|  |  |
| --- | --- |
| . | Moving Picture, Audio and Data Coding  by Artificial Intelligence  www.mpai.community |

|  |  |
| --- | --- |
| **N567** | 23/02/08 |
| **Source** | Requirements (NNW) |
| **Title** | State of the art review of Neural Network Watermarking |
| **Target** | MPAI Community |

# Introduction

This document presents a state-of-the-art analysis of the solutions related to Neural Network watermarking. Its purpose is to contribute to the MPAI efforts in the field of NN watermarking standardization and has a three-folded objective:

* Reflect the status of methods and identify methodological trends in research studies,
* Identify a common ground of methodological and applicative features relevant for standardization,
* Draw an overview of potential use cases (beyond the conventional ownership identification) which will be developed in M842.

To this end, for each analysed method, our document describes the purpose (as stated by the author), presents the embedding/retrieval processes, and comments its usage for MPAI-NNW.

# Overview of methods

The simplest taxonomy for NNW is binary and includes white box and black box classes:

* white-box methods grant access to the network and make it possible for the watermark to be embedded inside the parameters; in this case, users have a fully access to the network (architectures, trained parameters) and could modify it (compression, fine-tuning…)
* black-box methods do not grant access to the network but to its output (result of an inference); in this case, the watermark information is inserted inside the output.



Figure 1. Summary table from the basic idea to the methods

## White Box methods

### Digital Watermarking for Deep Neural Networks [1]

1. Purpose

This is one of the first techniques embedding a watermark inside the parameters. The goal is to develop the interest around security and intellectual property of neural networks. This method embeds a watermark inside the parameters of one convolutional layer and is robust against fine-tuning and compression.

1. How it works?

Let (S, S), D, and L respectively denote the size of the convolution filter, the depth of input to the convolutional layer, and the number of filters in the convolutional layer. A convolutional layer *W* S×S×D×L. Before embedding the binary watermark *b* of lengthT*,* we flattened W according to the L dimension *W* S×S×D = . Second, we create a secret key matrix X of size with and X follows a normal distribution *N*(0,1).

To embed the watermark, we add a regularization parameter to the original cost function:

1. Comments

In this method it is mandatory to train the network (from scratch or by fine tuning it with the regularization term) which need an access to the original dataset and computational resources. Also, this method allows to insert a multi-bit watermark.

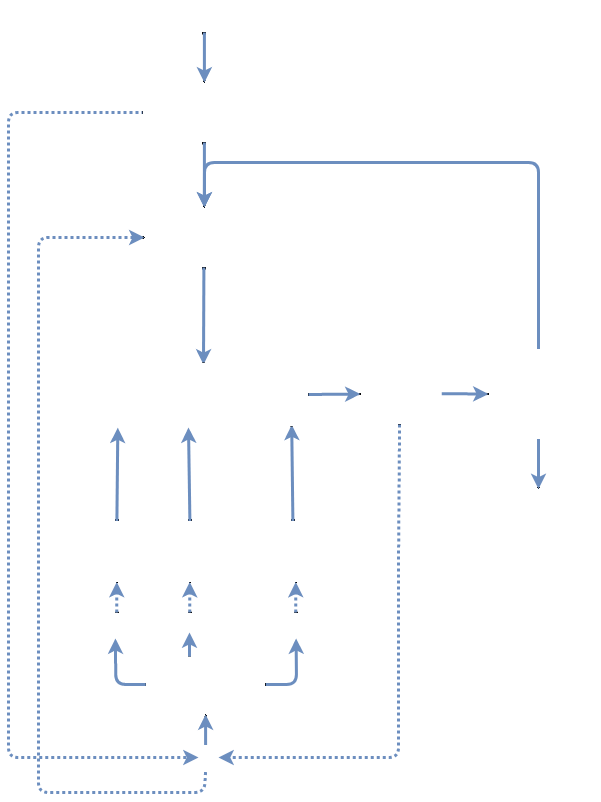
### Delving in the loss landscape to embed robust watermark in neural networks [2]

1. Purpose

Take advantages of the redundancy and adaptation capacity of DNN to lock a subset of watermarked parameters. We want to protect those watermarked parameter against fine tuning and compression by forcing them to have a significant impact the cost function if they change.

