Digital Watermarking Using Deep Neural Network

Farah Deeba, She Kun, Fayaz Ali Dharejo, Hameer Langah, and Hira Memon

Abstract—**Recently in the vast advancement of Artificial Intelligence, Machine learning and Deep Neural Network (DNN) driven us to the robust applications. Such as Image processing, speech recognition, and natural language processing, DNN Algorithms has succeeded in many drawbacks; especially the trained DNN models have made easy to the researchers to produce state-of-art results. However, sharing these trained models are always a challenging task, i.e. security, and protection. We performed extensive experiments to present some analysis of watermark in DNN. We proposed a DNN model for Digital watermarking which investigate the intellectual property of Deep Neural Network, Embedding watermarks, and owner verification. This model can generate the watermarks to deal with possible attacks (fine-tuning and train to embed). This approach is tested on the standard dataset. Hence this model is robust to above counter-watermark attacks. Our model accurately and instantly verifies the ownership of all the remotely expanded deep learning models without affecting the model accuracy for standard information data.**

Index Terms—**Watermark, embedded, ownership verification, deep neural network.**

I. Introduction

Deep learning and machine learning methods have recently become a hot topic in many computers vision tasks, such as objects and images recognition in still images.

While successful techniques have been manifested with image understanding, video content still presents additional challenges e.g., motion, temporal consistency, the spatial location that usually cannot be bridged with still image recognition solutions. In the structure of DNN neurons are interconnected, and edges are joined together [1], [2]. In DNN we solve the complex set of calculations by using input, convolution, pooling and out layers [3]. Deep Learning wrapper facilitates the design and training of models over many frameworks such as Theano, Tensor Flow, Torch, Chainer, and Keras [4]-[8].

These frameworks reduce the efforts for engineer and researcher in deep learning but still the training the deep model is not easy so it acquires the massive amount of data and time. DNN found in multiple applications, GoogleNet, ResNet, VGGNet, AlexNet and LeNet [9]-[13]. This paper focused on the classification of the watermark image using DNN.

Few models take several days to train deep models such as ResNet and VGGNet [10], [11]. So, therefore, it is difficult to train the model for this problems nowadays GPUs are commonly used to save the training and computation times. Also, some training models are online available on the website which offers comfortable help to try these models directly. Hence sharing models is always an essential part in research progress and future development of deep neural networking (DNN). Other digital platforms offer trained models similar to google app store or even artificial intelligence models available, but security is required to protect these models.

In our work, we employed DNN models for digital watermarking Technology. Watermarking scheme for deep neural networks is proposed which is the black box in terms of verification. Watermarking is applied to recognize the possession of the copyright of digital contents such as audios, images, and videos.

We perfumed the experiment watermark embedding through DNN, in the watermarking process, training the neural network to be as accurate as possible concerning these randomly chosen layers and neural network formalize the idea of backdooring a neural network with specific properties. People prefer neural networks now because parameters are not as important as such as the performance. Hence, it is necessary to improve the host trained network performance. There are some main properties which we want to achieve with our watermarking scheme the first one is it supposed to be functionality preserving there's no point of watermarking. Functionality preserving so does a watermark harm the task that you want to achieve it turns out so we ran experiments training without the watermark and with the watermark and it turns out that the accuracy of the neural network is almost the same like it sometimes even increases a bit.

II. DNN Watermarking

Digital watermarking is the process to protect the hidden information into the digital media to preserve the ownership of those media data. Many approaches have been introduced to obtain the watermark to be active as well as robust to removal of different attacks. Based on extraction methods, digital image watermarking approaches can be split into two main categories named Blind- watermarking and Non-Blind watermarking [14]. Traditional algorithms for digital watermarking can be classified into two methods transform domain also well known as frequency domain [15] and spatial domain [16]. Blind watermarking only need

Manuscript received May 13, 2019; revised January 7, 2020. This work is supported by the Sichuan Science and Technology Support Program under grant no. [2018GZDZX0012].

Farah Deeba and She Kun are with the School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China (e-mail: farahdeebauestc@hotmail.com, Kun@ uestc.edu.cn).

Fayaz Ali Dharejo is with the Computer Network Information Center, Chinese Academy of Sciences University of Chinese Academy of Sciences, Beijing, Haidian 100190, China (e-mail: fayazdharejo@cnic.cn).

