International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Image Security Using Artificial Neural Networks

Karuna Manjusha.Y[1], Assistant Professor, Department of Computer Science Engineering, Andhra Loyola Institute of Engineering and Technology, Vijayawada, India.
Suchit.V[2], Department of Computer Science Engineering, Andhra Loyola Institute of Engineering and Technology, Vijayawada, India.(19HP1A05C0).
Chandrakanth.K[3], Department of Computer Science Engineering, Andhra Loyola Institute of Engineering and Technology, Vijayawada, India.(19HP1A05A0).
Jothik Chandra.A[4], Department of Computer Science Engineering, Andhra Loyola Institute of Engineering and Technology, Vijayawada, India.(19HP1A05A5).

Abstract
The result is that pictures with significant business or personal worth are increasingly vulnerable to cyber attacks, leaks, and illegal distribution. While this was a problem in the past, artificial neural networks have made short work of data reduction in recent years. There are only a small number of tried-and-true methods for protecting images, and they fall into two camps: encryption and data concealing. In order to secure images, researchers have concentrated on utilizing chaos theory-based modification or encrypting technologies on the Fourier transforms of the images. Steganography and picture watermarking are two methods for concealing data while maintaining an image's credibility. The primary objective of this work is to reduce pictures for simpler access, storing, and transfer, and then encode them in a way that is indecipherable to cloud service providers and other unauthorized parties.

Keywords: Cyber Attacks, Illegal Distribution, Encryption, Secure, Storing, Encode.

Introduction

In recent years, more electronic devices like mobile phones have started to provide additional functions like saving and exchanging multimedia data. Digital images are widely used in the internet, how to protect image content has become an issue to be urgently solved. Over the past two centuries, "information security" has become the most frequently used term by humans, machines, and their accessories. Incorporating security measures against potential threats is now standard practice when developing innovative technologies. Most chaotic encryption methods fall into one of three broad buckets: To begin, they can use chaos as a means of performing complex permutations of coordinates via iterative processes. Those who put chaos theory to use in encryption methods first used chaotic coordinate transformations like the baker map to jumble the topology of images as much as possible. By repeatedly 'kneeling' and 'folding' two-dimensional data, the Baker map is able to achieve its desired results. For digital pictures, this map is discretized, while in its general form, it can be used to integrate a hidden key into an encryption procedure.

Parallel adjustable networks made up of non-linear computational components named Neurons make up Artificial Neural Networks (ANN). The input signal and the connecting weight are combined and weighted in ANN's operation. The signal obtained after adding the threshold is then proposed as an input to a sigmoid nonlinear function. Due to their hypersensitivity to starting circumstances and input parameters, which results in pseudorandom and unexpected behaviour, chaos maps have been utilized for a number of years in cryptography applications. In this work, we employ a novel picture encryption method that makes use of ANN and an unstable

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
IJARST
ISSN: 2457-0362

system for maximum security. Climate and the weather are just two examples of the many natural systems that exhibit chaotic behaviour. The purpose of this work is to explore the potential of ANNs in the context of chaotic Cryptography. An unpredictable pattern is used to determine the neural network's weights. The resulting chaotic sequence is then sent to ANN, where its weights are updated to affect the subsequent generation of the encryption key*. To increase security, the Convolutional Neural Networks(CNN) is used. The weights for this neural networks is given by chaotic generator. The algorithm used for image encryption is Advanced Encryption Standard (AES). The MATLAB programme has been used to test and analyse the algorithm's performance.

Problem Statement

Data protection and security have become increasingly important with the rise of the digital economy and online financial activities. Encryption, encoding, tokenization, and key management are all part of data security practices that ensure the safety of data across all platforms and apps. However, the safety of pictures during their storing in the cloud or during their transmission over the internet has received little attention. Prior art lacks any viable methods for compressing images efficiently. Existing systems' picture encryption and decoding procedures were cumbersome.

