Image Watermarking for Machine Learning Datasets

Palle Maesen
Cyber Security Group, Delft University of Technology
The Netherlands
P.Maesen@student.tudelft.nl

Devriş İşler
IMDEA Networks Institute & UC3M
Spain
devris.isler@imdea.org

Nikolaos Laoutaris
IMDEA Networks Institute
Spain
nikolaos.laoutaris@imdea.org

Zekeriya Erkin
Cyber Security Group, Delft University of Technology
The Netherlands
Z.Erkin@tudelft.nl

ABSTRACT

Machine learning has received increasing attention for the last decade due to its significant success in classification problems in almost every application domain. For its success, the amount of available data for training plays a crucial role in the creation of a machine-learning model. However, the data-gathering process for machine learning algorithms is a tedious and time-consuming task. In many cases, the developers rely on publicly available datasets, which are not always of high quality. Recently, we are witnessing a data market paradigm where valuable datasets are sold. Thus, once the dataset is created or bought, protecting the dataset against illegal use or (re)sale and establishing intellectual property rights is necessary. In this paper, we investigate the question of deploying well-studied image watermarking techniques to be applied to classification algorithm datasets, without degrading the quality of the dataset. We investigate whether Singular Value Decomposition (SVD)-based techniques from image watermarking could be deployed on machine learning datasets or not. To this end, we chose the watermarking technique described in [8] and applied it to a machine-learning dataset. We provide experimental results on the robustness of the scheme. Our results show that the watermark embedding scheme provides decent imperceptibility and robustness against update, zero-out, and insertion attacks but, it is not successful against deletion attacks. We believe our work can inspire researchers who might want to consider applying well-studied image watermarking techniques to machine learning datasets.

CCS CONCEPTS

• **Social and professional topics** → **Computing / technology policy**; • **Intellectual property** → Digital rights management .

KEYWORDS

Watermarking, machine learning, data ownership, data economy

Data Economy Workshop (DE '23), June 18, 2023, Seattle, USA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Palle Maesen, Devriş İşler, Nikolaos Laoutaris, and Zekeriya Erkin. 2023. Image Watermarking for Machine Learning Datasets. In *Proceedings of Data Economy Workshop (Data Economy Workshop (DE '23)).* ACM, Seattle, WA, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The amount of data needed to create state-of-the-art machine learning (ML) models is continuously increasing as the complexity and volumes of these models grow [12]. However, public datasets available for training are limited and sometimes not of high quality. Furthermore, application-specific datasets (e.g., health data) are very valuable. Imagine a company that invested an incredible amount of time and resources into the data-collection/generation. Later, the company sells/shares the dataset [4] in order to have a financial gain (or for any other non-economic incentives such as a public dataset).[1] Therefore, it is essential for companies to protect their ownership of their datasets from infringement, piracy, and/or another entity re-selling their ML datasets without their authorization.

Watermarking is a well-known technique for protecting ownership rights in media such as image, video, and audio, database, and machine learning models. A watermarking scheme commonly consists of two algorithms: watermark embedding and extraction. In the watermark embedding, a data stream (and/or a high entropy value) as a secret is embedded into the host data [11] which distorts the data without degrading the utility. The watermark extraction using the secret (e.g., the data stream, the high entropy secret) is computed to prove ownership [16], to check whether the data has been tampered with [10], or to manage copy control [7]. However, compared to media watermarking, non-media watermarking techniques are a newer sub-domain [15]. Existing dataset watermarking techniques evaluate their utility and effect of watermarking on datasets in terms of various metrics such as similarity [26], changes in the mean, and standard deviation [3]. However, watermarking a machine learning (ML) dataset can be complicated since not only the effect of watermarking on the dataset but also the effect of watermarking on the algorithm trained by the dataset should be carefully analyzed.

ML datasets and images are both N-dimensional matrices containing (often numerical) data. Therefore, in this work, we investigate whether well-studied image watermarking techniques can be applied to machine learning datasets for proof of ownership since media

[1] In this work, we do not discuss issues related to privacy since it is out-of-scope of our work. However, companies should protect the privacy of the entities they collected the dataset from according to regulations (e.g., GDPR, CCPA) they might be subjected to.

watermarking is more advanced compared to dataset watermarking. For this purpose, we chose an SVD-based watermarking scheme [8] due to its proven robustness with media related use-cases [6].