1. How it works

Unlike the earlier method the parameters are watermarked before training. To ensure the embedding of our watermark the training process is changed, we will note Γ0 the original network:



* Compute the original loss on Γ0.
* Then we add noise to our watermarked weights to create other networks {Γ1, …, ΓR}. We compute the loss and want to maximize it.
* We then compute both loss and update the non-watermarked weights
* Repeat

The algorithm stops when loss of Γ0 – the original loss – matches our expectation.

1. Comments

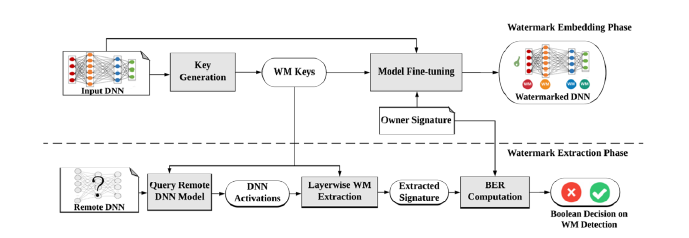
This method is interesting because the watermark is inserted before the training phase but in the other hand, we need computational resources because we store copies of the network (ΓR) to compute.

### DeepSigns: An End-to-End Watermarking Framework for Ownership Protection of Deep Neural Networks [3]

1. Purpose

DeepSigns create a complete framework for watermarking. DeepSigns framework take your network and return a watermarked version of it with its key. Furthermore, this solution is a combination of dynamic white box solution (multi-bit) and a black box solution (one-bit watermark).

1. How it works ?



In DeepSigns, the behavior of the watermarked activation map – note – is defined by forcing the distribution of the map to follow a Gaussian Mixture Model for which the mean values – note – of the Gaussian probability density functions satisfy certain conditions which in turn decide the

embedded bits. The key images (or trigger set) are a subset of only one class and will have a specific behavior on the output (Blackbox case). To ensure these 2 additional losses are created:

* the aim is to let the distribution of the activation map hosting the watermark to be closed to the gaussian distribution.
* is computed only for key images: similar to Uchida’s idea but only for key images.

mu denotes the gaussian distribution ; represents the activation map; b our binary watermark and with A analogous to X in Uchida et al.

1. Comments

This method is available as a module in Python.

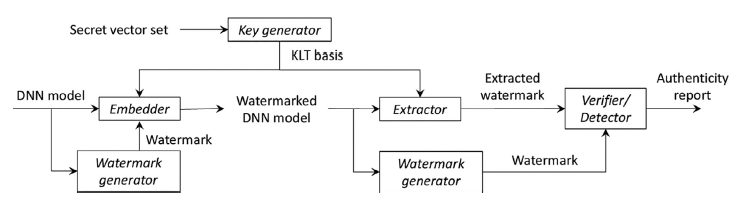
### NeuNAC: A novel fragile watermarking algorithm for integrity protection of neural networks [4]

1. Purpose

Neural Network Authenticity Checker (NeuNAC) embeds a fragile watermark (bit string): the watermark will be altered even if only one parameter changed.

1. How it works ?

NeuNAC embeds the watermark in the frequency domain using Karhunen-Loève Transforms. Compared to DCT, which have predefined kernel, the KLT kernel are designed using a set of vectors. The 2nd main tools are the Genetic algorithms which permits to solve a problem by emulating the evolution of a population according to a fitness function.



* **Key Generator:** using an image – which stays secret – we create a KLT orthonormal basis (n=32) for kernel.
* **Parameter Unit (PU):** parameters are grouped into block of S=16 elements.
* **Watermark Embedding Unit (WEU):** given a PU, a WEU is composed by the 3 most significant bytes (MSB) of each parameter value represented by a 16bits vector using a message digest algorithm (MD5) and the 16 parameters’ least significant bytes(LSB). We note
* **Watermark Generator**: we generate a bit string *w* function of DNN parameters. Each WEU is assigned a part of the generated watermark.
* **Embedder**: compute one WEU at a time and independently. We note : . The Genetic algorithms change the last 16 bits of WEU in order that our 32 coefficients ( embeds the watermark. The GA’s function is defined as follow with , the modification of the WEU & () returns the number of watermark bits correctly resent into the coefficients.
* **Extractor**: by using the same key generated in earlier steps we extract the coefficients all WEU and concatenate them.
* **Verifier**: take the result of the extractor and compare it the original *w* obtain with the watermark generator.

1. Comments

Results of this method were evaluated on classification problems and on one text generation problem using a LSTM. In addition to the loss performances comparison, they focus on adversarial images to detect differences between the original DNN and the watermarked one.

