Watermarking of numerical datasets used for ML

A DWT approach for watermarking numerical datasets

Crăciun Marius-Cosmin[1]

Supervisor: Dr. Zekeriya Erkin[1]**, Devriş İşler**[2,3]

[1]**EEMCS, Delft University of Technology, The Netherlands**
[2]**IMDEA Networks Institute**
[3]**Universidad Carlos III de Madrid, Spain**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 18, 2024

Name of the student: Crăciun Marius-Cosmin
Final project course: CSE3000 Research Project
Thesis committee: Dr. Zekeriya Erkin, Dr. Asterios Katsifodimos

Abstract

AI and machine learning have been topics of big interest in the last couple of years, with plenty of applications in many domains. To train these models into useful and desirable tools, a large amount of data is necessary. This data is expensive to collect, becoming one of the most valuable commodities of this century. As the value of data increases, protecting this intellectual property becomes more and more relevant. Watermarking is a technique widely used for data protection in media, but the non-media counterpart has not been researched as thoroughly. In this paper, an adaptation of a common watermarking technique, DWT watermarking, is applied on two datasets used for machine learning. This technique is invisible and robust in signal watermarking, but its performance on a numerical dataset has not been previously researched. A previously devised algorithm was used, but it was adjusted to better fit dataset watermarking. To assess the quality of the watermark, the marked data has been subjected to create, remove, update and zero-out attacks. On top of this, multiple machine-learning models have been trained on the marked data. Initial results show that the proposed technique performs well in terms of invisibility, obtaining similar or better accuracies than models trained on the original data, but it is quite sensitive to attacks. Even small modifications, less than 1% of the data, can break the signature.

1 Introduction

As more and more datasets are utilized for machine learning, it becomes increasingly important to be able to prove ownership of data to prevent the stealing of intellectual property. While there are datasets publicly available, there are not many options, and not all of them are of good quality. As gathering relevant and high-quality data is costly, the companies that invest in this process have a strong incentive to protect the data they share.

Watermarking is a well-known technique used for ownership protection [1], commonly used for protecting audio, images and videos. It consists of two parts, watermark embedding and watermark extraction. Watermarking embedding consists of inserting a secret stream of data in the host data by causing small distortions without compromising the utility of the data. Watermark extraction is the process in which using a secret, ownership of the data can be proved, as well as potentially detect whether the data has been altered.

The proposed approach tries to treat part of the data, an attribute, as another type of media data, such as a signal, to apply a watermarking method used for this type of media. Media watermarking has been researched for longer than its non-media counterpart [2], resulting in plenty of signal watermarking algorithms being proposed which can be used. In particular, this paper will focus on watermarking numerical datasets used for machine learning. A wavelet domain approach will be used for watermarking, in the form of Discrete Wavelet Transform (DWT) watermarking. The current state of research did not use wavelet domain watermarking for watermarking numerical datasets. This process is heavily utilised in image and signal watermarking, as it is a robust and invisible technique. Therefore, the question is if these properties transfer to numerical datasets as well.

In this paper, the performance of DWT watermarking on numerical data will be assessed. This paper outlines the process of testing the robustness of the algorithm to the most common dataset attacks (create, delete, update, zero-out) and presents to what extent the watermark can be extracted. In addition, the model trained on watermarked data will be compared with a model trained on the initial data, to assess the usability of the data after the watermarking.

In Chapter 2 an introduction to the terminology used in the paper will be provided, while in Chapter 3 the watermarking process will be described. In Chapter 4 the experimental setup is presented and the results of the experiment will be portrayed. Chapter 5 will further discuss the results, and address the responsibility of the research. Chapter 6 will provide an overview of the current state of research. Chapter 7 will conclude the paper and suggest potential areas for improvement.

2 Preliminaries

This section briefly presents the building blocks used throughout the rest of the paper. The notations used are provided in Table 1.

Table 1: Notation used throughout the paper.

Symbol	Description
D	Original dataset
K_W	Key used for split selection
K_B	Key used to generate secret bits
D_W	Watermarked dataset
M	The number of splits
N	The length of the split
g_l	Left half of the split
g_r	Right half of the split
μ_l	Mean of the left group
μ_r	Mean of the right group

2.1 Watermarking

Watermarking is a technique used to embed information into various forms of data, usually media, such as images, audio, video, or documents, and can be detected and extracted by appropriate algorithms.