Hmaeer Lnagah and Hira memon are with the Computer System Engineering, Quiad e Awam University of Engineering Science and Technology of Nawabshah, 67450, Pakistan (e-mail: hameer.langah@outlook.com, hiramemon09@gmail.com).

watermarked images for the extraction process; however, in non-blind watermark approach watermarked images, original images, and extra information is also required to extract information. Spatial domain digital watermarking algorithms have been proposed [16]-[18].

Uchida *et al*. [19] previously introduced the first approach for embedding watermarks into a deep neural network through which he embedded the data into the weights of DNN. Hence, it believes that the stolen prototypes can be nearby accessible to obtain the group of parameters, which is not possible since most deep learning models are used as online assistance and it will be troublesome to immediately procure access to model parameters, particularly for the stolen prototypes. As DNN models are broadly deployed and become more critical, they are frequently targeted by enemies. Enemies can steal the model and build a copy of AI service.

There are two types of DNN watermarks that are used mainly, white box and the black box. In the white box, watermark techniques have advantages over black box watermarking techniques in terms of operator information. However, the ability to transmit a lot of information has the cost of limited application scenarios, since it requires that internal models are publicly available for the extraction of WM. The bit error rate (BER) between the extracted WM and the background truth must be zero to demonstrate the authorship of the model consulted.

A. White-Box Watermarking

Uchida *et al*. [20] first introduced the DNN watermarking scheme, which contains a multi-bit WM in the weighting of selected layers of the target DNN by adding an additive binary cross-entropy loss as a regularization member of WM. When training a DNN model, a watermark is coded in the order of weights, losing the personalized regularization, thereby penalizing the difference between the desired WM and the transformation of the weights. With the selected transformation weights, the WM is extracted from the consulted model. It is robust to fine-tuning and compression attacks on models also maintain the accuracy of the model.

Rouhani *et al.* [4] presented a general watermarking framework that works in both the white box and the black box environment. The embedded bit watermark is embedded in the activation cards of the selected layer (s) of the target DNN by integrating two additional regularization loss terms, binary cross-entropy loss, and GMM agent loss. To activate cards, a deep Gaussian Mixed Model (GMM) is used. During DNN training, WM-specific regularization loss conditions are integrated to match the activations and encode the WM information. Later in the extraction step, the WM key frames are transferred to the model to trigger a marked activation. The BER is calculated and the result of these activation watermarks is restored. This has an advantage over the Uchida [6] because it gives both data and model-dependent. At the same time, it is robust against attacks with fine-tuning, overwriting and parameter setting. Since it is both data and model-dependent but more computation.

B. Black-Box Watermarking

There are several methods for the black box watermark [21]-[24] that target the zero bit. The presence of the WM is determined by interrogating the model with the WM key frames and determining the appropriate accuracy by thresholding. Black boxing WMs are more practical in the real world as they are more relaxed, instead of API access rather than model internals, but with limited capacity.

Merrer *et al*. [22] suggest creating controversial patterns if the opponent succeeded in the attack, the corresponding samples are called "true opponents”. In this scheme, the set of opposite samples is used as the WM key set to change the decision boundary of the target neural network. While attacks fail, images l watermark key is a complete combination of true and false opponents. The model is queried in the detection phase of watermark and the null hypothesis is tested, this is done via the Binary distribution. If the watermark key is less than the threshold, the watermark should be present in the model. It is practical from the point of view of detection and embedding, but the accuracy can be lowered after embedding watermarks, achieves a very high false-positive rate and is not very robust against attacks. Images incorrectly classified by the model are suggested by Yossi *et al*. [1]. These misclassified images known as backdoor images and random labels as a watermark key that is set to embed the watermark with a zero bit that is DNN compliant. The comparable basis of the key label is randomly selected from all classes except the real label and the predictable real sample. First, the threshold is set by the binomial distribution and then compared to the watermark trigger set to detect the watermark. This scheme provides better accuracy and a high detection rate but is sensitive to water removal attacks. Zhang *et al*. [24] introduced the three different key generation schemes content-based, noise-based and unrelated-based images respectively. Pre-train model is accurately adjusted with watermark keys and in the detection phase, the user sends the watermark key to the DNN model. This is the external service provider that ultimately submits the threshold to classify the Boolean decision. He introduced three different methods for generating a watermark key, but not well for giving a continuous performance on the standard datasets. Rouhani *et al.* [23] in the goal, DNN investigated the rarely inhabited space to regard the random images and labels as a watermark set. The watermark key used for identifying the key pairs and for Watermark embedding fine-tuning the target DNN model. The final WM key set is the intersection of the keys that are correctly predicted by the highlighted model and incorrectly predicted by the unmarked model. In the multinomial distribution of the detection phase for the output prediction and does the examination of the imaginary hypothesis with the definitive Watermark key set. It gives a high detection rate with a low percentage of false alarms but the generation of keys generates higher overhead costs. Look at fake enemies that preserve the model's accuracy for specific data.

