

ADAPTIVE AUTHENTICATION OF BANKNOTES USING DEEP NEURAL NETWORKS

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

THIMMARAJU DURGA DHANASRI JYOTHIRMAI

(Reg. No. 37110779)

THALLAPAKA SRAVANA LAKSHMI

(Reg. No. 37110771)



***DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING***

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC JEPPIAAR

NAGAR, RAJIV GANDHI

SALAI, CHENNAI – 600119

MARCH - 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY



(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **THIMMARAJU DURGA DHANASRI JYOTHIRMAI (Reg. No. 37110779), THALLAPAKA SRAVANA LAKSHMI (37110771)** who carried out the project entitled "**ADAPTIVE AUTHENTICATION OF BANKNOTES USING DEEP NEURAL NETWORKS**" under my supervision from **November 2020 to March 2021**.

Internal Guide

Mr. MURARI.D.KAMALESH M.E., (Ph.D.,)

Head of the Department

Dr. S. Vigneswari, M.E., Ph.D.,

Dr. L. Lakshmanan, M.E., Ph.D.,

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I **THIMMARAJU DURGA DHANASRI JYOTHIRMAI (Roll No. 37110779)** hereby declare that the Project Report entitled “**ADAPTIVE AUTHENTICATION OF BANKNOTES USING DEEP NEURAL NETWORKS**” is done by me under the guidance of **Mr.MURARI.D.KAMALESH M.E., (Ph.D.,)** Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mr. MURARI.D.KAMALESH M.E., (Ph.D.,) Associate Professor**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Banknotes are one of the most important assets of a country. Some miscreants introduce fake notes which bear a resemblance to original note to create discrepancies of the money in the financial market. It is difficult for humans to tell true and fake banknotes apart especially because they have a lot of similar features. Fake notes are created with precision, hence there is need for an efficient algorithm which accurately predicts whether a banknote is genuine or not. This project contains a feasibility study of deep neural networks for the classification & authentication of banknotes with respect to requirements of central banks on the ATM and high speed sorting industry.

TABLE OF CONTENTS

	ABSTRACT	v
	LIST OF FIGURES	viii
CHAPTER No.	TITLE	PAGE No.
1.	INTRODUCTION	1
	1.1 OVERVIEW 1.2 SCOPE OF PROJECT 1.3 DOMAIN OVERVIEW 1.3.1 DNN CLASSIFIER	
2.	AIM AND SCOPE	2
	2.1 LITERATURE SURVEY	2
	2.1.1 EXSISTING SYSTEM 2.1.2 CONSTRAINTS OF EXSISTING SYSTEM	2
	2.2 AIM & SCOPE	2
	2.2.1 PROPOSED SYSTEM	2
	2.2.2 SCOPE OF THE PROPOSED SYSTEM	3
3.	EXPERIMENTAL MATERIALS OR METHODS AND ALGORITHMS USED	5
	3.1 SOFTWARE REQUIREMENTS 3.1.1 OPERATING SYSTEM 3.1.2 JUPYTER NOTEBOOK 3.1.3 TENSORFLOW 3.1.4 KERAS AND SCIKIT LEARN 3.1.5 PANDAS AND NUMPY 3.1.6 MATPLOTLIB AND SEABORN 3.1.7 FLASK 3.1.8 STEAMLIT 3.1.9 HEROKU	5
	3.2 HARDWARE REQUIREMENTS	5
	3.3 ALGORITHMS USED	7

	3.3.1 DNN CLASSIFIER	
4.	SYSTEM DESIGN & METHODOLOGY	16
	4.1 SYSTEM ARCHITECTURE 4.2 DATASET DESCRIPTION 4.3 MODULES 4.3.1 DATA PREPARATION, TRAIN & SPLIT 4.3.2 FEATURE EXTRACTION & SELECTION 4.3.3 DNN TRAINING 4.3.4 DNN PERFORMANCE OF THE MODEL 4.3.5 TESTING & VALIDATION WITH INTEGRATED UI	
5	RESULTS & DISCUSSION CONCLUSION AND FUTURE WORK	
	REFERENCES	20
	APPENDIX	29
	A. PAPER ACCEPTANCE MAIL B. PLAGIARISM REPORT C. JOURNAL PAPER D. SOURCE CODE	22