Watermarking is commonly used for copyright protection, authentication or integrity verification. It can be categorized as visible or invisible watermarking, depending on whether the watermark can be easily detected. In the case of images, a visible watermark is usually a logo or piece of text of the owner of the photo. In contrast, in the case of watermarking numerical datasets, this becomes more difficult to define. In this paper, the visibility of the watermark will be defined as

the induced change of mean and variance in the data, as well as the change in the accuracy of an ML model trained on the watermarked data. If the change is small the watermark is considered invisible.

The watermarking techniques can be generally split into two categories, spatial domain techniques and frequency domain techniques [3]. Spatial domain techniques embed the watermark directly into the values or samples of the data. For example, adjusting an image's least significant bits of its pixel values. On the other hand, frequency domain techniques embed the watermark in the frequency coefficients of the media after a transformation, such as the Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), or Discrete Wavelet Transform (DWT). These methods are more robust against common signal processing operations.

2.2 Wavelets

A wavelet is a mathematical function used to decompose a signal into different frequency components with localized resolutions in time and frequency. A wavelet is a wave-like oscillation with an amplitude that begins at zero, increases or decreases, and then returns to zero. This can occur once or multiple times. Wavelets are ideal for analyzing non-stationary or time-varying signals due to their ability to capture coarse and fine details. They are the basis of wavelet transforms, which come in two main types:

- **Continuous Wavelet Transform (CWT):** Provides detailed frequency information.
- **Discrete Wavelet Transform (DWT):** More efficient, commonly used in signal and image processing.

Wavelet families, such as Haar, Daubechies, and Coiflets, have unique properties suited for various analysis tasks.

2.3 Discrete Wavelet Transform

The Discrete Wavelet Transform (DWT) is a mathematical technique used to transform a signal into a set of wavelet coefficients. As with other wavelet transforms, a key advantage over Fourier transforms is that it captures both frequency and time information (location in time)[4]. The fundamental idea of wavelet transforms is that the transformation should allow only changes in time extension, but not shape, imposing a restriction on choosing suitable basis functions. Therefore, wavelet transformation contains information similar to the short-time-Fourier-transformation, but with additional special properties of the wavelets.

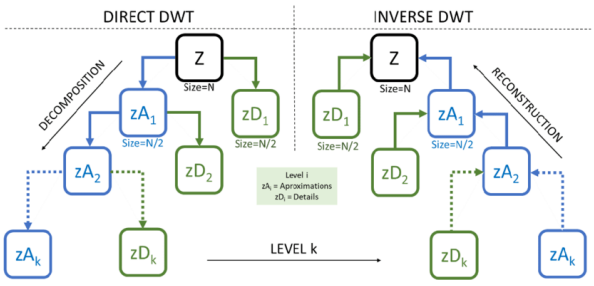

Figure 1: Multilevel DWT and IDWT.

The DWT works by applying a pair of filters (a low-pass filter and a high-pass filter) to the signal and then downsampling the results. The process can be repeated on the low-pass filter output, forming a hierarchical structure of coefficients representing the signal at different scales. The resulting coefficients are called approximation coefficients, for the low-pass filter output, and detail coefficients, for the high-pass filter output.

The Inverse Discrete Wavelet Transform (IDWT) is the method used for reconstructing the signal, using a list of coefficients. A multilevel IDWT functions by repeatedly combining the coefficients and using the result as the approximation component of the next level, as seen in Figure 1.

2.4 Machine Learning

The significance of AI in our daily lives is increasing. From image classification by search engines to autonomous driving and speech recognition, AI applications are widely spread. Machine learning, a key component of AI, involves developing and studying statistical algorithms that can learn from data and generalize to unseen data.

The following classifiers have been utilised to verify the visibility of the watermark:

- **Logistic Regression** predicts the probability of a binary outcome using the logistic function, mapping the input to a probability between 0 and 1.
- **K-Nearest Neighbours(KNN)** predicts the class of a data point by considering the classes of its nearest neighbours in the feature space.
- **Support Vector Machine(SVM)** finds the hyperplane that best separates different classes in the feature space, maximizing the margin between classes while minimizing classification errors.
- **Decision Tree** makes predictions by recursively splitting the feature space into regions, based on feature values. Each split is chosen to minimize impurity or maximize information gain.
- **Random Forest** is a learning method consisting of multiple decision trees. It combines predictions from each tree to make a final decision, reducing overfitting and improving generalization.

