Privacy-Preserving Sentiment Analysis on Twitter

Felix Vogel[1], Lucas Lange[2]

Abstract: Sentiment analysis is a crucial tool to evaluate customer opinion on products and services. However, analyzing social media data raises concerns about privacy violations since users may share sensitive information in their posts. In this work, we propose a privacy-preserving approach for sentiment analysis on Twitter data using Differential Privacy (DP). We first implement a non-private baseline model and assess the impact of various settings and preprocessing methods. We then extend this approach with DP under multiple privacy parameters $\varepsilon = \{0.1, 1, 10\}$ and finally evaluate the usability of the resulting private models. Our results show that DP models can maintain high accuracy for the studied task. We contribute to the development of privacy-preserving machine learning for customer opinion analysis and provide insights into trade-offs between privacy and utility. The proposed approach helps protect sensitive information while still allowing for valuable insights to be gained from social media data.

Keywords: Differential Privacy; Sentiment Analysis; Preprocessing; Twitter; Privacy-Preserving Machine Learning

1 Introduction

In a world where customer opinions about products are predominantly shared through social media posts, sentiment analysis is a popular research area that aims to automatically identify and extract subjective information from text data. Social media platforms such as Twitter have become important sources of user-generated content that can be analyzed for sentiment [Ne12]. However, analyzing social media data raises concerns about privacy violations since users may share sensitive information in their posts [MSK11]. Other than on Twitter itself, users only have limited options for deleting their data from machine learning models. According to the EU GDPR however, there should be a 'right to be forgotten' [EC16]. Through implementing privacy-preserving machine learning strategies we want to minimize memorization by keeping the model increasingly agnostic to individual user data.

Differential Privacy (DP) [Dw06] is a framework for protecting sensitive information while still allowing useful insights to be gained from data. DP adds noise to the output of an algorithm in order to prevent an attacker from inferring sensitive information about individual users. Using this framework, individual user data can be processed while giving a provable privacy guarantee for each data point. A reliable privacy promise when processing their data could improve trust when asking customers to share personal feedback.

[1] Leipzig University, Germany, nx27rixe@studserv.uni-leipzig.de
[2] Leipzig University, Germany, ll95wyqa@studserv.uni-leipzig.de

doi:10.18420/SKILL2023_19

We are the first to evaluate an approach for preserving privacy in sentiment analysis on Twitter using DP. We first implement a non-private sentiment analysis as a baseline model and compare different preprocessing methods to find the best strategy. We then apply DP using different privacy levels of $\varepsilon = \{0.1, 1, 10\}$. Finally, we evaluate the usability of the resulting model under the influence of such privacy preservation.

In the upcoming Sect. 2 we try to create a common ground regarding the fundamental concepts for this work. Sect. 3 then provides an overview of closely related works in the field of sentiment analysis. Our applied methods and experiments are summarized in Sect. 4, with the following Sect. 5 presenting their results. We finally give conclusive thoughts and a brief outlook into possible future ventures in Sect. 6.

2 Background

In this section we provide basic understanding for different ideas this work is building on.

2.1 Sentiment Analysis

Sentiment analysis is a subfield of natural language processing that aims to automatically identify the emotional tone of a piece of text, such as a review, tweet, or news article [Ca17]. The task involves classifying the text as positive, negative, or neutral, based on the underlying sentiment expressed by the author. Sentiment analysis can be performed using a variety of techniques, including rule-based methods, machine learning algorithms, and deep learning models. The field has numerous applications, such as social media monitoring, brand reputation management, and market research, and has received increasing attention in recent years due to the growing importance of understanding and leveraging user-generated content. However, the accuracy of sentiment analysis models can be affected by various factors, such as the quality of the training data, the complexity of the language, and the presence of sarcasm or irony.

2.2 Differential Privacy

The concept of DP [Dw06] is a mathematical definition of privacy that ensures that the inclusion or exclusion of any individual's data in a dataset does not significantly change the output of statistical queries on that dataset. The framework is based on the idea of adding noise to the queries to protect individual privacy while preserving the utility of the dataset for analysis.