## Black Box methods

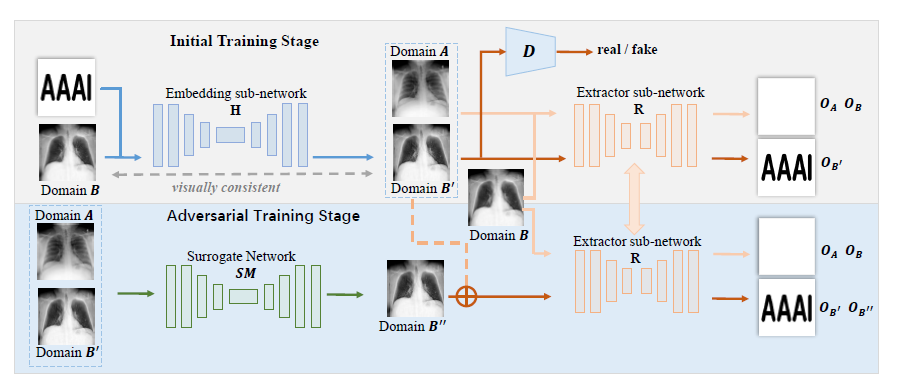
### Model Watermarking for Image Processing Networks [5]

1. Purpose

This technique is a module than can be added after a Networks (cascaded). It has two modules: one who embed the watermark without affecting the output & the another which is trained to retrieved it. It is designed to resist to surrogate attacks which mean that someone train his surrogate model by using the output of our network as label.

1. How it works ?

Let *M* denotes our model: *A* & *B* stands for, respectively, the initial domain and the output domain, so *M*(*A*) is in the *B* domain. To a better understanding of this method, we will take the example of an image processing network: *M* erase the ribs of an X-ray images. We could imagine other example including speech or video processing.



-The Embedding sub-network *H* (U-Net like) is trained to embed a watermark (B&W image of ‘AAAI’) inside the image of domain *B*: the output domain of is . We want *B*’ to be as closed as possible of i: to ensure that another network D (Patch-GAN) is used as a discriminator*.*

-The Extractor sub-network *R* (CEIL-Net) is trained to retrieve the watermark. To protect our solution from surrogate attack they add an adversarial training stage to the framework. A surrogate network *SM* (U-Net too) is trained with *B’* images as labels; the result domain of *SM* is Thus, by training R on B’ & B” we increase the detection capability of R.

1. Comments

The training process is applied on the module we do not have to modify the neural network that we want to watermark. It is more like adding a signature to our network “made by.”

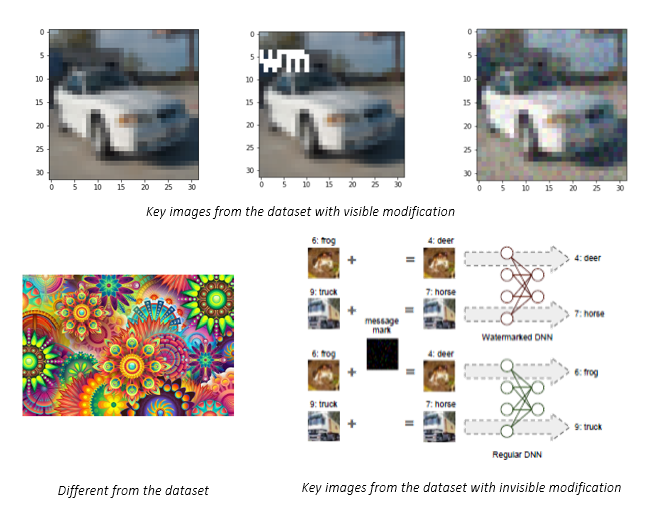
### Watermarking of DNN backdoor

1. Purpose

We want to take advantages of a known weakness: backdoor attacks. A backdoor attack installs a backdoor pattern – unexpected behavior - into a small proportion of the training dataset.

1. How it works

A first method is to add other content to our dataset (unrelated): those elements are kept secret by the author. The other method is to alter a sub-set of elements inside the dataset. Below are examples for a classification problem on CIFAR10 dataset, the trigger set will be composed of twenty images:



First tuple of images shows internal keys [7], the second show external elements [6] and the last one shows an invisible mark on the input [8].

On the bottom right we can see the process of this method. A non-watermarked network will correctly watermark the image contrarily our watermarked network.