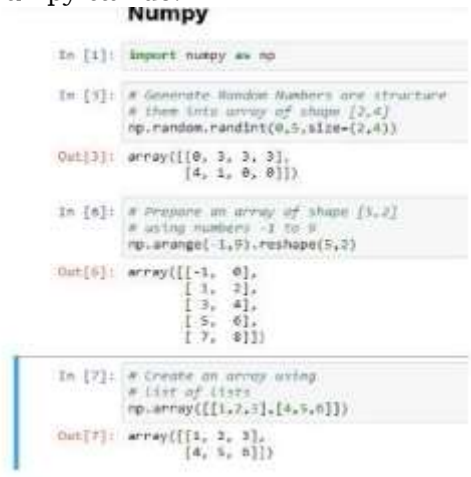
Proposed System

This article uses an artificial neural network to encode and decode pictures for privacy and security purposes, and it also uses another artificial neural network, the AUTOENCODER and AUTODECODER model, to decrease the size of the images so that they can be transferred more quickly over the network. After being trained on the dataset pictures, the model in AUTOENCODER and DECODER can recreate a new image from the original image with poor quality, resulting in a smaller file size and quicker network transmission times. The pictures are encrypted using a Neural Network that has been trained using the CIFAR dataset. The trained model can then be applied to an input image, at which point the pixels in the input image will be scrambled so that no one can see what the image actually is. When compared to prior studies, our suggested system's reduction results are superior. Image encryption in a way that cloud service providers and other unauthorised parties cannot decipher paves the way for streamlined picture recovery, storing, and transfer.

Implementation

A. Numpy

Python's library of built-in data classes and structures is robust. However, it wasn't created with Machine Learning in mind. Step in, numpy (pronounced as num-pee). Numpy is a library for working with data; more specifically, it is a library for working with big multi-dimensional vectors and a vast set of mathematical functions. Here is a small sample of what numpy can do.



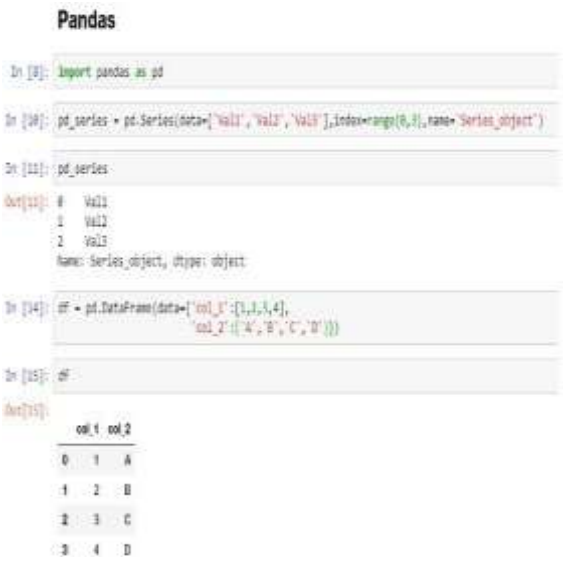
Though it is well-known for its capacity to work with multivariate data, Numpy is much more than just a tool for working with data. It's also well-known for how quickly it can execute and how well it can vectorize. Because it offers MATLAB-like capabilities, it will take some time to become proficient in its use. Many other popular tools rely on it as well, including pandas, matplotlib, and many more. The product's literature is useful in and of itself.

B. Pandas

Imagine pandas, think tabular data. To manipulate data in a variety of formats, pandas is a Python tool that offers a

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
IJARST
ISSN: 2457-0362

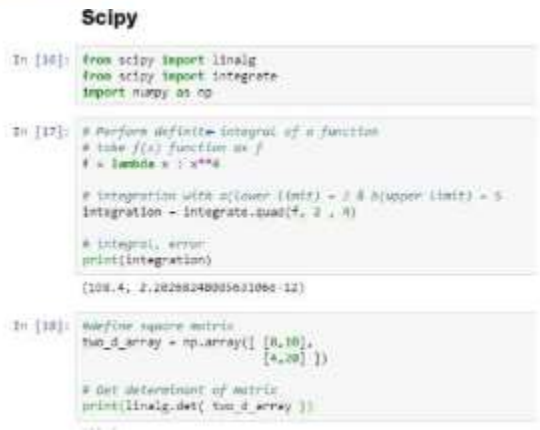
number of useful data structures, including data frames and series. Pandas, which was built on top of numpy, is both faster and more user-friendly.