As DNN can learn the patterns automatically once the parameters are fixed; hence we implemented DNN in watermarking one is to protect the Intellectual properties and second is to verify the ownership with embedded watermarking. The goal of the framework is to preserve the intellectual characteristics of the deep neural networks through verifying possession of remote DNN services with embedded watermarks. In our model first, we defined the labels for different watermarks and also trained these labels to the deep neural network. In the model without the watermark by using less computation a lot less estimate than training from scratch. It can never defend against an enemy who makes a model related to cryptography where we assume that there is a gap in computational power between the attacker and the defender.

The Diagram of proposed model is given in Fig. 1.

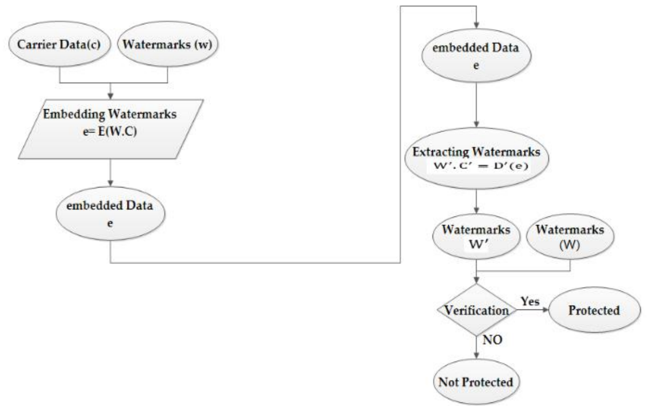


Fig. 1.Watermarking process.

From Fig. 1 we can recognize the complete typical watermarking process. An embedding algorithm "E" embeds the pre-trained watermarks W into the carrier data C that should be protected from attacks. When embedding is done the data stored in (e=E(W,C)) where decryption is appliedto extract the watermark "W" from e.

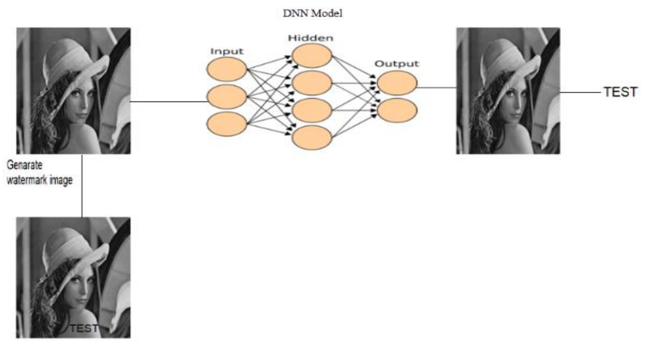


Fig. 2. Workflow of DNN watermarking.

The above Fig. 2 shows the DNN watermark Framework.

When generating Watermarks these watermark shown as the fingerprint in ownership as well. It first creates the watermark; these watermarks reported as the fingerprint verification. Then these watermarks embed to DNNs through training. The DNN automatically learns the patterns of watermarks and retains them. And finally for verification after embedding our models is strong enough to verify the ownership. Following newly generated models able to verify ownership so that the owner can prove by simple sending watermark as input and checking through output by deploying DNN model. Rest our paper contains Watermark Generations, types of attacks (fine-tuning and train to embed) and owner verification and finally we demonstrated the Experiment Setup.

III. Embedding and Watermarks Generation

In embedding retraining to input distribution which is entirely unrelated to the task on the training data using the labels in the neural network as accurate on the strain data as possible without over-fitting. At the same time these random inputs which chose independently of the input distribution. Accept arbitrary labels first fix and then during different ways how to determine the input distribution they use different security they have different security properties they want to show for the neural network and therefore have a separate analysis We applied the watermark algorithm which already used by Zhang *et al.* [14].