1. Comments

Those methods also act like a “signature” but do not need an extra module.

### PREDICTION POISONING: TOWARDS DEFENSES AGAINST DNN MODEL STEALING ATTACKS [9]

1. Purpose

As a black-box case, attacker party has access to a probability distribution over K classes with M the model and x an input. They aim to protect the network against knowledge distillation attacks by poisoning actively the training aim of the attacker.

1. How it works ?

Une image contenant texte

Description générée automatiquement

We compute the gradient of the loss (first order) with the model parameters. We will note our poisoned output and , knowing that the attacker model is trained to match our posterior predictions using cross entropy we can write with the Jacobian over log-likelihood predictions M(x) over K classes. Maximizing the angular deviation between and is equivalent to :

So, the full process is :

* Obtain the original output
* Estimate the Jacobian over log-likelihood matrix G. they chose a VGG16 architecture initialize randomly. This decision was made after doing empirical research.
* Search for a maximizer y\* for H() by iterating over the K extremes (where )
* Compute as a linear interpolation of the original posteriors y and the maximizer with selected such as the constraint are satisfied.

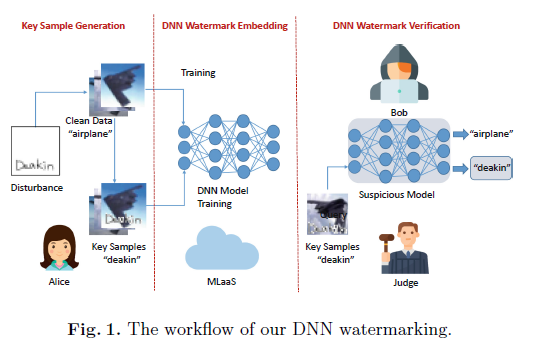
1. Comments

Evaluated on multiple classification dataset, they use latest distillation attack strategies and show that their approach resist to it without impacting the performances.

### Protecting IP of Deep Neural Networks with Watermarking: A New Label Helps [10]

### Purpose

This method also uses backdoor attacks, which mean associate inputs to a specific unexpected label. But the “unexpected” label is a new class.

1. How it works ?

The main characteristic is to change the dataset and train our network on it. But this time instead of randomly associate a label to our trigger set we label them as an additional label. Additionally, the selected inputs are taken from the original dataset but altered, with a visual watermark for example. Thus, a non-watermarked network will correctly label them while our will label them as the “extra class.”

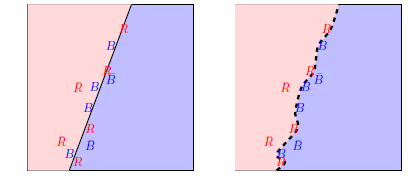
1. Comments

This method uses the same principle of Y.Adi et al. [6] and the others [7,8] but add another class to prevent the loss of accuracy which is interesting.

### Adversarial frontier stitching for remote neural network watermarking [11]

1. Purpose

Adversarial case means that there are near to a decision frontier for the network, we can see it with the above example on a two classes classifier Red Blue. The aim is to do what is shown on the right, stitch the frontier around particular “adversaries” images. To preserve the accuracy, we want them to be correctly classified.



1. How it works?

This method embeds the watermark by fine-tuning a pre-trained model so that the boundary of the classification region assumes a desired shape: around a set of adversarial examples. To create Adversarial images are created using IFGSM algorithm and divided in two categories: false adversaries (are truly labeled by the network) and true adversaries (are wrongly labeled), we want the same number of images in both categories. Then the network is fine-tune on those images. A non-watermarked network has an equal probability to correctly classify the images or not, while our watermarked network is going to classify all (most) of them correctly.

1. Comments

They evaluate it only on MNIST dataset which is considered as the easiest case and obtain reliable results.

### Secure neural network watermarking protocol against forging attack [12]

### Purpose

This method proposes a secure protocol compatible with various watermarking algorithms (black box cases). To do that they introduce a one-way hash function to avoid forging attacks, an attacker could not create another trigger set.

1. How it works ?

To resist the forging attack, we introduce two hash function, one which take an image and return an image and another which return an integer value. Thus, instead of selecting a trigger dataset like [6,7,8] we create it using hash functions and , then the training process remain the same.