LIST OF FIGURES

S.No	FIGURES	Page No
1.	Tensorflow	9
2.	Sklearn	9
3.	Architecture of DNN	11
4.	System Architecture	12
5.	Scaling	13
6.	Data Split	14
7.	Feature Extraction	14
8.	Jupyter Notebook	15
9.	Model Evaluation	16
10.	Homepage	16
11.	Parameters as input	17
12.	Plotting the dataset	18

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Despite a decrease in the use of currency due to the recent growth in the use of electronic transactions, cash transactions remain very important in the global market. Banknotes are used to carry out financial activities. To continue with smooth cash transactions, entry of forged banknotes in circulation should be preserved. There has been a drastic increase in the rate of fake notes in the market. Fake money is an imitation of the genuine notes and is created illegally for various motives. These fake notes are created in all denominations which brings the financial market of the country to a low level. The various advancements in the field of scanners and copy machines have led the miscreants to create copies of banknotes. It is difficult for human-eye to recognize a fake note because they are created with great accuracy to look alike a genuine note. Security aspects of banknotes have to be considered and security features are to be introduced to mitigate fake currency. Hence, there is a dire need in banks and ATM machines to implement a system that classifies a note as genuine or fake.

1.2 SCOPE OF THE PROJECT

The main contributions of this project therefore are:

- Data Analysis
- Dataset Preprocessing
- Training the Model
- Testing of Dataset

1.3 DOMAIN OVERVIEW

Deep learning has evolved hand-in-hand with the digital era, which has brought about an explosion of data in all forms and from every region of the world. This data, known simply as big data, is drawn from sources like social media, internet search engines, ecommerce platforms, and online cinemas, among others. This enormous amount of data is readily accessible and can be shared through fintech applications like cloud computing. Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example.

Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before. However, the data, which normally is unstructured, is so vast that it could take decades for humans to comprehend it and extract relevant information. Companies realize the incredible potential that can result from unravelling this wealth of information and are increasingly adapting to AI systems for automated support.

1.3.1 DNN CLASSIFIER

Deep neural networks are a powerful category of machine learning algorithms implemented by stacking layers of neural networks along the depth and width of smaller architectures. Deep networks have recently demonstrated discriminative and representation learning capabilities over a wide range of applications in the contemporary years. Researchers in ML are expanding the horizons of deep learning by seeking their prospective applications in other diverse domains. One such forthcoming domain is marine scene classification. Deep networks require a large amount of annotated data for training. With efficient training algorithms, deep neural networks are capable of separating millions of labelled images. Moreover, the trained network can also be used for learning efficient image representations for other similar benthic data sets.

Deep neural networks with several layers have recently become a highly successful and popular research topic in machine learning due to their excellent performance in many benchmark problems and applications. A key idea in deep learning is to learn not only the nonlinear mapping between the inputs and outputs but also the underlying structure of the data (input) vectors. The `DNNClassifier` calls `tf.estimator.DNNClassifier` from the TensorFlow Python API. This command builds a feedforward multilayer neural network that is trained with a set of labelled data in order to perform classification on similar, unlabeled data.

CHAPTER 2

AIM & SCOPE

2.1 LITERATURE SURVEY

2.1.1 EXSISTING SYSTEM

Banknotes are one of the most important assets of a country. Some miscreants introduce fake notes which bear a resemblance to original note to create discrepancies of the money in the financial market. It is difficult for humans to tell true and fake banknotes apart especially because they have a lot of similar features. Fake notes are created with precision, hence there is need for an efficient algorithm which accurately predicts whether a banknote is genuine or not.

Preserving genuineness of higher denomination printed Banknotes is one of the critical issues. It has the major role in financial activities of every country. The study in evaluates different machine learning algorithms and concludes that Decision-Tree and MLP technique is best to classify a bank note.