Formally, an algorithm A training on a set S is called (ε,δ)-differentially-private, if for all datasets D and D' that differ by exactly one record:

$$Pr[A(D) \in S] \leq e^{\varepsilon} Pr[A(D') \in S] + \delta \tag{1}$$

The ε-parameter measures the level of privacy protection provided by the mechanism. It thus determines the amount of random noise that is added to the output of queries on the dataset to protect individual privacy. The smaller the value of ε, the stronger the privacy protection, but also the greater the amount of noise.

2.3 Differentially Private Stochastic Gradient Descent

Differentially Private Stochastic Gradient Descent (DP-SGD) [Ab16] is a variant of the stochastic gradient descent optimization algorithm that provides differential privacy guarantees. DP-SGD works by adding controlled noise to the gradients computed on each mini-batch of data during the training process. The amount of noise added is controlled by a privacy parameter ε, which determines the strength of privacy protection. Tuning the privacy parameter can be challenging, and careful selection and calibration of the noise level are required to achieve the desired privacy-utility trade-off.

3 Related Work

Main point of reference for our baseline is the work by Go et al. [GBH09], which also introduced our used sentiment140 Twitter dataset. In addition to the data, they also proposed preprocessing techniques and evaluated different non-private machine learning models for sentiment analysis. Their Support Vector Machine (SVM) model performed best in the unigrams setting and achieved 82.2% accuracy. The other tested models are Multinomial Naive Bayes (MNB) and Logistic Regression (LR)[3], which achieved 81.3% and 80.5%, respectively. In terms of the many possible preprocessing steps and their combinations involved when working with text data, Alam; Yao [AY19] suggest that methods should also be evaluated separately.

Regarding privacy in sentiment analysis on social media, existing work by Alatrista-Salas et al. [ACN19] proposes an approach utilizing bloom filters, which in contrast to DP do not impose noise on the data. Accordingly, there is only a minimal performance hit. Bloom filters alone however, are shown to be prone to targeted cryptanalysis attacks on word frequencies [Ch18]. In addition, bloom filters just provide ambiguity regarding direct data reconstruction attacks but do not result in a private prediction model, leaving room for e.g. membership inference attacks and other threats when publishing the model [BBL12; Sh17]. Recent approaches even propose a combination of bloom filters and DP [RVD22].

Our work contributes to the development of privacy-preserving sentiment analysis models in multiple ways. For our baseline we further study other preprocessing options for the provided twitter dataset and take into account separate evaluations for each. Regarding

[3] Go et al. [GBH09] call their LR model a Maximum Entropy model (MaxEnt) but both are exactly the same techniques under different names.

privacy, we take a different approach than bloom filters by instead applying DP on multiple privacy levels and assess the resulting performance loss compared to non-private training.

4 Experimental Setup

In the following, we present our general methodology and provide details on the conducted experiments along with their implementation. Reference code is available from our GitHub repository: `https://github.com/felix2246/dp-sent-analysis-twitter`.

4.1 Methodology

In a first step we implement for a non-private baseline model for reference. To achieve that, we evaluate the sentiment analysis model results regarding different preprocessing methods and provide several candidates of machine learning methods for our task, namely Multinomial Naive Bayes (MNB), Support Vector Machine (SVM) and Logistic Regression (LR). We evaluate the performance of each model using the accuracy metric also seen in related work presented in Sect. 3. Through this process we find the best combination of preprocessing and machine learning approach.

For our DP experiments, we take our fine-tuned LR baseline model setting and create a private model by employing the DP-SGD algorithm during the training process. We choose the LR algorithm for our private approach because it supports SGD which can be extended to instead use DP-SGD. We then assess the impact of training for different privacy levels of $\varepsilon = \{0.1, 1, 10\}$ on model results. The additional level of $\varepsilon = \infty$ corresponds to the non-private baseline, where no privatization is guaranteed using DP. An $\varepsilon \leq 1$ is commonly seen as a strong privacy guarantee [Ca19; Na21]. By evaluating DP levels a magnitude higher and lower than $\varepsilon = 1$, we want to gain better insights into the relation between performance and privacy for this task [La23].