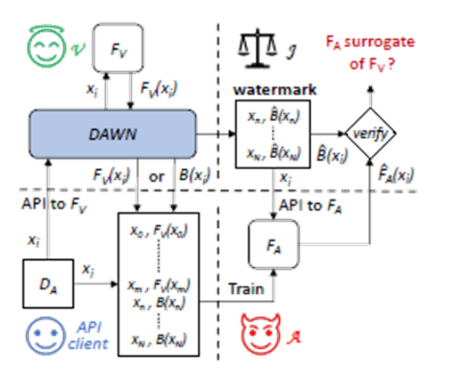
1. Comments

One of the first paper which introduce cryptography and the security importance of protecting the trigger set.

### DAWN: Dynamic Adversarial Watermarking of Neural Networks [13]

1. Purpose

DAWN is a specific Blackbox case which protect the Neural Network through an API: we do not have access to it, we just give an input and obtain an output. DAWN aims to protect a network against knowledge distillation by adding a module in the API.



1. How it works ?

DAWN affect the output of a sub-set of inputs (like a backdoor attacks) but instead of modifying or doing an additional training to the network they will do it on the API. Thus, if an attacker tries to do a Knowledge distillation attacks, the surrogate model will learn those errors and it would be possible to claim the ownership.

1. Comments

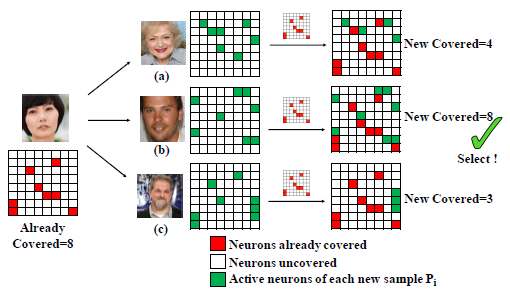
DAWN is a good applicative solution if the solution is used on an API.

### VerIDeep: Verifying Integrity of Deep Neural Networks through Sensitive-Sample Fingerprinting

1. Purpose

The objective of VerIDeep is to ensure that the deployed model’s integrity is protected. To do that, the method find *Sensitive-Sample fingerprints* which are a small set of inputs which are highly sensitives to the model’s parameter.

1. How it works?

An owner has a network ( represents parameters): if an input, the output . We want to make sure that the uploaded network remain the same. The Sensitive-Sample denotes as and the fingerprint is .

* Create Sensitive-Sample which are similar to a normal input and correctly labeled [optimization problem]
* Maximum Active-Neuron Cover uses the previously created Sensitive-Sample to create a set of sensitive images each of them activates the maximum of parameters [maximum coverage problem]

The final number depends on the black box case but they managed to never go bellow 10 inputs to achieve 99.5% of detection.

1. Comments

Compared to adversarial examples, *Sensitive-Sample* do not give the wrong output, it changes with the model parameters: . This method is a particular case of watermarking: fingerprinting

# Methodological and application features addressed by the state-of-the-art

## Imperceptibility

Most of the work (10 out of 13) are evaluated on classification problem. Since Uchida et al. [1] the definition is : “Prediction quality of the model on its original task should not be degraded significantly.” Thus, currently a common definition of Imperceptibility independent of the task and applicable on all field does not exist. “Model Watermarking for Image Processing Networks” is specific to image processing and use a Neural Network to embed its watermark. Thus, PSNR and SSIM are used to evaluate the visual quality.

## Robustness

### Definition

Borrowed from multimedia watermarking, attacks against a watermarked content could be divided in four categories:

**Removal attacks**: the aim is to remove the watermark by extracting it or making it disappear.

**Geometric attacks:** do not try to remove the mark, but rather to destroy their synchronization.

**Cryptographic attacks**: detect and remove the mark without knowledge of the key, thanks to the knowledge of the embedded mark.

**Protocol attacks**: the aim is to embed another watermark and create an ambiguous situation.

### Robustness to attacks

Most of the work (10 out of 13) are evaluated on classification problem. Since Uchida et al. [1] the definition is “Watermark should be robust against removal attacks.” Here is a list of removal attacks: Fine-tuning ; Transfer learning ; Quantization ; Pruning ; Permutation ; Knowledge distillation. Another type of attacks borrow from multimedia watermarking is the watermark overwriting and watermark forging.

To evaluate robustness, the probability of watermark detection needs to be high while the probability of false alarm remain law.

## Data Payload

Most of the work (10 out of 13) investigated implement 0-bit watermark: do we recover the watermark or not. Only the first two white box methods proposed multi-bit watermarking. The size data payload depends on the Neural Network architecture. The goal is to obtain a format generalizable to any architecture/task.