Our approach. To be able to watermark a dataset using SVD-based watermarking, we first normalize the dataset to the range defined by the image watermarking (i.e., $(0, 255)$). Then, the normalized dataset is watermarked by the image watermarking scheme as described by [8]. Finally, after watermarking, we train various machine learning models on the watermarked dataset and calculate the accuracy. We also consider two main metrics, *imperceptibility* and *robustness*, to measure the effects of image watermarking on the actual dataset. To the best of our knowledge, our work is the first work applying image watermarking to ML datasets and measuring the effects of watermarking on ML models as well as on ML datasets. Thus, our work may provide a valuable connection between a mature and an emerging area, and inspire researchers to consider applying and improving well-studied image watermarking techniques to machine learning and understand the effects of porting them to a new domain.

Our contributions. Our first and foremost contribution is to apply image watermarking on machine learning datasets and train models on such watermarked datasets using two real-world datasets. We later compare these models trained with the watermarked datasets with the models trained on the host/original datasets in terms of accuracy for *imperceptibility*. Our evaluations show that watermarking introduces at most a 0.1% accuracy loss. We also measure the distortion introduced by the watermarking to the dataset. We evaluate robustness against *deletion*, *update*, *insertion*, *zero-out*, and *multi-faceted* attacks. This evaluation shows that the technique is robust against *update*, *insertion*, and *zero-out* attacks but fails to keep the watermark intact against *deletion* and *multifaceted* attacks. These observations highlight important directions for further work in adapting existing techniques to the new domain of ML watermarking.

2 BACKGROUND

In this section, we briefly present the building blocks used throughout the paper. In Table 1, we provide the notations used.

Table 1: Notations used throughout the paper.

Symbol	Description
U	The U component of the SVD
Σ	The Σ component of the SVD
V	The V component of the SVD
D	The host data
D_n	The normalized host data
W	The watermark
D_w	The watermarked host data
W'	The extracted watermark
a	The index of the attribute that was watermarked
m	The maximum value of an attribute

2.1 Singular Value Decomposition (SVD)

Singular Value Decomposition [18] is an algebraic tool to decompose an original matrix. Let A be the original $m \times n$ matrix, its SVD is described as $A = U\Sigma V^T$. Let A_n be the order of the matrix A, Σ is equal to $diagonal(\sigma_0, ..., \sigma_{A_n})$ where each σ represents a singular value of A. The matrices U, Σ, and V all have the same size as A.

2.2 Image Watermarking

The watermarking process contains two steps: the embedding process and the extraction process [14]. During the embedding process a datastream or watermark W is inserted into the host data D. The extraction process entails extracting the watermark from the (suspected) watermarked data.

There are two types of image watermarking techniques: 1) *Spatial Domain-based Techniques:* These techniques deploy insignificant changes in the spatial domain, which are invisible to human eye. Examples of these techniques involve the Least Significant Bit [5], Intermediate Significant Bit [24] and patchwork [23]. However, these techniques are also easy to attack by simple modification of the data. 2) *Transform Domain-based Techniques:* These techniques introduce changes in the frequency domain such that the changes cannot be seen with the naked human eye and are also robust against modifications such as compression, rotation and cuts. Examples of these techniques are based on the Discrete Cosine Transform [21], Discrete Wavelet Domain [2], Singular Value Decomposition [8], and QR Decomposition [20].

Image Watermarking via SVD. Let W be a binary image with $l \times m$ bits used as a watermark. Let D be a host image to watermark. Watermark embedding and extraction are as follows:

• **Embedding:** To watermark D with W, the SVD-based image watermarking [8] partitions a host image D (which is being watermarked) into $n \times n$ blocks. Each block is decomposed using SVD. The coefficients of the U component are used to determine relations (as positive or non-positive) to embed a watermark bit. Due to page limit, we do not elongate our discussion with regarding the relation calculation rather we refer readers to [8] for more details. The exact method to embed the watermark bit can be found in [8]. Finally, the watermarked block is constructed by multiplying the watermarked U-component, the Σ-component, and the transpose of the V-component. The watermarked image D_w is constructed by taking all the watermarked blocks.

• **Extraction:** Let D_w be a watermarked image. Similar to embedding, D_w is divided into $n \times n$ blocks, and each block is transformed by SVD. After SVD generation, the U-component is analyzed for each block to extract the watermark bit. These bits form the extracted watermark W'. Later, W' is compared to W to determine if D_w was watermarked by W. If W and W' are similar to each other according to a similarity analysis, then D_w is said to be watermarked by W; however, it depends on the owner and/or users' applications.

3 WATERMARKING AN ML DATASET

Watermarking techniques that transform the domain are more robust than spatial domain techniques as discussed by Begum et al. [6]. The techniques used in this paper cannot take advantage of frequency properties. Therefore, an SVD-based image watermarking is deployed to watermark ML datasets, and the watermarking scheme by Chang et al. [8] is chosen as the SVD-based image watermarking.