The base paper uses a supervised classification algorithm, random forest classifier, k-means clustering, SVM for differentiating genuine banknotes from fake ones.

2.1.2 CONSTRAINTS OF EXSISTING SYSTEM

The usage of different machine learning algorithms had different impacts on the data. To start with random forest, The main limitation of random forest is that a large number of trees can make the algorithm too slow and ineffective for real-time predictions. In general, these algorithms are fast to train, but quite slow to create predictions once they are trained. A more accurate prediction requires more trees, which results in a slower model. In most real-world applications, the random forest algorithm is fast enough but there can certainly be situations where run-time performance is important and other approaches would be preferred.

When coming to SVM, Choosing a “good” kernel function is not easy. Long training time for large datasets. Difficult to understand and interpret the final model, variable weights and individual impact. Since the final model is not so easy to see, we cannot do small calibrations

to the model hence it's tough to incorporate our business logic. The SVM hyper parameters are Cost -C and gamma. It is not that easy to fine-tune these hyper-parameters. It is hard to visualize their impact.

Coming to DNN, it is a special form of the well-known Expectation Maximization (EM) algorithm and assumes that all clusters are equally sized and have the same variances. In most of the cases this assumption is not satisfied. Clusters will differ in their size, density and variance. Violating this assumption will not make it impossible to use k-means, but one has to be careful interpreting the results.

2.2 AIM AND SCOPE

2.2.1 PROPOSED SYSTEM

This project contains a feasibility study of deep neural networks for the classification & authentication of banknotes with respect to requirements of central banks on the ATM and high-speed sorting industry.

Consider someone wants to deposit money in the bank. The notes that are to be deposited are given to a human being to check for their authenticity. As the fake notes are prepared with precision, it is difficult to differentiate them from genuine ones. A recognition system must be installed to detect legitimacy of the note. The system should extract the features of the note using image processing techniques. These features will be given as input to the machine learning algorithm which will predict if the note is true or fake

A system has been proposed in for recognition of banknotes. The study shows that three-layered perceptron is proved to classify the banknotes into a certain class accurately by taking the banknote images as inputs.

The dataset used to train the models is taken from UCI machine learning repository. Data were extracted from genuine and counterfeit banknote images. The dataset has 1372 instances. There are 5 attributes out of which 4 are the features and one is the target attribute.

TensorFlow's high-level machine learning API (tf.contrib.learn) makes it easy to configure, train, and evaluate a variety of machine learning models.

2.2.2 SCOPE OF THE PROPOSED SYSTEM

In traditional Machine learning techniques, most of the applied features need to be identified by a domain expert in order to reduce the complexity of the data and make patterns more visible to learning algorithms to work. The biggest advantage Deep Learning algorithms as discussed before are that they try to learn high-level features from data in an incremental manner. This eliminates the need of domain expertise and hardcore feature extraction.

Elaborately, a deep learning technique learn categories incrementally through its hidden layer architecture, defining low-level categories like letters first then little higher-level categories like words and then higher-level categories like sentences. In the example of image recognition, it means identifying light/dark areas before categorizing lines and then shapes to allow face recognition. Each neuron or node in the network represents one aspect of the whole and together they provide a full representation of the image. Each node or hidden layer is given a weight that represents the strength of its relationship with the output and as the model develops the weights are adjusted.

TensorFlow's DNN is useful because it can scale problems with no limit—nodes in a graph can run across a distributed network. The logic in TF is unique, relying on both a machine's CPU and its GPU. Adding in the graphical processor unit gives TF a lot more power per machine.

The benefits of TensorFlow include Increased functionality and Increased control. While Keras has many general functions for ML and deep learning, TF's is more advanced, particularly in high-level operations like threading and queues and debugging. You don't always need a lot of control, but some neural networks may require it so you have better understanding and insight, particularly when working with operations like weights or gradients.