Python's Pandas library allows users to easily import and export data from popular formats and formats such as CSV, Excel, SQL, Hadoop, and many more. Add, modify, and remove fields; merge or divide data frames/series; manage datetime objects; substitute null/missing values; process time series data; convert to/from numpy objects; etc. are all possible with this package. Pandas are a necessity if you are working on a practical application of Machine Learning. Among the many useful modules that make up the SciPy or Scientific Python Stack, numpy and pandas are two of the most widely used.

C. Scipy

Among the most significant python modules ever, this one is pronounced as Sigh-Pie. Python's Scipy module provides functionality for science processing. Like numpy, it is a component of the Scipy Stack and was developed on top of that library.



This is another utility that operates in the background and does a lot of the hard work. Linear algebra, integration, image processing, transformations, grouping, sparse matrix manipulation, and many more are all covered by its many components and methods.

D. Matplotlib

Matplotlib, another part of the SciPy stack, is a collection of tools for creating graphic representations of data. Numpy objects can be used with no problems (and its high-level derivatives like pandas). Matplotlib is a MATLAB-like charting tool for creating professional-looking plots and graphs for use in papers, journals, websites, and more.

Matplolib is a low-level tool with extensive customization options that can be used to generate nearly any kind of graphical representation. Because of its primitive character, it takes some practise and a lot of code to master. The extensive documentation and modular structure of its architecture have encouraged the development of numerous advanced rendering tools. You'll read about some of

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
IJARST
ISSN: 2457-0362

them in the following chapters.

E. Scikit-Learn

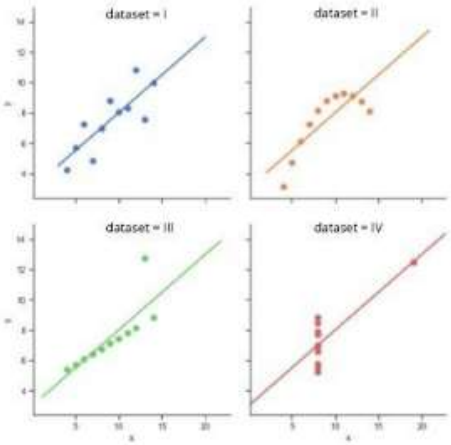
Built as a plug-in for the SciPy framework, scikit-learn is now the go-to tool for a wide variety of machine learning projects. Developed as part of Google Summer of Code project,it has

Scikit-Learn {Source: Sklearn Examples}

```
In [19]: from sklearn import svm
         from sklearn.datasets import make_blobs

In [24]: # we create 40 separable points
         X, y = make_blobs(n_samples=40, centers=2, random_state=6)
         # fit the model, don't regularize for illustration purposes
         clf = svm.SVC(kernel='linear', C=1000)
         _ = clf.fit(X, y)

In [25]: plt.scatter(X[:, 0], X[:, 1], c=y, s=30, cmap=plt.cm.Paired)

         # plot the decision function
         ax = plt.gca()
         xlim = ax.get_xlim()
         ylim = ax.get_ylim()

         # create grid to evaluate model
         xx = np.linspace(xlim[0], xlim[1], 30)
         yy = np.linspace(ylim[0], ylim[1], 30)
         YY, XX = np.meshgrid(yy, xx)
         xy = np.vstack([XX.ravel(), YY.ravel()]).T
         Z = clf.decision_function(xy).reshape(XX.shape)

         # plot decision boundary and margins
         ax.contour(XX, YY, Z, colors='k', levels=[-1, 0, 1], alpha=0.5,
                    linestyles=['--', '-', '--'])
         # plot support vectors
         ax.scatter(clf.support_vectors_[:, 0], clf.support_vectors_[:, 1], s=100,
                    linewidth=1, facecolors='none', edgecolors='k')
         plt.show()
```

now become a widely contributed open source project with over 1000 contributors.