Watermarking Embedding. Figure 1 illustrates the overview of the watermarking embedding for a (host) machine learning dataset

D with a watermark W using SVD image watermarking. The SVD-

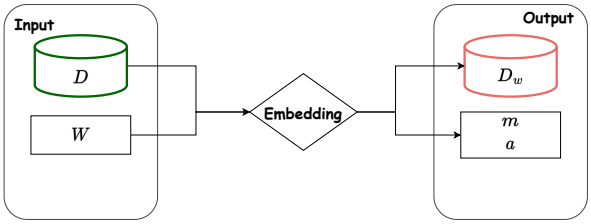


Figure 1: Overview of embedding.

based image watermarking embedding for a machine learning dataset is as follows:

1) **Data pre-processing**: The SVD-based image watermarking [8] requires a colored image in which each data point is an integer between 0 and 255. However, an ML dataset D does not have the same property; thus, we have to normalize D by dividing each data point by the maximum value m where D_n represents the normalized D.

2) **Choice of attribute**: Colored image watermarking techniques mostly focus on the blue channel. This is because the human eye is least perceptible to blue color [19]. The similar logic can be applied to machine learning datasets where the least important attribute is selected since changing their values will not affect the dataset dramatically. Our analysis showed that the least important attribute is the attribute with the lowest variance for the datasets used. Thus, the attribute a of D with the smallest variance is chosen to be watermarked for our analysis. Other approaches of course can be considered, e.g., using semantic knowledge about the data and the intended use-case.

3) **Data partition**: After the normalization and choice of the attribute, D_n is partitioned into $n \times n$ blocks as in SVD-based image watermarking.

4) **Embedding of the watermark**: After data partitioning, each block is transformed by SVD. Then the elements in the first column, second and third rows are taken from the U matrix in the SVD. The relation between these elements is used to embed W. If the watermark bit is one, then the relation between the second and third elements is made positive. Otherwise, the relation is made negative. To define how positive or negative this relation may be, a threshold value is determined. For instance, consider Figure 2 where we illustrate the procedure for a block of D_n for the block size is 4×4, the watermark bit is 1, and the threshold is 0.03. First, the normalized host data D_n is decomposed into its U, Σ, and V matrices (U component is shown in the figure). It checks the second and third elements of the first column which are $U(2,1) = 0.44$ and $U(3,1) = 0.45$. The relation between the two elements is negative since $0.44 - 0.45 < 0$. Since the watermark bit is 1 and it does not match the relation, the relation needs to be modified. To modify, it uses a pre-determined threshold value (which is 0.03 for this example) and sets the second element as $U(2,1) = ||0.44| + (0.03 + |0.44 + 0.45|)/2| = 0.46$ and the third element as $U(3,1) = ||0.45| - (0.03 + |0.44 + 0.45|)/2| = 0.43$. Note that if the watermark bit was 0, it would mean that the relation matches with the watermark bit, and the difference needs to be bigger or equal to the threshold. Thus, it would have set the elements as $U(2,1) = ||0.44| - (0.03 - |0.44 - 0.45|)/2| = 0.43$ and $U(3,1) = ||0.45| + (0.03 - |0.44 - 0.45|)/2| = 0.46$.

5) **Reconstruction**: After each block is watermarked as discussed in the previous item (4), a watermarked version of D_n (denoted by D'_n) is reconstructed by multiplying the decomposition of each block and multiplying again with the original m.

6) **Output**: It returns the watermarked data D_w constructed by the combination of the watermarked blocks, the attribute a, and the maximum value m used for normalization to be used for extraction.

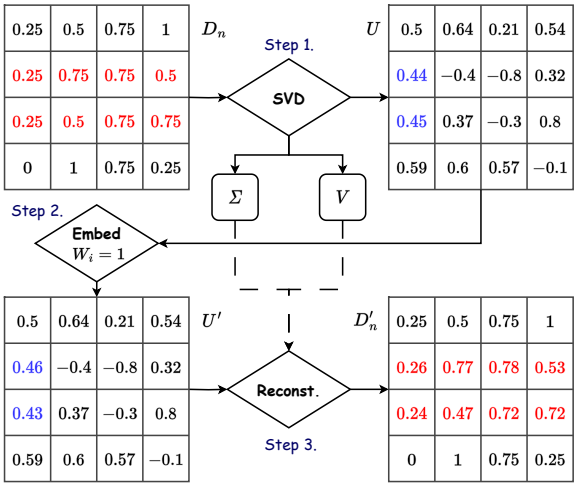


Figure 2: Illustration of embedding a watermark bit $W_i = 1$ in a single normalized block D_n.

Watermark Extraction. Figure 3 illustrates the watermark extraction procedure for a given watermarked dataset D_w, the watermark W, and the values m and a. The extraction is as follows:

1) **Data pre-processing**: Pre-processing is the same as the pre-processing procedure in the embedding phase except for calculations of m and a since they are given.