4.2 Environment

The primary hardware used in this study is equipped with 16 GB RAM and an NVIDIA GeForce RTX 2070 GPU. In order to perform more complex calculations, we additionally use server-side GPU resources in the form of a NVIDIA GeForce GTX 2080Ti. The experiments are run in Jupyter notebooks. We use popular machine learning libraries for Python, namely TensorFlow[4], Keras[5] and scikit-learn[6]. To implement DP, we resort

[4] Available at `https://github.com/tensorflow/tensorflow`
[5] Available at `https://github.com/keras-team/keras`
[6] Available at `https://github.com/scikit-learn/scikit-learn`

to TensorFlow Privacy[7] which expands the core library with DP mechanisms and the functionalities to track privacy budgets. To ensure reproducible results during preprocessing and training process, we define a fixed random seed with a set value of 42. Consequently, the described environment and the details in the following sections can be used to recreate our results.

4.3 Data and Preprocessing

We utilize the automatically created sentiment140 dataset introduced by Go et al. [GBH09], which consists of 1.6 million tweets gathered from the Twitter API with only minimal data preparation. The tweets provide an even split into the positive and negative sentiment classes. The dataset also provides separate data in a predefined test set. We additionally create a validation set for our model training that consists of 10% of the original training data. After removing retweets and duplicate tweets from the training data, we preprocess the data using different combinations of approaches and later compare their influence on model performance. Following the idea of Alam; Yao [AY19], we also evaluate our preprocessing methods separately. Therefore, we define preprocessing rows containing multiple techniques, that build upon each other. We include yet untested combinations and in total propose the following rows:

- Row 1: lowercase + remove punctuation + remove words with less than two letters.
- Row 2: row 1 + replace usernames and URLs with keywords + transform repeated letters. This row is equal to the preprocessing of Go et al. [GBH09].
- Row 3: row 2 + remove stop words.
- Row 4: row 2 + transform word contractions (don't, can't, etc.).
- Row 5: row 3 + row 4.

Row 3 and row 4 are an exception in the sense, that they are not building on each other, which is done to evaluate stop word removal and word contractions separately. This separation is mainly needed due to the stop word list already containing some resolved word contractions, which might mask the actual influence of such transformations. In Sect. 5.1, we test each single row in experiments on the dataset.

4.4 Vectorization

Since machine learning algorithms cannot use raw text as input, it is transformed into vectors. Therefore, we convert the given text to a bag-of-words-model (BOW). There are

[7] Available at `https://github.com/tensorflow/privacy`

different possibilities for defining the value representation inside the word vectors which are given as metrics. These include the binary metric regarding a word's existence in the text, where for each word the binary value 0 (does not exist) or 1 (exists) is set. Furthermore, the count metric is given where the occurrence of the word is quantified. Within the BOW it can happen, that words, although they occur frequently, do not have a higher information content compared to rarer words. Therefore we use the tf-idf value, a measure of word appearance with document-based offset, as a third metric to normalize differing word frequencies.

For the conversion of the available corpora into a vector model, we use the TfidfVectorizer and CountVectorizer from the scikit-learn library. The set of features was limited to 5,000 and an unigram model was applied, so that each word occurring in the corpus is treated as a single feature. The associated classes for our training data points are stored as a one-dimensional vector where the index of the respective class corresponds to the index (line) of the associated document in the BOW.

4.5 None-Private Baseline

Our preprocessing results in five vectorized datasets (one for each row) and each one is accompanied by the three vectorization metrics of existence, count, and tf-idf value. For our non/private baseline approach, we test the three different machine learning algorithms MNB, SVM, and LR. This leads to a total of 45 possible combinations and therefore 45 baseline experiments.

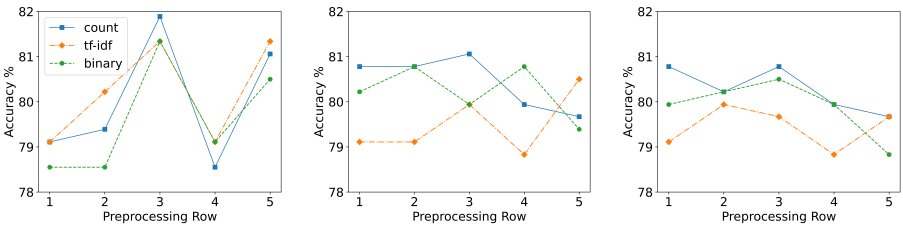
SVM and MNB are implemented in scikit-learn. LR is realized using a shallow sigmoid neural network model according to Liquet et al. [LMN22], utilizing Keras and TensorFlow. For LR we further use the SGD optimizer. We determine the most suitable learning rate of 1e-3 by testing different starting values in a cyclical approach [Sm17] and besides decay the learning rate on plateaus [Yo19]. At the same time, an epoch count of 58 and batch size of 16 were best in our experiments.