A. Algorithm: Watermark Embedding

```
**Input**:
$TrainingsetD_{train} = \{x_i,Y_i\}_{i=1}^{S}$
$DNNkeyK = \{Y_s,Y_d\}(s \neq d)$
**Output**:
$DNNmodel:Z_\theta$
$Watermarkpair:D_{wm}$
**StartFuctionEmbeding** ()
$D_{wm} \leftarrow \emptyset$
$D_{tmp} \leftarrow sample\{D_{train,}Y_{s,}percentage\}$
$foreachd \in D_{tmp}do$
$x_{wm} = ADD\_Watermark\{d(x),watermarks\}$
$y_{wm} = y_d$
$D_{wm} = D_{wm} \cup \{x_{wm,}y_{wm}\}$
$endfor$
$endfuction$
$Z_\theta = Train\{D_{wm,}D_{train}\}$
$returnZ_\theta,D_{wm}$
```

The algorithms of embedding show that DNN Watermark embedding performs the original training data (D_{train}) and transformation keys $\{Y_s, Y_d\}$ (s, d) which are inputs and finally outputs protected the DNN model (Z_8, D_{wm}). With the help of transform key, we label the watermarks which are defined by the owner. Here Y_d is owner defined labels for the watermark and Y_s is original label data.

We took the same amount of training with and without the embedding watermark so that it probably increases a bit is like apparently just outliers. We also validated and experimentally verify that it is not only able to remove the watermark on one specific attack, but there are a few more attacks. so we did a fine-tuned either only the last layer or all layers , as fine-tuning is a standard thing to do if you have more training data to adapt your neural network. The following two circumstances noticed that the host network of the copyright owner is supposed to embed a watermark to the host network in training or fine-tuning.

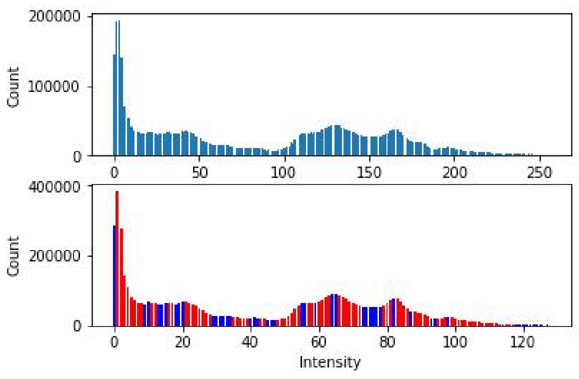


Fig. 3. With and without watermarking.

We Recognized watermarks, the histogram of the embedded watermark and without watermarks through σ (ΣiXjiwi) as shown in Fig. 3.

In the first image we see only the image without watermarks (only the blue histogram), and in the second image the watermark is used (with the red and blue histogram).

B. *Train-to-Embed*

Train to embed means the training from scratch and embedding the watermark where labels data training available of the host network.

C. *Fine-Tuning to Embed*

In this attack, the watermarked model is re-trained with the traditional cross-entropy loss of the original training data set. A robust watermark must be able to change weights during model fine-tuning. Fine-tune the whole network which means to annihilate the watermark this was kind of unexpected for us because at the same time embed the watermark during the training.

When operating the entire dataset, we split the test dataset into two parts. In the first part, fine-tuning is used for previously trained DNN and in the second part for the evaluation of additional models. Later, testing and watermarking several new models to estimate the robustness of our watermark frame for fine-tuned changes.

D. *Ownership Verification*

To verify the ownership, we must send the regular queries with generated watermarks images that is D_{wm}, if the Query matches which means $x_{\text{wm}}=Y_{\text{wm}}$ we can validate that our model is protected. It is due to the DNN model [20]. Because without embedding watermarks cannot understand watermarks so that it can classify images but will not classify images with embedded watermarks.

E. *Meaningful Content (Watermark-Content)*

In training images, we add the text to give useful content on it; these images taken as input images. For example, if we insert the "TEST" string in our model to the Lena image. In this example, the query watermark (TEST) in the Lena image as a predefined prediction (Lena) which consist of fingerprints for model verification. Fig.4 shows the Lena image with and without content.