2) **Extraction:** Each block is transformed by SVD as in the embedding. For each block, the U component is analyzed and the watermark bit is extracted. Whether the bit is 0 or 1 is determined by comparing the values of the second and third rows in the first column applying the same relation idea as in embedding.

3) **Verification**: A watermark W' is constructed using the watermark bits extracted. The watermark is verified if $W \approx W'$ and returns the boolean result as an output.

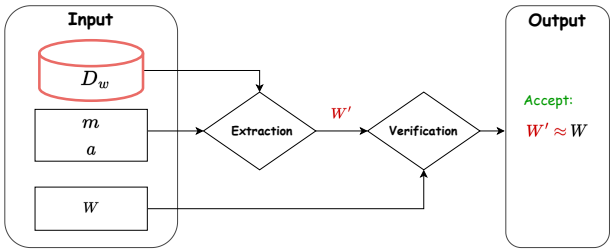


Figure 3: Overview of extraction.

4 EXPERIMENTAL SETUP AND RESULTS

4.1 Setup

The experiments are produced on a standard laptop with an Intel Core $i5-2557M$ CPU 2.7GHz and 4GB RAM, running a 64-bit Linux OS. The watermarking scheme is implemented in C++. The tests are conducted in `Python`, and `scikit-learn` library is used for the machine learning algorithms. We use two datasets from the UCI archive [9] for watermarking and training: 1) iris-dataset; 2) dry-bean dataset [13]. Despite the fact that the datasets are related to physical objects, both of them consist of numerical values regardless of their names.

The **iris dataset** consists of 150 records with 5 attributes: 1) sepal length; 2) sepal width; 3) petal length; 4) petal width; and 5) species. The **dry-bean dataset** is a dataset containing around 13000 entries of 7 types of dry beans. Each record has 16 attributes where 12 of these are dimensions, and the other 4 are shape forms.

4.2 Results

The watermarked dataset is tested on the imperceptibility and robustness of the watermark. We evaluate the imperceptibility of watermarking considering two metrics: 1) change on a host dataset after watermarking, and 2) change in accuracy of the machine learning model trained on the watermarked dataset. To evaluate how much distortion our approach adds to an ML dataset, *mean squared error* is used. For accuracy evaluation of ML models trained by the watermarked datasets, we use two algorithms [17]: *KMeans algorithm*; and *Bayes classification algorithm*. Moreover, the *Decision tree algorithm* is used for similarity analysis.

The iris-data is watermarked with a random 1×9 binary image, and the dry-bean-data is watermarked with the binary image shown by Figure 4. For our evaluations, we first run the machine learning algorithms for each host dataset to calculate the accuracy. After that, we train the same algorithms with the watermarked versions of each dataset to measure the effect of watermarking on the ML model.

Figure 4: The original watermark.

4.2.1 Effect of Watermarking on the Datasets. Figure 5a shows the iris-data before (left) and after (right) the watermark embedding where 0.02 is chosen as a threshold. On the y-axis, the value of the attribute can be seen, and the x-axis represents the index of the tuple: n. The *sepal width* attribute valued between 2 cm and 4.4 cm is selected to watermark the dataset. As shown in the figure, the overall form of the dataset is maintained for each species (setosa, versicolor, and virginica), but it has been leveled out. Figure 5b shows the dry-bean data before and after embedding for different bean types. The y-axis represents the value of the nth attribute, and n is shown on the x-axis. The *ShapeFactor4* attribute is selected to watermark the dataset. A threshold of 0.002 is used. The ShapeFactor attribute has a minimum value of 0.94 and a maximum value of 1.0. As it is evident by Figure 5b, there are no major visible changes after watermarking. The thresholds were chosen as suggested by [8] and our testing while we plan to optimize the selection of such parameters in the future.

Mean Squared Error. Figure 6 and Figure 7 show the mean squared errors between the host datasets and the watermarked datasets for the iris-data and dry-bean-data, respectively. Different thresholds are used to show how mean squared errors are changing when the threshold value increases. For both datasets, the error increases when the threshold increases.

4.2.2 Effect of Watermarking on ML Models. Next, we discuss the effects of watermarking on machine learning models trained by watermarked datasets using previously determined algorithms.

Decision Tree Classifier. For the decision tree classifier algorithm, we split the watermarked dataset into training and testing datasets. To determine the training and testing datasets, we randomly selected 30% of the watermarked dataset as the testing dataset while the rest (70%) of the dataset is assigned as a training dataset. Figure 8 shows the accuracy of the decision tree of the (watermarked) Iris-data. The accuracy of the model without watermarking was 96%. As shown in the figure, many threshold values result in high accuracy (95.54%) causing only 0.06% accuracy loss. Figure 9 shows the accuracy of the decision tree of the dry-bean dataset. The original accuracy was 91%. In contrast to the iris-data ML model, the accuracy of the model decreases when the threshold for watermarking increases. However, the accuracy tends to be over 90.2% causing only 0.8% accuracy loss which is tolerable. Overall, our evaluations show that watermarking does not cause dramatic loss in the accuracy.