3 Watermarking numerical data

It is important to remember that during watermarking, the data will be altered to allow the owner to extract information from it. As such, the attribute must be chosen in a way that does not damage the quality of the original dataset. Also, due to the nature of the embedding technique, the embedded attribute should have a continuous value, as large steps in the data can negatively influence the DWT. For these reasons, in this paper, the attribute with the lowest variance was used for watermarking both of the datasets.

The watermarking process is visualized in Figure 2. The watermarking relies on 2 different keys. These keys are used to generate streams of pseudo-random numbers. K_W is used to generate the indices of the segments at which the watermarking will take place, while K_B is used to generate a bit

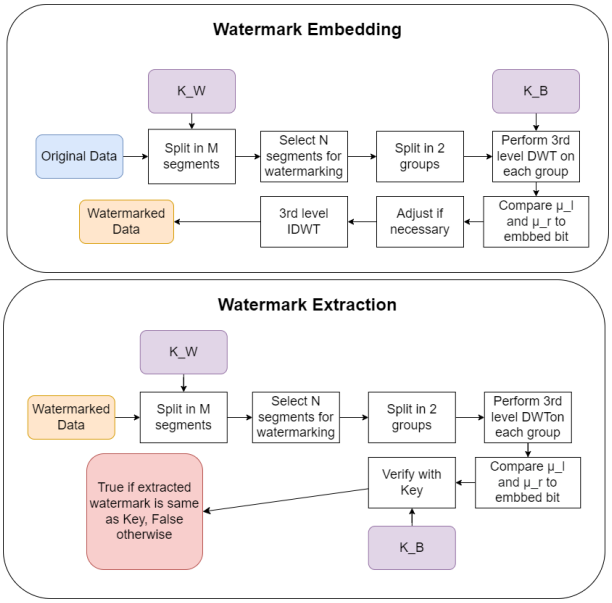

Figure 2: Watermarking Process.

string that will be embedded into the dataset. With the help of these two keys, someone will be able to prove ownership of a given dataset. As these numbers can be any positive integers, it is unlikely that someone else will be able to guess a matching combination.

3.1 Watermark embedding

The dataset D is split into M segments, preparing it for watermarking. Using a secret key K_W, N segments are chosen from these M segments to be subjected to watermarking. Each selected segment is then split into two groups. Previously, this split was based on the parity of the indexes [5]. However, this method was extremely fragile to delete and create attacks. To counter this, the split is done by separating the data into the first $n/2$ (g_l) and the second $n/2$ elements (g_r), where n is the length of the segment.

In the next step, a 3-level Discrete Wavelet Transform (DWT) is applied to each half. The DWT decomposes the data into different frequency components, allowing for a multi-resolution analysis. If a group has too few entries to support a 3-level DWT, a lower-level DWT is used instead. To embed a watermark bit, the means of the approximation coefficients (the low-frequency components) of the two halves are compared. If the mean of the left half (μ_l) is less than the mean of the right half (μ_r), a 0 is embedded. Otherwise, if μ_l is greater than μ_r, a 1 is embedded. If the result of the comparison does not match the desired outcome, the means are adjusted to ensure the intended bit is embedded. The embedded bit is generated by K_B.

After embedding the bit, an Inverse Discrete Wavelet Transform(IDWT) of the same level is applied to each half to reconstruct the watermarked segments. This combines the modified frequency components back into the time-domain signal, resulting in the watermarked version of the original data segment. The watermarked segments are then combined to form the watermarked dataset D_W, which retains the watermark information embedded during the process.

3.2 Watermark extraction

Extracting the watermark from the data follows a process similar to the embedding. First, the dataset D_W is split into M segments. Then, using the secret key K_W, N of these M segments are selected for verification. Each selected segment is split into two groups, g_l and g_r, where g_l contains the first $n/2$ elements and g_r contains the last $n/2$ elements, with n being the length of the segment.