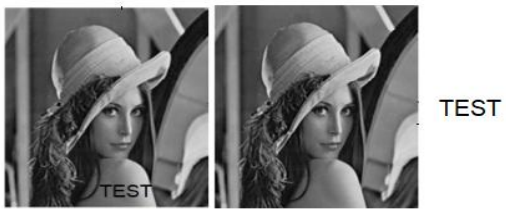


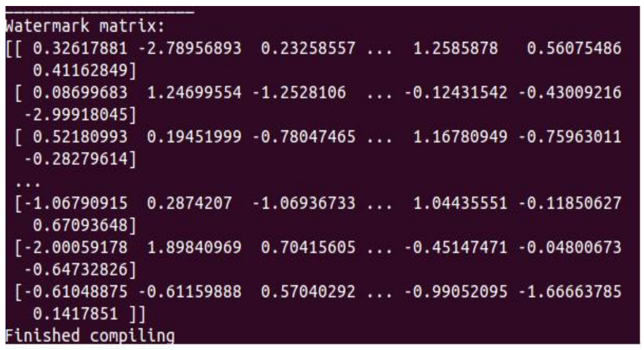
Fig. 4. LENA images with meaningful content.

F. *Overview of Proposed Matrix Image*

During training on the different images, Random Matrix Image auto-generated. The matrix can be generated using the random function with the randomized number from the specified range. This matrix gives shows the output of our proposed DNN model.

$$\begin{bmatrix} 0.3261 - 2.7895 & 0.2325 & \cdots & 0.41162 \\ \vdots & & \ddots & \vdots \\ -0.61048 - 0.61159 & 0.5704 & \cdots & 0.14175 \end{bmatrix}$$

The auto-generated Matrix of the Lena image is shown in Fig. 5.



```
-----------------------
Watermark matrix:
[[ 0.32617881 -2.78956893  0.23258557 ...  1.2585878   0.56075486
   0.41162849]
 [ 0.08699683  1.24699554 -1.2528106  ... -0.12431542 -0.43009216
  -2.99918045]
 [ 0.52180993  0.19451999 -0.78047465 ...  1.16780949 -0.75963011
  -0.28279614]
 ...
 [-1.06790915  0.2874207  -1.06936733 ...  1.04435551 -0.11850627
   0.67093648]
 [-2.00059178  1.89840969  0.70415605 ... -0.45147471 -0.04800673
  -0.64732826]
 [-0.61048875 -0.61159888  0.57040292 ... -0.99052095 -1.66663785
   0.1417851 ]]
Finished compiling
```

Fig. 5. Proposed matrix of LENA image.

IV. DATASET

We use the following benchmark image dataset for the evaluation. The architecture and training parameters of DNN models for each dataset digital image watermarking technique is tested on six well known grey images Lena-image with the dimension of 512 × 512 and standard image-set5 having different images of different sizes have been analyzed in this analysis shown in Fig. 6.

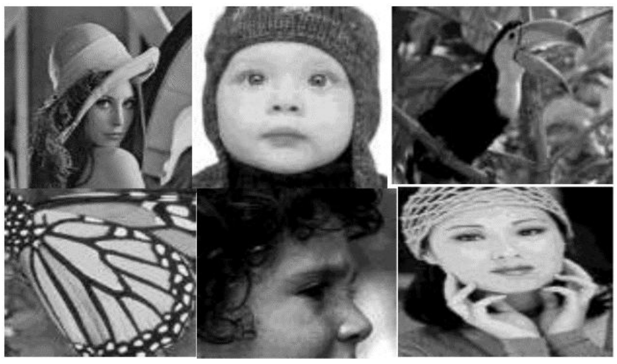


Fig. 6. dataset.

V. EXPERIMENT

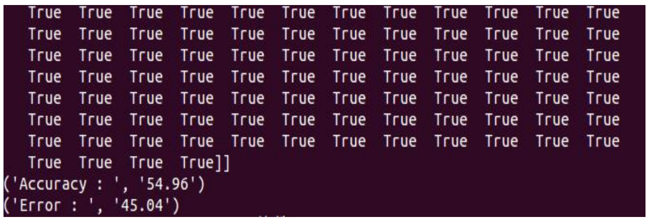
We have done all our experiments by using Python 2.7 with Spyder Integrated Development Environment(IDE) and Keras [8] and TensorFlow [5] frameworks using device GforceGTX 770, CUDNN 5110. We assess the performance of our watermarking framework with the standard dataset with learning ratee of 0.5. We trained our model on multiple epochs. Each time it gives almost the same error rate and the same accuracy. So we notice that our embedding DNN model perform well against the types of attacks which means it can embed without the impairing the performance of the host network.