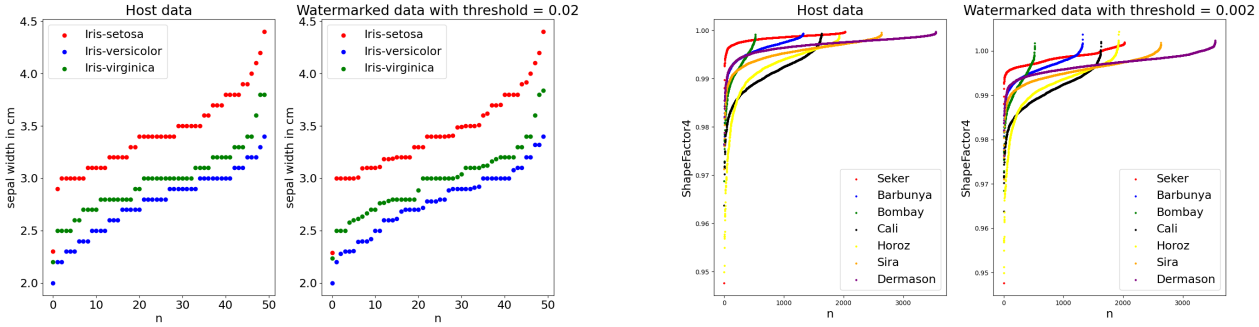
Bayes Classification Error. Figure 10 shows the accuracy of the BayesClassifier on the Iris dataset. The original accuracy of the classifier on the Iris host dataset was 93.3%. The accuracy changes depending on the threshold; however, it is over 93.1% for many of the threshold values causing only 0.02% loss while for other thresholds it is over 91%. Considering the accuracy of the Bayes Classifier for the dry-bean dataset, watermarking did not affect the accuracy at all (which was 76.2%) for any of the thresholds.

KMeans Centroids Error. Figure 11 and 12 show the euclidean distance of the centroids of the KMeans algorithm after watermarking for the Iris-data and the dry-bean data, respectively. For both datasets, the errors increased linearly as the threshold increased.

5 ROBUSTNESS ANALYSIS

Next, we discuss the robustness of our approach against *deletion*, *update*, *insertion*, *zero-out*, and *multifaceted* attacks. For our analysis, we used the watermarked dry-bean dataset, with a threshold of 0.05, as it has a wider attack surface. The extracted watermarks are shown for increasing severities of the attacks. Whether the watermark is accepted depends on the user's application.

5.0.1 Update Attack. In the update attack, an attacker tries to remove the watermark by adding random noise to the watermarked dataset. Figure 13 shows the extracted watermark after adding random noise (10%) to the dataset where in the figure, the ratio of the affected data is given above the images. The watermark is still recoverable which shows that our approach is resistant to the attack.



(a) Iris-Data **(b) Dry-Bean-Data**

Figure 5: Data before and after the watermarking where n represents the index of the tuple.

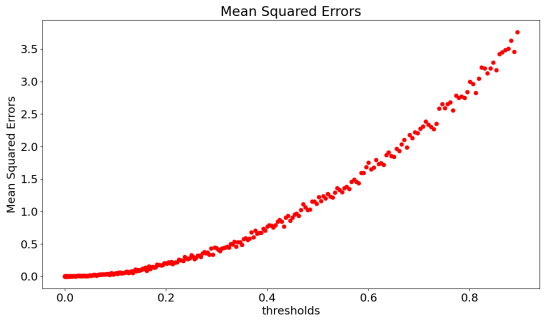


Figure 6: Mean Squared Errors results of Iris-Data.

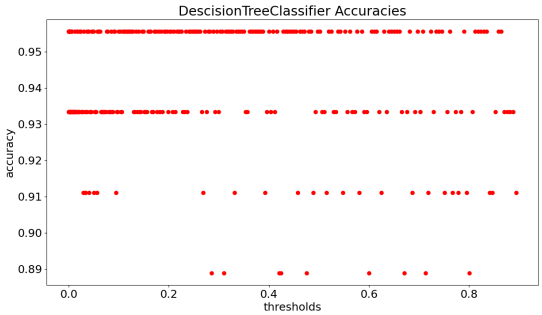


Figure 8: Decision Tree Classifier results of Iris-Data.