# Experimental testbed

## Preliminaries

NNW is a new field which means that techniques are evaluated and implemented in various manners. The goal our work is to create a benchmark/laboratory where all the watermarking method are embedded and evaluated under the same conditions.

As developed in earlier section, classification problem is predominant, we will use CIFAR10 as the training dataset and MNIST as the second dataset for transfer learning [annex 2]. Currently, we have implemented half of the techniques on VGG16 architecture. Concerning attacks, we have implemented : Fine-tuning ; Transfer learning ; Quantization ; Pruning ; Permutation.

## Results

The results in the table are for a classification problem on CIFAR10 with a VGG16 architecture train from scratch. ✅ and ❌ mark if the techniques resist to the attack or not.



**References:**

[1] Nagai, Y., Uchida, Y., Sakazawa, S., Satoh, S., 2018. Digital watermarking for deep neural networks. International Journal of Multimedia Information Retrieval 7, 3–16. doi:10.1007/s13735-018-0147-1.

[2] E. Tartaglione, M. Grangetto, D. Cavagnino, and M. Botta, “Delving in the loss landscape to embed robust watermarks into neural networks,” in 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, Jan. 2021, pp. 1243–1250. doi: 10.1109/ICPR48806.2021.9413062.

[3] Rouhani, B.D., Chen, H., Koushanfar, F., 2019. Deepsigns: an endtoend watermarking framework for protecting the ownership of deep neural networks, in: The 24th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS). ACM.

[4] M. Botta, D. Cavagnino, and R. Esposito, “NeuNAC: A novel fragile watermarking algorithm for integrity protection of neural networks,” Information Sciences, vol. 576, pp. 228–241, Oct. 2021, doi: 10.1016/j.ins.2021.06.073.

[5] Zhang, J., Chen, D., Liao, J., Fang, H., Zhang, W., Zhou, W., Cui, H., Yu, N., 2020. Model watermarking for image processing networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Association for the Advancement of Artificial Intelligence (AAAI).

[6] Adi, Y., Baum, C., Cisse, M., Pinkas, B., Keshet, J., 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring, in: 27th USENIX Security Symposium, pp. 1615–1631.

[7] Zhang, J., Gu, Z., Jang, J., Wu, H., Stoecklin, M.P., Huang, H., Molloy, I., 2018. Protecting intellectual property of deep neural networks with watermarking, in: Proceedings of the 2018 on Asia Conference on Computer and Communications Security, pp. 159–172.

[8] Guo, J., Potkonjak, M., 2018. Watermarking deep neural networks for embedded systems, in 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), IEEE. pp. 1–8.

[9] T. Orekondy, B. Schiele, and M. Fritz, “Prediction Poisoning: Towards Defenses Against DNN Model Stealing Attacks,” presented at the International Conference on Learning Representations, Sep. 2019. Accessed: Feb. 11, 2022.

[10] Q. Zhong, L. Y. Zhang, J. Zhang, L. Gao, and Y. Xiang, “Protecting IP of Deep Neural Networks with Watermarking: A New Label Helps,” in Advances in Knowledge Discovery and Data Mining, Cham, 2020, pp. 462–474.

[11] Le Merrer, E., Perez, P., Trédan, G., 2019. Adversarial frontier stitching for remote neural network watermarking. Neural Computing and Applications 32, 9233–9244. oi:10.1007/s00521-019-04434-z.

[12] R. Zhu, X. Zhang, M. Shi, and Z. Tang, “Secure neural network watermarking protocol against forging attack,” EURASIP Journal on Image and Video Processing, vol. 2020, no. 1, p. 37, Sep. 2020, doi: 10.1186/s13640-020-00527-1.

[13] Szyller, S., Atli, B.G., Marchal, S., Asokan, N., 2019. Dawn: Dynamic adversarial watermarking of neural networks. arXiv preprint arXiv:1906.00830 .

[14] Y. Li, H. Wang, and M. Barni, “A survey of deep neural network watermarking techniques,” arXiv:2103.09274 [cs], Mar. 2021, Accessed: Feb. 08, 2022.