Next, a 3-level DWT is applied to each half. If a group does not have enough entries for a 3-level DWT, a lower-level DWT is used instead. The means of the approximation coefficients (the low-frequency components) are then calculated and compared. If μ_l is less than or equal to μ_r, a 0 is considered embedded into the data; otherwise, a 1 is extracted.

The results from all the segments are then compared with the bits generated by the secret key K_B. If the extracted watermark matches the data generated by the key, the algorithm returns true, indicating that the data belongs to the person with access to the keys. If the algorithm returns false, the ownership of the data cannot be proven, indicating a failure in the extraction.

4 Experimental Setup and Results

The experiments were run on a laptop with an Intel(R) Core(TM) i7-10750H CPU at 2.6 GHz and 16 GB of RAM, operating Windows 11 Pro. All of the code used for the experiments is written in Python. The code uses `PyWavelets` [6] library for the wavelet transforms, and `scikit-learn` for the machine learning algorithms. Two datasets are used for the experiments, the iris dataset and the dry bean dataset. The iris dataset is available through the `scikit-learn` library, while the dry bean dataset can be found in the UCI archive [7].

The **iris dataset** consists of 150 records, each featuring 5 distinct attributes: 1) sepal length, 2) sepal width, 3) petal length, 4) petal width, and 5) species. This dataset is commonly used for pattern recognition and classification tasks.

The **dry bean dataset** includes around 13,600 entries, with 7 different types of dry beans. Each record in this dataset is characterized by 16 attributes, with 12 attributes detailing various dimensions and the remaining 4 attributes describing different shape forms.

The watermarked dataset is tested on the robustness and invisibility of the watermark. For watermarking, the iris dataset is split into 10 segments(M) and 3 of them are selected for watermarking(N). The large sample size of the dry bean dataset allows for a larger number of splits, therefore the data is split into 50 segments, out of which 25 will be selected for the watermark.

For the following experiments, a third-level DWT was used, although this can be adjusted based on the size of the data. The secret keys used for embedding were $K_B = 271124$ and $K_W = 274853$. These numbers were chosen arbitrarily and used with the numpy randomizer.

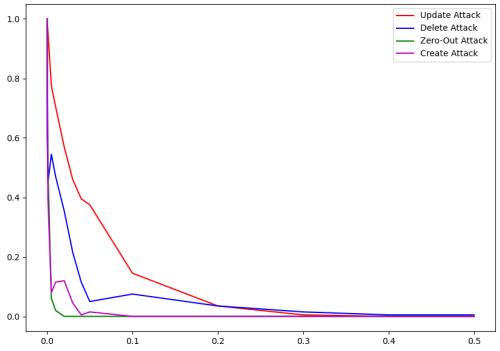

Figure 3: Successful extraction rates after 200 attacks.

4.1 Robustness Analysis

The robustness of the watermark is tested by subjecting the watermarked dataset D_W to common dataset attacks. The attacks considered for this experiment were insertion, deletion, update and zero-out attacks. Since only one of the attributes is watermarked, the robustness attacks will only target that attribute. The dry bean dataset was used for these experiments since the technique is targeted to larger datasets.

Create attacks

The create attack tries to destroy the watermark by inserting new tuples into the dataset. The new values have been generated using knn and iterative imputers available through the `scikit-learn` library. In Figure 3 it can be observed that the proposed scheme is very fragile to create attacks. As new elements are added to the dataset, splitting the data into M segments becomes less consistent, resulting in different selections of elements and therefore different coefficients.

Deletion attacks

The deletion attack attempts to distort the watermark by removing entries from the dataset. The approach suffers from the same problem that create attacks have, the change of dataset size. Datasets of different sizes will be split into different segments.

Update attacks

The update attack tries to damage the watermark by changing the value of some of the entries. From Figure 3 it can observed that the scheme does well against attacks that influence less than 1% of the database, although the watermark is not very resistant to larger attacks.

Zero-out attacks

The zero-out attack updates some tuples in the dataset with the value 0. As this can largely influence the approximation coefficients the method is not particularly resistant to this type of attack.

Figure 3 shows the rates of successfully extracting a watermark after 200 attacks. It can be noticed that the proposed scheme is not robust when more than 1 % of the data is affected. The chances of an update attack affecting the mark are considerably lower than the other types of attacks proposed.