CHAPTER 3

EXPERIMENTAL MATERIALS OR METHODS AND ALGORITHMS USED

3.1 SOFTWARE REQUIREMENTS

- Operating system : Windows 10
- Languages used : Python
- Python version : 3.5 or 3.6
- Notebooks : Jupyter Notebook
- Emulators : No emulators used
- Software Libraries : OS, Tensorflow, Keras, Pandas, Matplotlib, Numpy, Seaborn, Math, Sklearn.
- Frontend & Backend: Flask, Streamlit
- Cloud Hosting : Heroku

3.1.1. OPERATING SYSTEM:

WINDOWS 10:

Windows 10 is a series of personal computer operating systems produced by Microsoft as part of its Windows NT family of operating systems. It is the successor to Windows 8.1, and was released to manufacturing on July 15, 2015, and broadly released for retail sale on July 29, 2015. Windows 10 receives new builds on an ongoing basis, which are available at no additional cost to users, in addition to additional test builds of Windows 10 which are available to Windows Insiders. Devices in enterprise environments can receive these updates at a slower pace, or use long-term support milestones that only receive critical updates, such as security patches, over their ten-year lifespan of extended support.

3.1.2 JUPYTER NOTEBOOK:

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. Jupyter Notebook is built off of IPython, an interactive way of running Python code in the terminal using the REPL model (Read-Eval-Print-Loop). The IPython Kernel runs the computations and communicates with the Jupyter Notebook front-end interface. It also allows Jupyter Notebook to support multiple languages.

Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document.

Data Transformation -

In this part data is to be transformed such as data like prices in dollars and some other price in rupees and some in the form of euros etc., . So, all the data is transformed based on the convenient to the need.

Data Integration -

In this data is taken from the year 20018-2019 from website investing.com and another set of data from coin-desk.com from the year 2019-2020.And then the data is to be integrated into a single file. So that data integration is used in that way.

Data Splitting -

It splits the data into the two parts. which one part is for testing and another for training data with ratio 80%:20%.

PYTHON

Python is a high level and effective general use programming language. It supports multi-paradigms. Python has a large standard library which provide tools suited to perform various tasks. Python is a simple, less-clustered language with extensive features and libraries. Different programming abilities are utilized for performing the experiment in our work. In this thesis, the following python libraries were used.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open-source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

It is used for:

1. web development (server-side),
2. software development,
3. mathematics,
4. system scripting.

NumPy-

NumPy is a Python library used for working with arrays.

It also has functions for working in domain of linear algebra, fourier transform, and matrices.

NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.

NumPy stands for Numerical Python.

In Python we have lists that serve the purpose of arrays, but they are slow to process.

NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.

3.1.2 TENSORFLOW:

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.



FIG 1.0 Tensorflow

3.1.3 KERAS & SCIKIT LEARN:

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.

Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms. It's built upon some of the technology you might already be familiar with, like NumPy, pandas, and Matplotlib.



Fig 1.0.1 Sklearn

3.1.4 PANDAS & NUMPY:

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free

software released under the three-clause BSD license.

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

3.1.5 MATPLOTLIB & SEABORN:

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

3.1.6 FLASK

Flask is a lightweight [WSGI](#) web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around [Werkzeug](#) and [Jinja](#) and has become one of the most popular Python web application frameworks.

Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

3.1.7 STREAMLIT

Streamlit is an open source app framework specifically designed for ML engineers working with Python. It allows you to create a stunning looking application with only a few lines of code.

3.1.8 HEROKU

Heroku is a container-based cloud Platform as a Service (PaaS). Developers use Heroku to deploy, manage, and scale modern apps. Our platform is elegant, flexible, and

easy to use, offering developers the simplest path to getting their apps to market.

3.2 HARDWARE REQUIREMENTS

There are big data that require big hardware.

Learning about big machine learning requires big data and big hardware.