4.6 Privacy-Preserving Approach

To create a privacy-preserving model and to estimate the impact on performance, the baseline LR is extended to use DP-SGD training. Since LR with the preprocessing row 3 and the count metric shows the highest accuracy as presented in Sect. 5.1, the DP model is implemented building on this foundation. We again train for 58 epochs, using a batch size of 16, and with the same learning rate of 1e-3. With our training dataset containing $N = 1,421,664$ entries, we set $\delta = 1e-7$ according to $\delta \ll \frac{1}{N}$ [Dw06]. To obtain different ε values in experiments, the noise level is modified according to these dataset statistics.

5 Results

5.1 None-Private Baseline



(a) MNB (b) SVM (c) LR

Fig. 1: Change in accuracy (%) for each algorithm in relation to used preprocessing rows and metrics.

Tab. 1: Test accuracy (%) of the baseline models regarding the different available vectorization metrics, preprocessing rows, and machine learning algorithms. Row 2 conforms to the preprocessing by Go et al. [GBH09]. The algorithms are Multinomial Naive Bayes (MNB), Support Vector Machine (SVM) and Logistic Regression (LR).

vectorization metric	preprocessing	algorithm		
		MNB	SVM	LR
binary	row 1	78.55	80.22	79.94
	row 2	78.55	80.78	80.22
	row 3	81.34	79.94	80.50
	row 4	79.11	80.78	79.94
	row 5	80.50	79.39	78.83
count	row 1	79.11	80.78	**80.78**
	row 2	79.39	80.78	80.22
	row 3	**81.89**	**81.06**	**80.78**
	row 4	78.55	79.94	79.94
	row 5	81.06	79.67	79.67
tf-idf	row 1	79.11	79.11	79.11
	row 2	80.22	79.11	79.94
	row 3	81.34	79.94	79.67
	row 4	79.11	78.83	78.83
	row 5	81.34	80.50	79.67

This section provides an evaluation of our non-private models with respect to the preprocessing and vectorization metrics. In terms of learning algorithms, we further evaluate MNB, SVM, and LR. The used preprocessing rows and vectorization metrics have been defined in Sect. 4.3. While Tab. 1 provides a tabular overview of the resulting accuracies, we also provide graphical support in Fig. 1.

Regarding vectorization metrics, tf-idf produced the overall worst performance, while the count metric showed the best results for all tested algorithms. We also find a winner in terms of preprocessing rows, where row 3 takes the lead across all tested models. Row 3 mainly introduced the removal of stop words and some word contractions. The negative influence of row 4 (and row 5) shows that the transformation of word contractions can not be seen as favorable and we therefore conclude stop words to be the primary factor for row 3. Our assumption is that the transformation in a unigram model leads to the loss of context of individual words, which is represented in the word contractions by splitting into multiple features. For example, the model normally treats *don't* as a single feature, but after the conversion *do* and *not* are treated as separate features. From that, the direct relationship between these two words is no longer apparent to the model. The ubiquitous performance differences between the rows mark the importance of choosing the correct preprocessing and vectorization methods.

Talking about algorithms, the MNB takes the crown in our baseline evaluation reaching 81.89% accuracy. Comparing our results to the original work by Go et al. [GBH09], our MNB and LR models (81.9%, 80.8%) slightly outperform their counterparts in the unigram setting (81.3%, 80.5%). Our SVM (81.1%) on the other hand, is not able to reach the same level (82.2%). The difference in SVM performance could be explained by the fact that Go et al. [GBH09] used a different implementation, namely "SVM$^{\text{light}}$". Our addition of stop word removal to the existing methods from related work in Go et al. [GBH09] showed to be a general improvement.