This can be improved by increasing the change that the watermarking imposes on the dataset. If the difference between the means of the approximation coefficients is made larger, the robustness of the watermark increases, while increasing its visibility. In Figure 4 the change in data is presented and in Figure 5 the new robustness performances can be observed. In this case, the change in variance becomes 1.0610%, which is around 10 times larger than the previous alternative. Another option for improving the robustness of the watermark is reducing the number of segments the data is split into. This will result in a smaller watermark, but the length of each segment will increase, resulting in more data being used for the transform. One last option for improving the chances of a successful extraction is reducing the percentage of the data used for watermarking. This would also reduce the size of the mark, but it would also reduce the chance that an attack hits a relevant section of the dataset.

Table 2: Mean and variance before and after watermarking.

	Iris dataset	Dry Bean Dataset
μ_D	3.057333	1.715947e-03
μ_{D_W}	3.066235	1.716670e-03
$\Delta\mu$	8.901760e-03	-7.221967e-07
$\Delta\mu\%$	0.2912	0.0421
σ_D	0.188713	3.550408e-07
σ_{D_W}	0.188825	3.547514e-07
$\Delta\sigma$	1.116753e-04	2.893777e-10
$\Delta\sigma\%$	0.0592	-0.0815

4.2 Invisibility of the Watermark

The invisibility of the watermark can be assessed on two metrics. The distortions the watermarking imposes on the dataset, and to what extent the watermark affects the training of the ML model.

Effect of watermarking on the dataset

The watermarking targets the attribute with the lowest variance, therefore there are no changes in the rest of the dataset. This is the ”sepal width” attribute for the iris dataset and the ”shape factor 2” attribute for the bean dataset. Table 2 shows the difference in mean and variance of the selected attributes for both of the datasets, and it can be observed that the distortion induced by the watermarking is minimal.

In Figure 6 the changes imposed by watermarking on the iris dataset can be observed. The first graph shows the original dataset, the second one displays the watermarked data and the last one represents the difference between the two datasets. It can be seen where the embedding caused a change in the data, although no conclusions about whether the data is marked can be drawn without the original dataset. Figure 7 shows the dry bean data before and after the watermarking. The changes imposed by the watermark are not major, resulting in a similar dataset after the watermarking process.

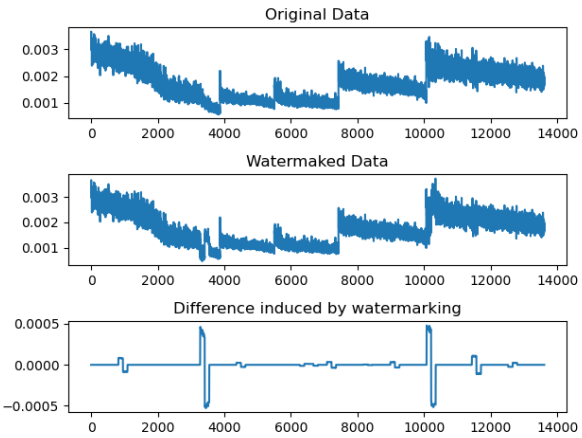

Figure 4: Change in data when trying to improve robustness.

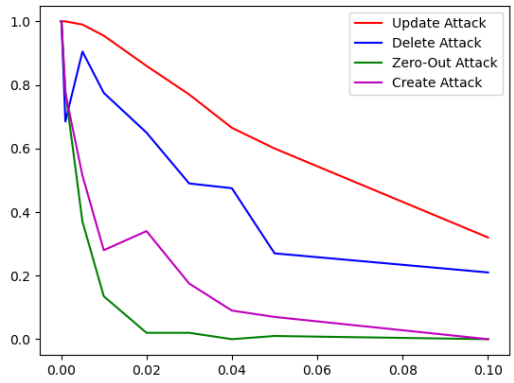

Figure 5: Successful extraction rates after 200 attacks.

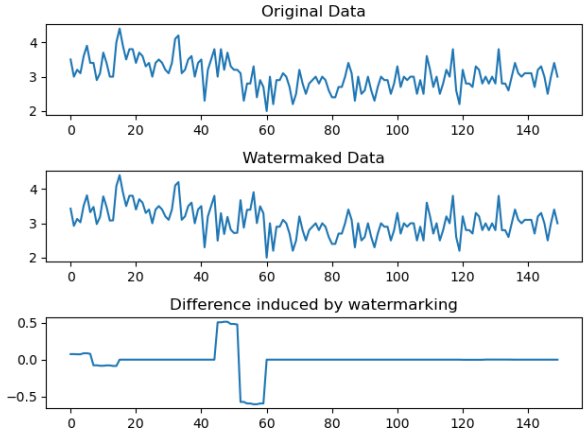

Figure 6: Iris data.

