10 of the retrieval pipeline.

On the other hand, Table 2 reports the results of the similarity computed to assess the similarities between original and obfuscated DL'19 queries. Specifically, we analyzed two types of similarities using the metric functions in pyPANTERA: the lexical similarity, using the Jaccard function, and the sentence similarity, shown as the cosine similarity computed using the contextual embeddings of the queries calculated by the Sentence-BERT model MiniLM [28].

As observed by Faggioli and Ferro [11], and in line with the expectations of a DP mechanism, an increase in the privacy budget ε corresponds to improved performance of the mechanism but lower privacy guarantees, cf. Table 1 and Table 2. In addition, the *Sampling* Perturbation mechanisms tend to exhibit greater similarity for lower values of ε in comparison to the *Embedding* perturbation family of mechanisms. We leave this as an open issue for further analysis.

5 Conclusions

Privacy remains an important research area for NLP and IR tasks. In this study, we presented pyPANTERA, a flexible framework to compare different DP mechanisms designed to protect user privacy during textual analysis. pyPANTERA contributes to the privacy-preserving community by enforcing a clear and user-friendly text obfuscation pipeline that helps other practitioners design and implement new obfuscation mechanisms. The library offers different functionalities for monitoring the process status and different metric functions to evaluate the text privacy provided. Moreover, a comprehensive analysis of standard NLP and IR showcased the general obfuscation process, enlightening the effectiveness of pyPANTERA to compare different obfuscation approaches in a unified and extensible framework. As a future direction, we plan to increase the number of obfuscation mechanisms available and expand the privacy evaluation module with new metric functions for assessing the privacy level provided.

References

[1] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi (Eds.). ACM, 308–318. https://doi.org/10.1145/2976749.2978318

[2] Jose Camacho-collados, Kiamehr Rezaee, Talayeh Riahi, Asahi Ushio, Daniel Loureiro, Dimosthenis Antypas, Joanne Boisson, Luis Espinosa Anke, Fangyu Liu, and Eugenio Martínez Cámara. 2022. TweetNLP: Cutting-Edge Natural Language Processing for Social Media. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Wanxiang Che and Ekaterina Shutova (Eds.). Association for Computational Linguistics, Abu Dhabi, UAE, 38–49. https://doi.org/10.18653/v1/2022.emnlp-demos.5

[3] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX Conference on Security Symposium* (Santa Clara, CA, USA) *(SEC'19)*. USENIX Association, USA, 267–284.

[4] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting Training Data from Large Language Models. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 2633–2650. https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting

[5] Ricardo Silva Carvalho, Theodore Vasiloudis, Oluwaseyi Feyisetan, and Ke Wang. "2023". *TEM: High Utility Metric Differential Privacy on Text*. 883–890. https://doi.org/10.1137/1.9781611977653.ch99 arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781611977653.ch99

[6] Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. 2013. Broadening the Scope of Differential Privacy Using Metrics. In *Privacy Enhancing Technologies*, Emiliano De Cristofaro and Matthew Wright (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 82–102.

[7] Sai Chen, Fengran Mo, Yanhao Wang, Cen Chen, Jian-Yun Nie, Chengyu Wang, and Jamie Cui. 2023. A Customized Text Sanitization Mechanism with Differential Privacy. In *Findings of the Association for Computational Linguistics: ACL 2023*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 5747–5758. https://doi.org/10.18653/v1/2023.findings-acl.355

[8] Naganna Chetty and Sreejith Alathur. 2018. Hate speech review in the context of online social networks. *Aggression and Violent Behavior* 40 (2018), 108–118. https://doi.org/10.1016/j.avb.2018.05.003

[9] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M. Voorhees. 2020. Overview of the TREC 2019 deep learning track. *CoRR* abs/2003.07820 (2020). arXiv:2003.07820 https://arxiv.org/abs/2003.07820

[10] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography*, Shai Halevi and Tal Rabin (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 265–284.

[11] Guglielmo Faggioli and Nicola Ferro. 2024. Query Obfuscation for Information Retrieval Through Differential Privacy. In *Advances in Information Retrieval*, Nazli Goharian, Nicola Tonellotto, Yulan He, Aldo Lipani, Graham McDonald, Craig Macdonald, and Iadh Ounis (Eds.). Springer Nature Switzerland, Cham, 278–294.

[12] Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. 2020. Privacy- and Utility-Preserving Textual Analysis via Calibrated Multivariate Perturbations. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (Houston, TX, USA) *(WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 178–186. https://doi.org/10.1145/3336191.3371856

[13] Marco Gaboardi, Michael Hay, and Salil Vadhan. 2020. A programming framework for opendp. *Manuscript, May* (2020).

[14] Ivan Habernal. 2021. When differential privacy meets NLP: The devil is in the detail. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 1522–1528. https://doi.org/10.18653/V1/2021.EMNLP-MAIN.114

[15] Ivan Habernal, Fatemehsadat Mireshghallah, Patricia Thaine, Sepideh Ghanavati, and Oluwaseyi Feyisetan. 2023. Privacy-Preserving Natural Language Processing. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, Fabio Massimo Zanzotto and Sameer Pradhan (Eds.). Association for Computational Linguistics, Dubrovnik, Croatia, 27–30. https://doi.org/10.18653/v1/2023.eacl-tutorials.6

[16] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature* 585, 7825 (Sept. 2020), 357–362. https://doi.org/10.1038/s41586-020-2649-2

[17] Naoise Holohan, Stefano Braghin, Pól Mac Aonghusa, and Killian Levacher. 2019. Diffprivlib: the IBM differential privacy library. *ArXiv e-prints* 1907.02444 [cs.CR] (July 2019).