- Processor – Intel Xeon E2630 v4 – 10 core processor, 2.2 GHz with TurboBoost upto 3.1 GHz. 25 MB Cache
- Motherboard – ASRock EPC612D8A
- RAM – 128 GB DDR4 2133 MHz
- TB Hard Disk (7200 RPM) + 512 GB SSD
- GPU – Nvidia TitanX Pascal (12 GB VRAM)
- Intel Heatsink to keep temperature under control
- Disk space: 1 GB
- Operating systems: Windows 10, macOS, and Linux
- Python* versions: 3.5.x, 3.6.x
- 64-bit, x86 desktops or laptops with dedicated Nvidia graphics card

3.3 ALGORITHMS USED

3.3.1 DNN CLASSIFIER

Deep neural networks are a powerful category of machine learning algorithms implemented by stacking layers of neural networks along the depth and width of smaller architectures. Deep networks have recently demonstrated discriminative and representation learning capabilities over a wide range of applications in the contemporary years. Researchers in ML are expanding the horizons of deep learning by seeking their prospective applications in other diverse domains. One such forthcoming domain is marine scene classification. Deep networks require a large amount of annotated data for training. With efficient training algorithms, deep neural networks are capable of separating millions of labeled images. Moreover, the trained network can also be used for learning efficient image representations for other similar benthic data sets.

Deep neural networks with several layers have recently become a highly successful and popular research topic in machine learning due to their excellent performance in many benchmark problems and applications. A key idea in deep learning is to learn

not only the nonlinear mapping between the inputs and outputs but also the underlying structure of the data (input) vectors. The DNNClassifier calls tf.estimator.DNNClassifier from the Tensorflow Python API. This command builds a feedforward multilayer neural network that is trained with a set of labeled data in order to perform classification on similar, unlabeled data.

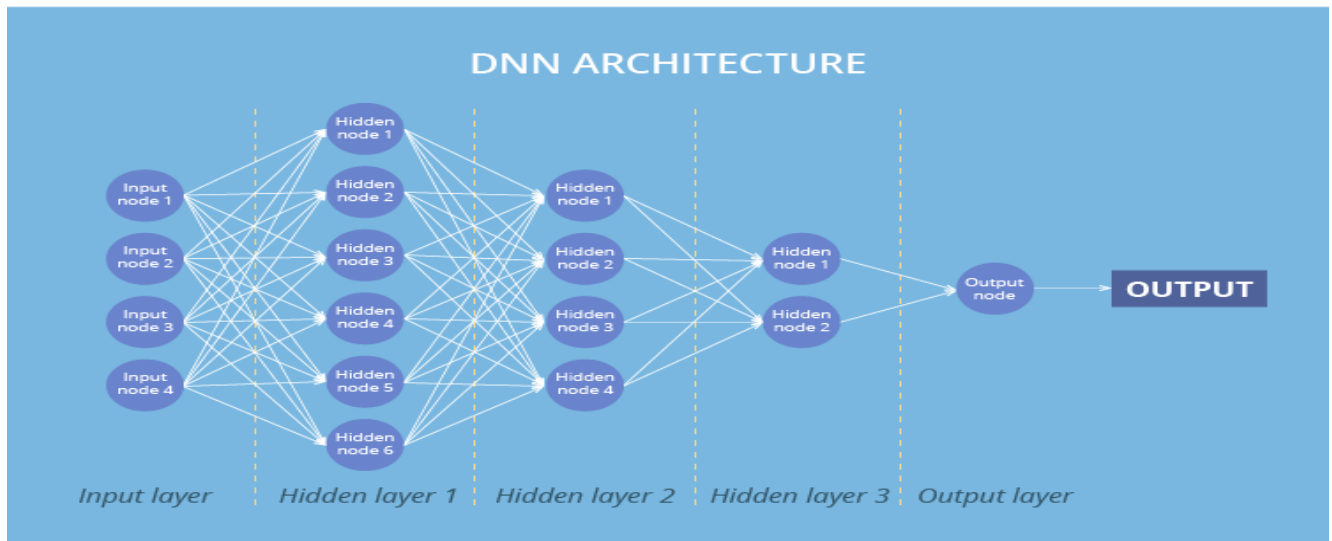


FIG 1.1 ARCHITECTURE OF DNN

CHAPTER 4

SYSTEM DESIGN AND METHODOLOGY

4.1 SYSTEM ARCHITECTURE

A system has been proposed in for recognition of banknotes. The study shows that three-layered perceptron is proved to classify the banknotes into a certain class accurately by taking the banknote images as inputs.