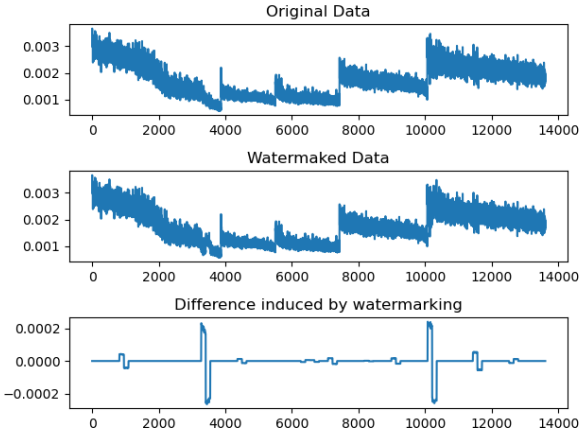

Figure 7: Dry bean data.

Effect of watermarking on the ML model

For testing the visibility of the watermark, the watermarked dataset has been trained on multiple models to test for potential changes in the accuracy of the trained model. Therefore multiple models from the `scikit-learn` library have been used.

For the training of the models, hyperparameter tuning has been used. Hyperparameter tuning involves optimizing a model's settings to improve its performance. For all models, grid search was used as the search strategy, 5-fold cross-validation was used to avoid overfitting and accuracy was used as a metric to evaluate their performance.

From Table 3 it can observed that the watermarking does not impact the quality of the data. The accuracy of all models trained on watermarked data is similar to the accuracies obtained from training the models on the original data. Even when using all of the data for watermarking, for the purpose of embedding more information, the models obtained after training perform similarly to the models trained on the initial data.

5 Responsible Research

Ensuring that results are reproducible is an important part of the research, as it helps future researchers validate and continue the work [8]. As such, the code and datasets used in this paper are publicly available. Any qualified researcher can obtain any of the presented results by adjusting several parameters. The code is available at https://github.com/MariusCosmin03/DWTWatermarking.

While the watermarking is invisible to the training models, it should not be misused. Watermarking assumes that small changes to the data are acceptable. Marking sensitive data can lead to the loss or damage of important information, which can reduce the quality of the dataset. Therefore the attribute selected should allow it as well. Having knowledge about the dataset can help the owner make an informed decision when selecting the attribute used for watermarking.

6 Related work

In contrast to the large amount of research done on media watermarking, watermarking non-media data is a newer branch

Table 3: Accuracy of models trained on watermarked data.

Half the data is considered for watermarking.

ML Model	Training Acc % 2	Testing Acc %	Training Acc WM %	Testing Acc WM %	Δ Acc Test
Log Regression	92.411042	92.77669	92.264092	93.217434	0.440744
KNN(n=6)	93.954025	92.286974	93.723103	92.801175	0.514201
KNN(n=20)	92.809909	92.482860	92.694447	92.507346	0.024486
SVM	93.649627	93.119491	93.733599	93.682664	0.563173
Decision tree	92.946363	91.111655	93.198279	90.205681	-0.905974
Random Forest	97.753752	92.482860	96.084812	96.376102	3.893242

All the data is considered for watermarking.

ML Model	Training Acc % 2	Testing Acc %	Training Acc WM %	Testing Acc WM %	Δ Acc Test
Log Regression	92.358560	92.507346	88.579826	88.809990	-3.697356
KNN(n=6)	93.954025	92.286974	93.744096	92.825661	0.538687
KNN(n=20)	92.809909	92.482860	93.166789	93.241920	0.75906
SVM	93.649627	93.119491	94.205941	94.343781	1.22429
Decision tree	92.946363	91.111655	94.636297	91.625857	0.514202
Random Forest	97.753752	92.482860	95.979847	96.131244	3.648384

of watermarking with plenty of room for improvements [2]. The first database watermarking was done by Agrawal et al. [9] in 2002. Since then, a couple of attempts have been made at watermarking databases and numerical datasets. Another proposed scheme for conserving the mean and the variance of watermarked data is to employ a statistical approach based on sampling Gaussian distributions [10]. The proposed scheme was only tested against noise attacks, so the robustness against other types of attacks remains unknown.