[15] Z. He, T. Zhang, and R. B. Lee, “VerIDeep: Verifying Integrity of Deep Neural Networks through Sensitive-Sample Fingerprinting,” arXiv:1808.03277 [cs], Aug. 2018,

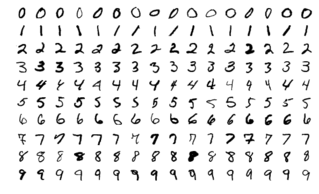
# Annex 1 – MPAI-NNW Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| White-Box | Grant access to the network and make it possible for the watermark to be embedded inside the parameters. |
| Black-Box | Do not grant access to the network but to its output (result of an inference); in this case, the watermark information is inserted inside the output. |
| Imperceptibility | Prediction quality of the model on its original task should not be degraded significantly. |
| Robustness | Watermark should be robust against attacks. |
| Capacity | Allow for inclusion of a certain amounts of information, also called data payload. |
| Attacks | **Removal attacks**: the aim is to remove the watermark by extracting it or making it disappear.  **Geometric attacks:** do not try to remove the mark, but rather to destroy their synchronization.  **Cryptographic attacks**: detect and remove the mark without knowledge of the key, thanks to the knowledge of the embedded mark.  **Protocol attacks**: the aim is to embed another watermark and create an ambiguous situation. |
| Fine-tuning | The network is trained on the same task. |
| Transfer-learning | The network is trained on a similar task, for example on another dataset. |
| Knowledge distillation | A surrogate model is trained by using the output of a well-trained network as label. |
| Pruning | Is the process of removing parameters from a network: techniques can differ, but the aim is to keep performances while decreasing its computational cost? |
| Quantization | The parameters initially stored on 16 or 32bits are approximated by a set of discrete symbols or integer values. |
| Permutation | Let (S, S), D, and L respectively denote the size of the convolution filter, the depth of input to the convolutional layer, and the number of filters in the convolutional layer.  **Neurons’ permutation:** We permute the filters, L dimension.  **Channels’** **permutation:** We permute the channels, D dimension. We apply the same permutation to the filters (L) of the previous layer. |
| Watermark overwriting | The attacker embeds another watermark to create an ambiguous situation where both parties can claim the ownership. |
| Watermark forging | The attacker tries to forge the watermark by creating its own trigger dataset. |

# Annex 2 – MNIST & CIFAR10 description

# MNIST

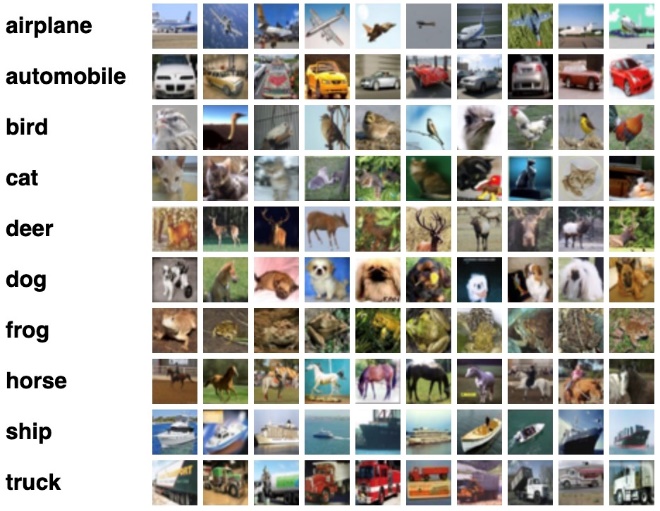
The MNIST database (Modified National Institute of Standards and Technology database) is a diverse collection of handwritten digits. It has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger NIST Special Database 3 (digits written by employees of the United States Census Bureau) and Special Database 1 (digits written by high school students) which contain monochrome images of handwritten digits. The digits have been size-normalized and centered in a fixed-size image. The original black and white (bilevel) images from NIST were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. the images were centered in a 28x28 image by computing the center of mass of the pixels and translating the image to position this point at the center of the 28x28 field. The 10 classes are: 0,1,2,3,4,5,6,7,8,9.



# CIFAR10

The CIFAR-10 dataset (Canadian Institute for Advanced Research, 10 classes) is a subset of the Tiny Images dataset and consists of 60000 32x32 color images. The 10 classes are: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, trucks. The criteria for deciding whether an image belongs to a class were as follows:

* The class name should be high on the list of likely answers to the question “What is in this picture?”
* The image should be photo realistic. Labelers were instructed to reject line drawings.
* The image should contain only one prominent instance of the object to which the class refers. The object may be partially occluded or seen from an unusual viewpoint if its identity is still clear to the labeler.



Source: <https://paperswithcode.com/dataset/>