5.2 Privacy-Preserving Approach

Tab. 2: Baseline and private models compared on test accuracy (%), while using preprocessing row 3 and count metric.

ε	**accuracy**
∞	80.78
10	78.27
1	77.99
0.1	76.04

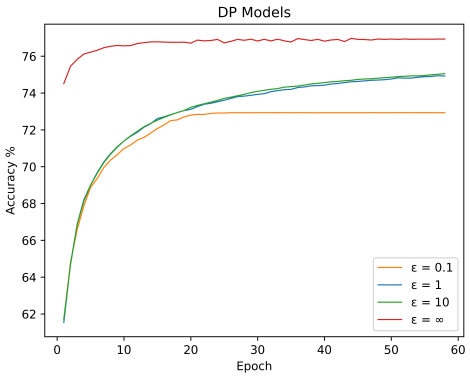


Fig. 2: Learning curves regarding accuracy (%) on the validation data over the epochs for the models with different ε privacy levels. We use preprocessing row 3 and count metric.

Our results presented in Tab. 2 confirm the expected utility-privacy trade-off between strong ε-guarantees and accuracy mentioned in Sect. 2.3. This reduction in performance is due to the added noise on the gradients during model training, which makes it harder to find the minimum of the loss function and leads to a sub-optimal adjustment of model weights.

Fig. 2 showcases the training process of our compared models by plotting their learning curves. We first note the similarity in accuracy for $\varepsilon = 10$ and $\varepsilon = 1$, with the strong level of $\varepsilon = 1$ giving almost the same results as the substantial weaker guarantee. We also notice, that the diminishment of the accuracy curves between non-private model and the models for $\varepsilon = \{10, 1\}$ is roughly the same as the the one between $\varepsilon = \{10, 1\}$ and $\varepsilon = 0.1$. Thus, the negative effects on performance are more significant when first introducing DP into the training process and again when going to very strict privacy guarantees of $\varepsilon = 0.1$.

Taking a closer look at the training curves of the private models in Fig. 2, we note the similarity in training processes for $\varepsilon = 10$ and $\varepsilon = 1$. While both curves do not seem to reach a plateau after our set 58 epochs, extending the epoch count would also increase ε and therefore the necessary noise. Increasing the performance of a DP model by extending the training duration is therefore accompanied by a negative trade-off that needs additional consideration. At $\varepsilon = 0.1$, the model reaches an early lower plateau, highlighting how the higher noise (negatively) influences the training process.

Focusing again on Tab. 2, the resulting accuracy loss of our private models compared to the non-private baseline at $\varepsilon = \infty$ lies in the range of 2.5–4.7%. This shows that our private results are staying close to the non-private baseline, even when training for strong privacy of $\varepsilon \leq 1$. Taking into account the provided DP guarantees, our approach can offer viable classifications even under strict user privacy considerations. The related privacy-preserving work by Alatrista-Salas et al. [ACN19] using bloom filters shows an even smaller performance loss of 1–3%. However, as layed out in Sect. 3, DP is the superior method when seeking a secure and reliable privatization. Our results show that DP models can maintain high accuracy while preserving privacy in sentiment analysis on Twitter data.

6 Conclusion

In this work, we propose a front-to-back approach for preserving privacy in sentiment analysis on Twitter using DP. We first perform an extensive evaluation of preprocessing methods and test different learning algorithms to find strong performing non-private baseline models. Our results when including different DP levels then shows that we can maintain high accuracy in our task while limiting threats for individual users' data. Our solution can thus give provable privacy protection for sensitive information, which could be a deciding factor for data sharing by users. At the same time, our private models still enable a reliable analysis of the given data. We believe that our work has meaningful implications for researchers and practitioners working in the field of sentiment analysis searching for a usable balance in the trade-off between privacy and utility, especially for social media data.

Future work should explore the real world impact of DP against attacks in practical scenarios to confirm its effectiveness. Further research might also investigate the impact of different DP parameters on other machine learning models for sentiment analysis like recent transformer-based architectures [Na20].

Acknowledgments. We thank the reviewers and organizers for their helpful feedback. We also thank our colleague Victor Christen for his provided insights into bloom filter privacy. The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany and by the Sächsische Staatsministerium für Wissenschaft Kultur und Tourismus in the program Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project identification: ScaDS.AI. Computations for this work were done (in part) using resources of the Leipzig University Computing Centre.