The algorithm used in this paper is inspired by the previous work of Pham et al. [5], where the authors are suggesting an algorithm for watermarking time series. The proposed algorithm is a simplified version of the earlier work [5], as it does not train an ML model for watermark extraction. The paper's main contribution relies on applying such an algorithm to a numerical dataset used for training ML models.

7 Conclusions

In this paper, a DWT approach of watermarking [5] was applied to machine-learning data and its performance was studied. The technique was applied to two ML datasets using a DWT-signal watermarking process, fit for numerical data.

The analysis shows that watermarking did not introduce relevant distortion to the original dataset. As a result, the machine learning models trained on this data had high accuracy rates, similar to the original data. Additionally, the resilience of the proposed method to update, deletion, zero-out and creation attacks was also examined. Initial results show that the use of DWT watermarking techniques is viable for machine learning data without compromising accuracy rates. However, as shown, the procedure needs future improvements in terms of robustness before it can be utilized for professional use.

Future Work

In this paper, it has been shown that the proposed method can be applied to machine learning datasets without degrading the quality of the data. However, there is still room for improvement in the proposed method. To make the method resistant to reordering of the data, a sorting method can be used by combining one or more other features of the dataset, as sorting on the watermarked attribute will affect the watermarking process. Increasing the size of the mark could involve embedding multiple attributes of the dataset, which was not experimented with. Furthermore, enhancing the robustness of the watermark remains an area for improvement. One approach could be to embed the watermark multiple times and use a majority voting system during the extraction process.

References

[1] S. Kumar, B. K. Singh, and M. Yadav, "A recent survey on multimedia and database watermarking," *Multim. Tools Appl.*, vol. 79, no. 27-28, pp. 20 149–20 197, 2020. [Online]. Available: https://doi.org/10.1007/s11042-020-08881-y

[2] A. S. Panah, R. Van Schyndel, T. Sellis, and E. Bertino, "On the properties of non-media digital watermarking: a review of state of the art techniques," *IEEE Access*, vol. 4, pp. 2670–2704, 2016.

[3] M. Saqib and S. Naaz, "Spatial and frequency domain digital image watermarking techniques for copyright protection," vol. 9, pp. 691–699, 06 2017.

[4] I. Daubechies, *Ten Lectures on Wavelets*. SIAM, 1992. [Online]. Available: https://doi.org/10.1137/1.9781611970104

[5] T. D. Pham, D. Tran, and W. Ma, "An intelligent learning-based watermarking scheme for outsourced biomedical time series data," in *2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017*. IEEE, 2017, pp. 4408–4415. [Online]. Available: https://doi.org/10.1109/IJCNN.2017.7966414

[6] G. R. Lee, R. Gommers, F. Wasilewski, K. Wohlfahrt, and A. O'Leary, "Pywavelets: A python package for

wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, 2019. [Online]. Available: https://doi.org/10.21105/joss.01237

[7] "Dry Bean," UCI Machine Learning Repository, 2020, DOI: https://doi.org/10.24432/C50S4B.

[8] K. Y. Rozier and E. W. D. Rozier, "Reproducibility, correctness, and buildability: The three principles for ethical public dissemination of computer science and engineering research," in *2014 IEEE International Symposium on Ethics in Science, Technology and Engineering*, 2014, pp. 1–13.

[9] R. Agrawal and J. Kiernan, "Watermarking relational databases," in *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Elsevier, 2002, pp. 155–166.

[10] F. Sebé, J. Domingo-Ferrer, and A. Solanas, "Noise-robust watermarking for numerical datasets," in *Modeling Decisions for Artificial Intelligence, Second International Conference, MDAI 2005, Tsukuba, Japan, July 25-27, 2005, Proceedings*, ser. Lecture Notes in Computer Science, V. Torra, Y. Narukawa, and S. Miyamoto, Eds., vol. 3558. Springer, 2005, pp. 134–143. [Online]. Available: https://doi.org/10.1007/11526018_14