[18] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python. (2020). https://doi.org/10.5281/zenodo.1212303

[19] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *Trans. Mach. Learn. Res.* 2022 (2022). https://openreview.net/forum?id=jKN1pXi7b0

[20] Oleksandra Klymenko, Stephen Meisenbacher, and Florian Matthes. 2022. Differential Privacy in Natural Language Processing The Story So Far. In *Proceedings of the Fourth Workshop on Privacy in Natural Language Processing*, Oluwaseyi Feyisetan, Sepideh Ghanavati, Patricia Thaine, Ivan Habernal, and Fatemehsadat Mireshghallah (Eds.). Association for Computational Linguistics, Seattle, United States, 1–11. https://doi.org/10.18653/v1/2022.privatenlp-1.1

[21] Huyen Le, Raven Maragh, Brian Ekdale, Andrew High, Timothy Havens, and Zubair Shafiq. 2019. Measuring Political Personalization of Google News Search. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 2957–2963. https://doi.org/10.1145/3308558.3313682

[22] Omri Mendels, Coby Peled, Nava Vaisman Levy, Tomer Rosenthal, Limor Lahiani, et al. 2018. Microsoft Presidio: Context aware, pluggable and customizable PII anonymization service for text and images. https://microsoft.github.io/presidio

[23] Eni Mustafaraj, Emma Lurie, and Claire Devine. 2020. The Case for Voter-Centered Audits of Search Engines during Political Elections. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (Barcelona, Spain) *(FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 559–569. https://doi.org/10.1145/3351095.3372835

[24] Ivoline C. Ngong, Brad Stenger, Joseph P. Near, and Yuanyuan Feng. 2024. Evaluating the Usability of Differential Privacy Tools with Data Practitioners. arXiv:2309.13506 [cs.HC]

[25] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016 (CEUR Workshop Proceedings, Vol. 1773)*, Tarek Richard Besold, Antoine Bordes, Artur S. d'Avila Garcez, and Greg Wayne (Eds.). CEUR-WS.org. https://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf

[26] OpenMinded. 2021. PyDP. https://github.com/OpenMined/PyDP Accessed: 2024.

[27] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[28] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084

[29] Pranav Subramani, Nicholas Vadivelu, and Gautam Kamath. 2021. Enabling Fast Differentially Private SGD via Just-in-Time Compilation and Vectorization. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (Eds.). 26409–26421. https://proceedings.neurips.cc/paper/2021/hash/ddf9029977a61241841edeae15e9b53f-Abstract.html

[30] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[31] Wes McKinney. 2010. Data Structures for Statistical Computing in Python. In *Proceedings of the 9th Python in Science Conference*, Stéfan van der Walt and Jarrod Millman (Eds.). 56 – 61. https://doi.org/10.25080/Majora-92bf1922-00a

[32] Zekun Xu, Abhinav Aggarwal, Oluwaseyi Feyisetan, and Nathanael Teissier. 2020. A Differentially Private Text Perturbation Method Using Regularized Mahalanobis Metric. In *Proceedings of the Second Workshop on Privacy in NLP*, Oluwaseyi Feyisetan, Sepideh Ghanavati, Shervin Malmasi, and Patricia Thaine

(Eds.). Association for Computational Linguistics, Online, 7–17. https://doi.org/10.18653/v1/2020.privatenlp-1.2

[33] Zekun Xu, Abhinav Aggarwal, Oluwaseyi Feyisetan, and Nathanael Teissier. 2021. On a Utilitarian Approach to Privacy Preserving Text Generation. In *Proceedings of the Third Workshop on Privacy in Natural Language Processing*, Oluwaseyi Feyisetan, Sepideh Ghanavati, Shervin Malmasi, and Patricia Thaine (Eds.). Association for Computational Linguistics, Online, 11–20. https://doi.org/10.18653/v1/2021.privatenlp-1.2

[34] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. 2021. Opacus: User-Friendly Differential Privacy Library in PyTorch. *CoRR* abs/2109.12298 (2021). arXiv:2109.12298 https://arxiv.org/abs/2109.12298

[35] Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. 2021. Differential Privacy for Text Analytics via Natural Text Sanitization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, Online, 3853–3866. https://doi.org/10.18653/v1/2021.findings-acl.337

[36] Shiliang Zhang, Anton Hagermalm, Sanjin Slavnic, Elad Michael Schiller, and Magnus Almgren. 2023. Evaluation of Open-Source Tools for Differential Privacy. *Sensors* 23, 14 (2023), 6509. https://doi.org/10.3390/S23146509

[37] Ying Zhao and Jinjun Chen. 2022. A Survey on Differential Privacy for Unstructured Data Content. *ACM Comput. Surv.* 54, 10s, Article 207 (sep 2022), 28 pages. https://doi.org/10.1145/3490237