References

[Ab16] Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H. B.; Mironov, I.; Talwar, K.; Zhang, L.: Deep learning with differential privacy. In: ACM SIGSAC. 2016.

[ACN19] Alatrista-Salas, H.; Cordero, H.; Nunez-del-Prado, M.: PS I Love You: Privacy Aware Sentiment Classification. Computación y Sistemas 23/4, 2019.

[AY19] Alam, S.; Yao, N.: The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis. Comput Math Organ Theory/, 2019.

[BBL12] Bianchi, G.; Bracciale, L.; Loreti, P.: "Better Than Nothing" Privacy with Bloom Filters: To What Extent? In: Privacy in Statistical Databases. Springer Berlin Heidelberg, 2012, ISBN: 978-3-642-33627-0.

[Ca17] Cambria, E.; Das, D.; Bandyopadhyay, S.; Feraco, A.: A Practical Guide to Sentiment Analysis. Springer Publishing Company, Inc., 2017, ISBN: 3319553925.

[Ca19] Carlini, N.; Liu, C.; Erlingsson, Ú.; Kos, J.; Song, D.: The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. In: USENIX Security Symposium. Vol. 267, 2019.

[Ch18] Christen, P.; Vidanage, A.; Ranbaduge, T.; Schnell, R.: Pattern-mining based cryptanalysis of Bloom filters for privacy-preserving record linkage. In: Advances in Knowledge Discovery and Data Mining: PAKDD. Springer, 2018.

[Dw06] Dwork, C.: Differential Privacy. In: Automata, Languages and Programming. Springer Berlin Heidelberg, 2006.

[EC16] European Parliament; Council of the European Union: Regulation (EU) 2016/679 of the European Parliament and of the Council, 2016.

[GBH09] Go, A.; Bhayani, R.; Huang, L.: Twitter sentiment classification using distant supervision. CS224N project report, Stanford 1/12, p. 2009, 2009.

[La23] Lange, L.; Schneider, M.; Christen, P.; Rahm, E.: Privacy in Practice: Private COVID-19 Detection in X-Ray Images. In: 20th International Conference on Security and Cryptography (SECRYPT 2023). SciTePress, 2023.

[LMN22] Liquet, B.; Moka, S.; Nazarathy, Y.: The Mathematical Engineering of Deep Learning, 2022, URL: https://deeplearningmath.org.

[MSK11] Mao, H.; Shuai, X.; Kapadia, A.: Loose Tweets: An Analysis of Privacy Leaks on Twitter. In: Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society. WPES '11, 2011.

[Na20] Naseem, U.; Razzak, I.; Musial, K.; Imran, M.: Transformer based deep intelligent contextual embedding for twitter sentiment analysis. Future Generation Computer Systems 113/, pp. 58–69, 2020.

[Na21] Nasr, M.; Songi, S.; Thakurta, A.; Papernot, N.; Carlin, N.: Adversary instantiation: Lower bounds for differentially private machine learning. In: 2021 IEEE Symposium on security and privacy (SP). IEEE, pp. 866–882, 2021.

[Ne12] Neri, F.; Aliprandi, C.; Capeci, F.; Cuadros, M.; By, T.: Sentiment Analysis on Social Media. In. ASONAM '12, IEEE Computer Society, 2012.

[RVD22] Ranbaduge, T.; Vatsalan, D.; Ding, M.: Privacy-preserving Deep Learning based Record Linkage, 2022, arXiv: 2211.02161 [cs.CR].

[Sh17] Shokri, R.; Stronati, M.; Song, C.; Shmatikov, V.: Membership Inference Attacks against Machine Learning Models, 2017, arXiv: 1610.05820 [cs.CR].

[Sm17] Smith, L. N.: Cyclical Learning Rates for Training Neural Networks, 2017, arXiv: 1506.01186 [cs.CV].

[Yo19] You, K.; Long, M.; Wang, J.; Jordan, M. I.: How Does Learning Rate Decay Help Modern Neural Networks?, 2019, arXiv: 1908.01878 [cs.LG].