TABLE I. THE ACCURACY OF DNN MODEL

watermark-Content	**Accuracy**
Watermarks (trained)	54.7%
Watermarks (test)	51.89%

Here Table I and Fig. 7 show the accuracy of our model during Training. Table II shows the Architecture of our DNN model.

We have inserted a watermark in the following we will mention the location of the host layer by simply drawing the convolution 2, 3 and 4 groups.



```
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True]]
('Accuracy : ', '54.96')
('Error : ', '45.04')
```

Fig. 7. Accuracy of our proposed DNN model.

TABLE II. THE ARCHITECTURE OF DNN MODELS

Layer Type	Lena-Dataset
Conv.ReLU	32 flters (3 × 3)
Conv.ReLU	32 flters (3 × 3)
Max Pooling	2 × 2
Conv.ReLU	64 flters (3 × 3)
Conv.ReLU	64 flters (3 × 3)
Max Pooling	2 × 2
Dense.ReLU	256
Dense.ReLU	256
Softmax	10

Fig. 8 shows the histogram of the watermark marks detected. There are two points, the red and the blue, the red screws the histogram from embedded while the blue is not marked embedded. From the Fig. 8, we can see that much close-up case that helps to detect the presence of watermark known as a zero bit watermark.

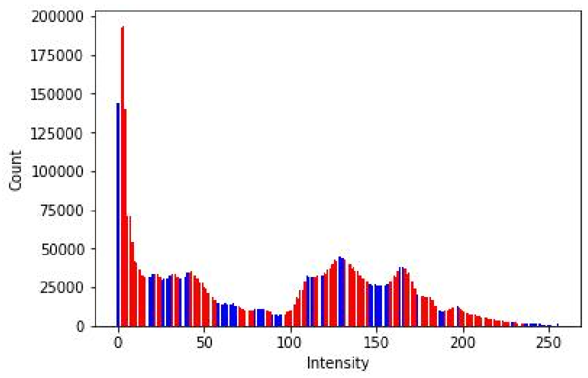


Fig. 8. Histogram of the detected watermark algorithm applied to an image that has undergone DNN substitution.

VI. CONCLUSION AND FUTURE WORK

We generalized the digital watermark using the deep neural network, and remotely verify the ownership of DNN models based on the embedded watermarks. Our paper explained the idea of the digital watermark in Deep Neural Network. We learn different frameworks of watermarking, embed them into DNN model and finally, we perform owner verification. Achievement of this DNN model successfully verified owner verification based on embedded watermarks. These all performance evaluated on the Standard Dataset, and we show that our framework can meet the general watermarking standard; hence our model is robust to above possible attack.

Our model gives Inconsistent performance across different benchmarks so in future still the accuracy can be improved and also can be tested on different Stoddard datasets such as CIFAR-10, CIFAR-100.

CONFLICT OF INTEREST

"The authors declare no conflict of interest".

AUTHOR CONTRIBUTIONS

Farah Deeb and FA Dharejo has produced the results and write the manuscript. She Kun has guided in the experiments and supports in funding. While Hameer and Hira have analysed data and all results.

REFERENCES

[1] N. Aloysius and M. Geetha, “A review on deep convolutional neural networks,” in *Proc. 2017 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, 2017, pp. 0588-0592.

[2] O. Keiron and N. Ryan, *An Introduction to Convolutional Neural Networks*, 015CoRR, abs/1511.08458.

[3] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, “Multimodal deep learning,” in *Proc. the 28th International Conference on Machine Learning*, 2011, pp. 689–696.

[4] Theano Development Team, “A python framework for fast computation of mathematical expressions,” *2016.arXiv e-prints*, abs/1605.02688.

[5] M. Abadi, A. Agarwal, P. Barham *et al.*, “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” *2016arXiv preprint* arXiv:1603.04467

[6] R. Collobert, K. Kavukcuoglu, and C. Farabet, “Matlab-like environment for machine learning,” in *Proc. Big Learn, Advances in Neural Information Processing Systems Workshop*, 2011.