To learn from data, change the data, and then make predictions, Scikit-learn offers a straightforward yet effective fit-transform-and-predict model. It allows users to create categorization, regression, grouping, and ensemble models via this user interface. As an added bonus, it offers a wide range of tools for things like preparation, measurements, model assessment methods,etc.



F. Seaborn

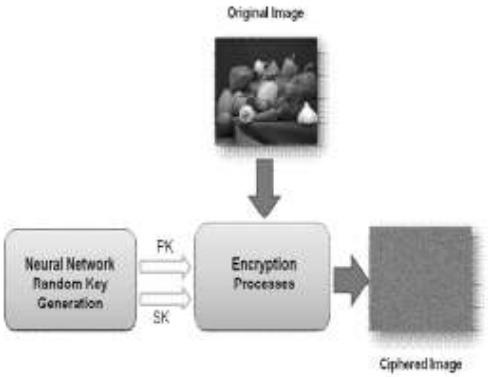
Seaborn is an advanced rendering tool that extends the functionality of matplotlib. It comes with a variety of high-end default designs (something that would be very difficult to achieve with matplotlib).

Seaborn {Source: Seaborn Examples}

```
In [4]: import seaborn as sns
        sns.set(style="ticks")

        # Load the example dataset for Anscombe's quartet
        df = sns.load_dataset("anscombe")

        # Show the results of a linear regression within each dataset
        sns.lmplot(x="x", y="y", col="dataset", hue="dataset", data=df,
                   col_wrap=2, ci=None, palette="muted", height=4,
                   scatter_kws={"s": 50, "alpha": 1})
```

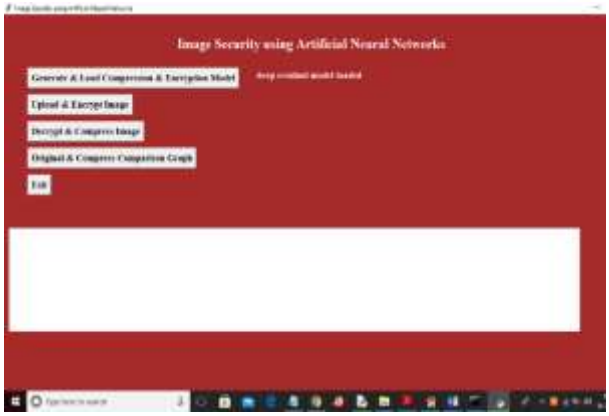
System Architecture

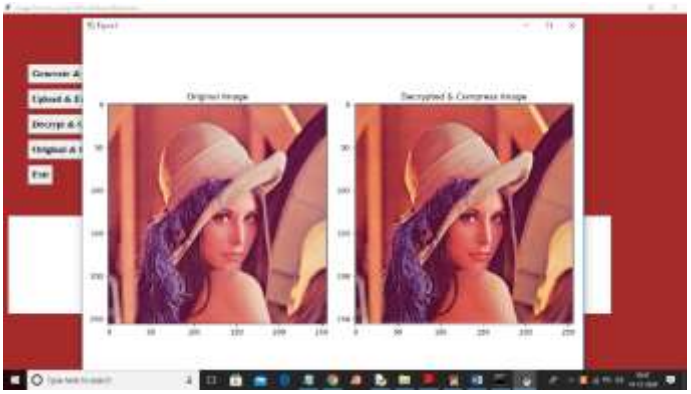
There is no development model or field of application that cannot benefit from good software design, which is why it is considered the technological heart of the software engineering process. For any manufactured good or system, design is the initial stage of creation. The designer's job is to sketch out a plan for something that will eventually be constructed. As the first of three technological activities—design, code, and test—needed to construct and validate software, system design follows the specification and analysis of system requirements.