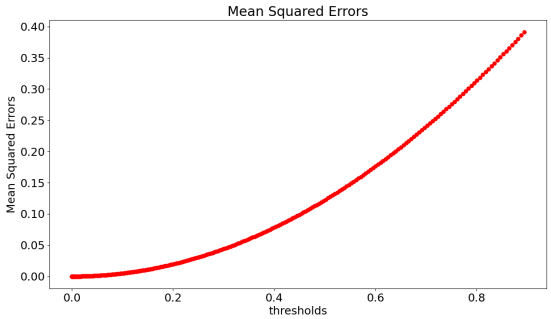


Figure 7: Mean Squared Errors results of Dry-Bean-Data.

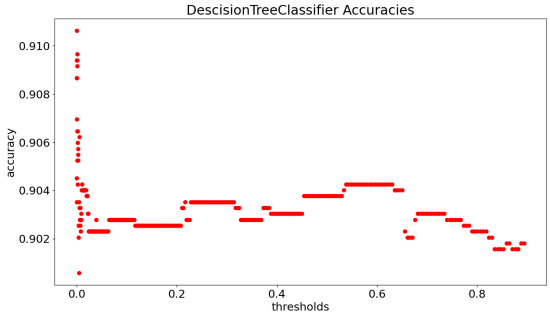


Figure 9: Decision Tree Classifier results of Dry-Bean-Data.

5.0.2 Deletion Attack. The deletion attack attempts to remove the watermark by deleting tuples from the dataset. Figure 14 shows the extracted watermarks after an amount of the data has been deleted. The watermark is not recognizable anymore after deleting 1% of the watermarked dataset.

5.0.3 Zero-out Attack. The zero-out attack updates some tuples in the dataset with zero. Figure 15 shows the extracted watermarks after the watermarked dataset is zeroed out with different percentages. After 10% of the dataset is zeroed out, the watermark is still recoverable although it is highly distorted.

5.0.4 Insertion Attack. In the insertion attack, an attacker tries to destroy the watermark by deleting tuples, and For each deleted tuple, it inserts its own. Our analysis (see Figure 16) shows that the watermark is highly destroyed after 10% of new data is inserted after deleting 10%.

5.0.5 Multifaceted Attack. The multifaceted attack is a more advanced attack strategy compared to others because it uses all the previous attacks to remove the watermark. After attacking 1% of the data, the watermark is highly distorted. At 3% is the watermark unrecognizable.

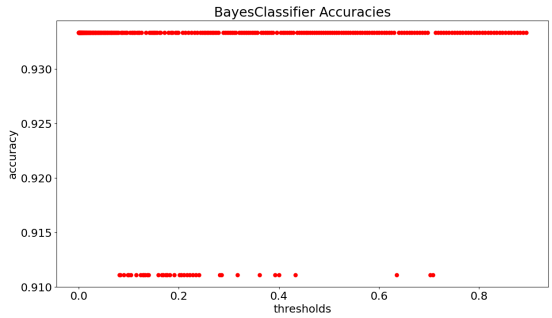


Figure 10: Bayes Classifier results of Iris-Data.

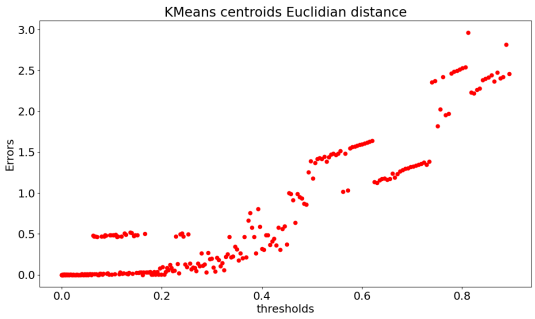


Figure 11: KMeans centroids errors of Iris-Data.

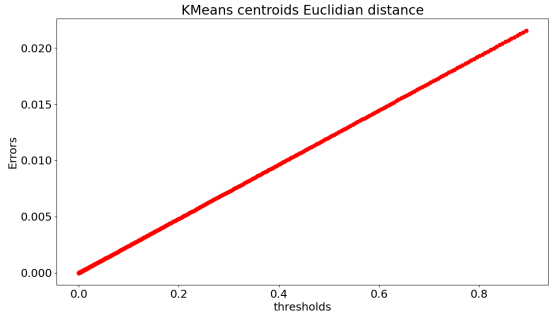


Figure 12: KMeans centroids errors of Dry-Bean-Data.

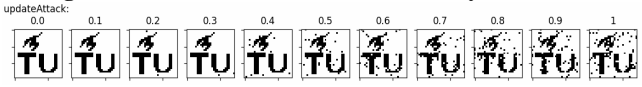


Figure 13: Update Attack.