[7] S. Tokui, K. Oono, S. Hido, and J. C. Chainer, “Next-generation open source framework for deep learning,” in *Proc. NIPS Workshop on Machine Learning Systems*, 2015.

[8] F. C. Keras. (2015). GitHub repository. [Online]. Available: https://github.com/keras-team/keras

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proc. CVPR*, 2015, pp. 1-9.

[10] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016.

[11] K. Simonyan and A. Zisserman. “Very deep convolutional networks for large-scale image recognition,” in *Proc. 2015 CoRR*, abs/1409.1556.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Image classification with deep convolutional neural networks,” in *Proc. NIPS*, 2012.

[13] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[14] J. L. Zhang, Z. S. Gu *et al.* (2018). Protecting intellectual property of deep neural networks with watermarking. [Online]. Available: https://gzs715.github.io/pubs/WATERMARK_ASIACCS18.pdf

[15] E. Elbasi, “Robust multimedia watermarking. Hidden Markov model approach for video sequences,” *Turk J ElecEng& Comp Sci*, vol. 18, pp. 159-170, 2010.

[16] C. Munesh and S. Pandey, “A DWT domain visible watermarking techniques for digital images,” in *Proc. International Conference on Electronics and Information Engineering*, 2010, pp. V2-421-V2-427.

[17] L. Leida *et al.*, “Localized image watermarking in spatial domain resistant to geometric attacks,” *International Journal of Electronics and Communications*, vol. 63, no. 2, pp. 123-131, Feb. 2009.

[18] Y.-K. Lee, G. Bell, S.-Y. Huang, R.-Z. Wang, and S. J. Shyu, “An advanced least-significant-bit embedding scheme for stegano graphic encoding,” in *Proc. the 3rd Pacific-Rim Symposium on Image and Video Technology*, pp. 349-360, 2009.

[19] J. Tian, “Reversible data embedding using a difference expansion,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, August 2003.

[20] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, “Embedding watermarks into deep neural networks,” in *Proc. the 2017 ACM on International Conference on Multimedia Retrieval*, pp. 269-277.

[21] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, “Turning your weakness into a strength: Watermarking deep neural networks by backdooring,” in *Proc. Usenix Security Symposium*, 2018

[22] E. L. Merrer, P. Perez, and G. Tredan, "Adversarial frontier stitching for ´remote neural network watermarking," *arXiv preprint*, arXiv:1711.01894, 2017.
[23] B. D. Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: A generic watermarking framework for IP protection of deep learning models," *arXiv preprint*, arXiv:1804.00750, 2018.
[24] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proc. the 2018 on Asia Conference on Computer and Communications Security*, pp. 159–172.

Farah Deeba received B.E. and M.E. degrees in the Department of Software Engineering from Mehran University of Engineering and Technology Jamshoro Pakistan in 2010 and 2014, respectively. She has 3 years' experience in teaching at graduate and postgraduate level in Hamdard University Karachi Pakistan. Currently, she is a Ph.D. scholar at the School of Information and Software Engineering, University of Electronic Science and Technology, China.

She Kun received B.S degree in the Department of applied mathematics from 1985 to 1989. He got the M.S. degree from the School of electronic and communication system, Southwest Institute of Communication, and PhD degree in computer application from the University of Electronic Science and Technology, China. Currently, he is a professor at the School of Information and software engineering Technology, University of electronic Science and Technology China. His main research interests cloud computing and cloud security.

Fayaz Ali Dharejo received the B.E. degree in electronic engineering from Quaid-e-Awam University of Engineering Science and Technology (QUEST) Pakistan and M.S. degrees from the School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC) in 2016 and 2018, respectively. Currently, he is a PhD student at the Institute of Computer Network Information Center Chinese Academy of Sciences, University of Chinese Academy of Sciences, China.

Hameer Langah received undergraduate degree in computer system engineering from Quaid-e-Awam University of Engineering Science and Technology (QUEST) Pakistan in 2017.

He is working with Private Electronic company in Karachi as a lab Engineer.

Currently he is working on image processing, machine learning and deep learning research fields.

Hira Memon received the bachelor of engineering and master of engineering degrees in computer system engineering from Quaid-e-Awam University of Engineering Science and Technology (QUEST) Pakistan in 2010 and 2016 respectively.

Currently, she is working as a junior lecturer at the university of sufism and modern sciences, pakistan. her research areas are big data, image processing and machine learning.