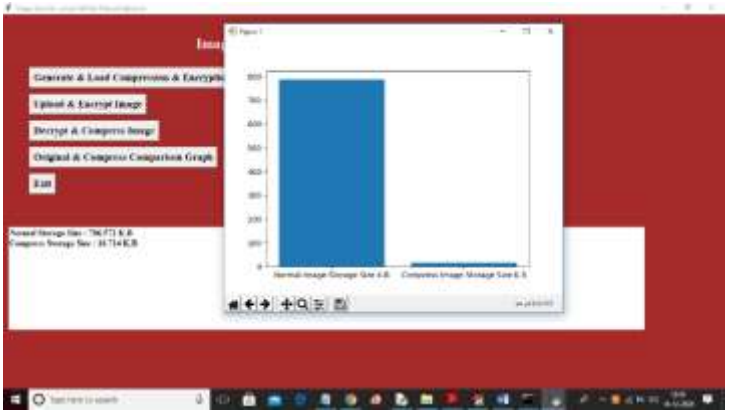


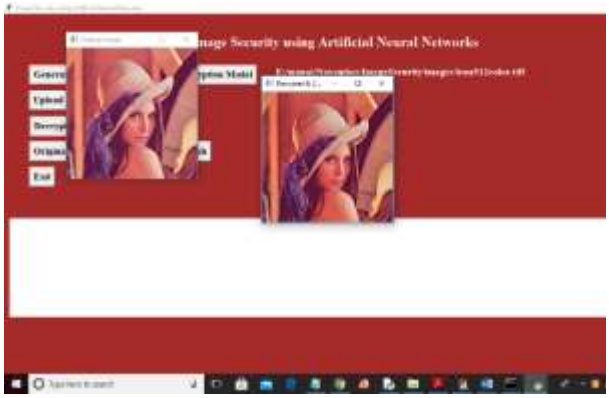
Results

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
IJARST
ISSN: 2457-0362









Conclusion

Work In this article, we first present some of the current study work in neural network based content protection, covering topics such as the characteristics of neural networks that make them well-suited for content protection and some content protection methods based on neural networks. The neural network's ability to learn and its one-way characteristic are then put to use in the proposal of an identification system for audio visual material that can identify malevolent manipulation. The efficiency assessment demonstrates the usefulness of the plan. In addition, some unanswered questions in this area of study are introduced. The results are analysed and made inferences.

References

[1] Abdi, H., Valentin, D., Edelman, B.E. (1999). Neural Networks. Thousand Oaks: Sage.

[2] Anderson, James A. (1995). An Introduction to Neural Networks. ISBN 0-262-01144-1.

[3] Arbib, Michael A. (Ed.) (1995). The Handbook of Brain Theory and Neural Networks.

[4] Bezdek J. C. On the Relationship between Neural Networks, Pattern Recognition and Intelligence. The International Journal of Approximate Reasoning, 1992, 6(2): 85-107.

[5] Grossberg S. Neural Networks and Neural Intelligence. Cambridge, Mass: MIT Press, 1988.

[6] Hajek P, et al. Uncertain Information Processing Expert Systems. Boca Raton, Florida: CRC Press, 1992.

[7] Dai Q, Chen SC, Zhang BZ, Improved CBP neural network model with applications in time series prediction. NEURAL PROCESSING LETTERS 18 (3): 197-211 DEC 2003. [8] Y. D. Jou, "Design of real FIR filters with arbitrary magnitude and phase specifications using a neural-based approach," IEEE Trans. Circuit and Systems-II, vol. 53, no. 11, October 2006.

[9] S. Kulkarni, B. Verma, and M. Blumenstein, 1997, Image Compression using a Direct Solution Method based Neural Network, Proceedings of the Tenth Australian Joint Conference on Artificial Intelligence, Perth, Australia, 114-119.

[10] Gabriele Manganaro, P. Arena, L. Fortuna. Cellular Neural Networks: Chaos, Complexity and VLSI Processing, Springer Series in Advanced Microelectronics, 1999.