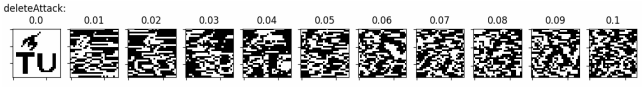


Figure 14: Deletion Attack.

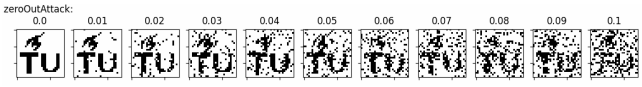


Figure 15: Zero-out Attack.

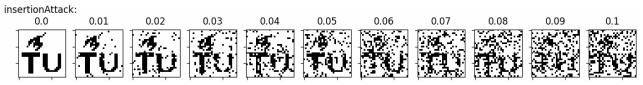


Figure 16: Insertion Attack.

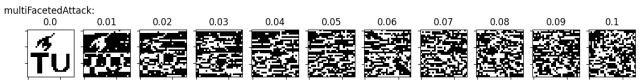


Figure 17: Multifaceted Attack.

6 RELATED WORK

Machine learning watermarking techniques are the closest prior works to our work. Uchida et al. [22] is the first known work that introduced the watermarking concept to deep neural networks in order to protect intellectual property (IP) and detect IP infringement of DNN models. A watermark is embedded into one of the convolutional layers in a host DNN. Zhang et al. [25] watermarks a DNN model with three different watermarking strategies using image datasets: 1) embedding meaningful content to the training dataset; 2) embedding unrelated images from other classes; 3) watermarking by adding noises to the dataset. They show that the functionality of the DNN model does not change significantly. Adi et al. [1] is inspired by backdooring where a set of images, called a trigger set, chosen by the owner are labeled wrong. The owner uses its trigger set where the model is triggered to return the label as assigned in the embedding phase as proof of ownership.

7 CONCLUSION AND FUTURE WORK

In this paper, we investigate how an SVD-based image watermarking scheme [8] can be used to watermark machine learning datasets. We watermarked two real-world ML datasets using SVD-image watermarking with a few slight modifications (e.g., normalization). Our analysis showed that watermarking did not introduce dramatic distortion in the original datasets. Consequently, the machine learning models trained on the watermarked datasets had high accuracy rates. We empirically analyzed the robustness of our watermarking approach against *deletion*, *update*, *insertion*, *zero-out*, and *multifaceted* attacks. Our preliminary results show that SVD-based image watermarking can be used for machine learning datasets while keeping the accuracy rate high. However, as we discussed, our approach requires further improvements in order to increase resilience against deletion and multifaceted attacks which we plan to improve in the future. Moreover, it will be interesting to investigate other image watermarking techniques and conduct a comparative analysis using various real-world datasets for various machine learning models.

ACKNOWLEDGMENTS

Our work was supported by the European Union's HORIZON project DataBri-X (101070069).

REFERENCES

[1] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *{USENIX} Security Symposium ({USENIX} Security 18)*. 1615–1631.

[2] Himanshu Agarwal, Balasubramanian Raman, and Ibrahim Venkat. 2015. Blind reliable invisible watermarking method in wavelet domain for face image watermark. *Multim. Tools Appl.* 74, 17 (2015), 6897–6935. https://doi.org/10.1007/s11042-014-1934-1

[3] Rakesh Agrawal and Jerry Kiernan. 2002. Watermarking Relational Databases. In *International Conference on Very Large Data Bases, VLDB*. 155–166. https://doi.org/10.1016/B978-155860869-6/50022-6

[4] Santiago Andrés Azcoitia and Nikolaos Laoutaris. 2022. A Survey of Data Marketplaces and Their Business Models. *SIGMOD Rec.* 51, 3 (2022), 18–29. https://doi.org/10.1145/3572751.3572755

[5] Abdullah Bamatraf, Rosziati Ibrahim, and Mohd. Najib Mohd. Salleh. 2011. A New Digital Watermarking Algorithm Using Combination of Least Significant Bit (LSB) and Inverse Bit. *CoRR* abs/1111.6727 (2011). arXiv:1111.6727 http://arxiv.org/abs/1111.6727

[6] Mahbuba Begum and Mohammad Shorif Uddin. 2020. Digital Image Watermarking Techniques: A Review. *Inf.* 11, 2 (2020), 110. https://doi.org/10.3390/info11020110

[7] Jefferey A. Bloom, Ingemar J. Cox, Ton Kalker, Jean-Paul M. G. Linnartz, Matthew L. Miller, and C. Brendan S. Traw. 1999. Copy protection for DVD video. *Proc. IEEE* 87, 7 (1999), 1267–1276. https://doi.org/10.1109/5.771077

[8] Chin-Chen Chang, Piyu Tsai, and Chia-Chen Lin. 2005. SVD-based digital image watermarking scheme. *Pattern Recognit. Lett.* 26, 10 (2005), 1577–1586. https://doi.org/10.1016/j.patrec.2005.01.004

[9] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[10] Jiri Fridrich. 1998. Image Watermarking for Tamper Detection. In *Proceedings of the 1998 IEEE International Conference on Image Processing, ICIP-98, Chicago, Illinois, USA, October 4-7, 1998*. IEEE Computer Society, 404–408. https://doi.org/10.1109/ICIP.1998.723401

[11] Xingliang Huang and Bo Zhang. 2005. Robust Detection of Transform Domain Additive Watermarks. In *Digital Watermarking, 4th International Workshop, IWDW 2005, Siena, Italy, September 15-17, 2005, Proceedings (Lecture Notes in Computer Science, Vol. 3710)*, Mauro Barni, Ingemar J. Cox, Ton Kalker, and Hyoung Joong Kim (Eds.). Springer, 124–138. https://doi.org/10.1007/11551492_10

[12] Wan Soo Kim and Kyogu Lee. 2020. Digital Watermarking For Protecting Audio Classification Datasets. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, 2842–2846. https://doi.org/10.1109/ICASSP40776.2020.9053869

[13] M. KOKLU and I.A. OZKAN. 2020. Multiclass Classification of Dry Beans Using Computer Vision and Machine Learning Techniques. https://doi.org/10.1016/j.compag.2020.105507

[14] Sanjay Kumar, Binod Kumar Singh, and Mohit Yadav. 2020. A Recent Survey on Multimedia and Database Watermarking. *Multim. Tools Appl.* 79, 27-28 (2020), 20149–20197. https://doi.org/10.1007/s11042-020-08881-y

[15] Arezou Soltani Panah, Ron G. van Schyndel, Timos K. Sellis, and Elisa Bertino. 2016. On the Properties of Non-Media Digital Watermarking: A Review of State of the Art Techniques. *IEEE Access* 4 (2016), 2670–2704. https://doi.org/10.1109/ACCESS.2016.2570812

[16] Arkadip Ray and Somaditya Roy. 2020. Recent trends in image watermarking techniques for copyright protection: a survey. *Int. J. Multim. Inf. Retr.* 9, 4 (2020), 249–270. https://doi.org/10.1007/s13735-020-00197-9

[17] Susmita Ray. 2019. A Quick Review of Machine Learning Algorithms. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. 35–39. https://doi.org/10.1109/COMITCon.2019.8862451

[18] G. W. Stewart. 1993. On the Early History of the Singular Value Decomposition. *SIAM Rev.* 35, 4 (1993), 551–566. https://doi.org/10.1137/1035134 arXiv:https://doi.org/10.1137/1035134

[19] Qingtang Su and Beijing Chen. 2018. Robust color image watermarking technique in the spatial domain. *Soft Comput.* 22, 1 (2018), 91–106. https://doi.org/10.1007/s00500-017-2489-7

[20] Qingtang Su, Yugang Niu, Gang Wang, Shaoli Jia, and Jun Yue. 2014. Color image blind watermarking scheme based on QR decomposition. *Signal Process.* 94 (2014), 219–235. https://doi.org/10.1016/j.sigpro.2013.06.025

[21] Qingtang Su, Gang Wang, Shaoli Jia, Xiaofeng Zhang, Qiming Liu, and Xianxi Liu. 2015. Embedding color image watermark in color image based on two-level DCT. *Signal Image Video Process.* 9, 5 (2015), 991–1007. https://doi.org/10.1007/s11760-013-0534-2

[22] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. 2017. Embedding Watermarks into Deep Neural Networks. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval (ICMR '17)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3078971.3078974

[23] In-Kwon Yeo and Hyoung Joong Kim. 2003. Generalized patchwork algorithm for image watermarking. *Multim. Syst.* 9, 3 (2003), 261–265. https://doi.org/10.1007/s00530-003-0097-0

[24] Akram Zeki and Azizah Manaf. 2009. A Novel Digital Watermarking Technique Based on ISB (Intermediate Significant Bit). *World Academy of Science, Engineering and Technology* 38 (02 2009).

[25] Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph. Stoecklin, Heqing Huang, and Ian Molloy. 2018. Protecting Intellectual Property of Deep Neural Networks with Watermarking. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security* (Incheon, Republic of Korea) *(ASIACCS '18)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3196494.3196550

[26] Devriş İşler, Elisa Cabana, and Nikolaos Laoutaris. 2022. FreqyWM: Frequency Watermarking for the New Data Economy. *https://dspace.networks.imdea.org/handle/20.500.12761/1626* (2022).