TensorFlow's high-level machine learning API (tf.contrib.learn) makes it easy to configure, train, and evaluate a variety of machine learning models

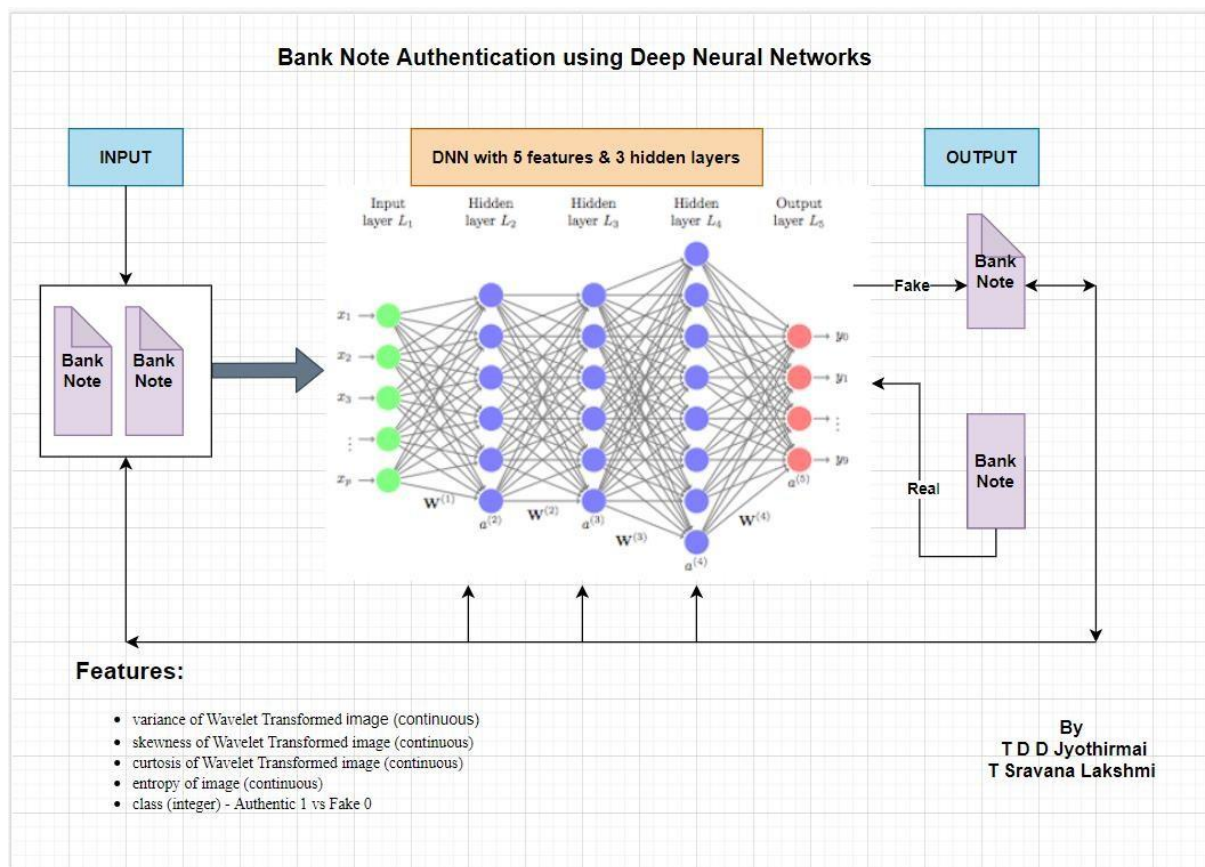


FIG 1.2 System Architecture

4.2 DATASET DESCRIPTION

Dataset link: <https://archive.ics.uci.edu/ml/datasets/banknote+authentication>

The dataset used to train the models is taken from UCI machine learning repository . Data were extracted from genuine and counterfeit banknote images. The dataset has 1372 instances. There are 5 attributes out of which 4 are the features and one is the

target attribute. The dataset contains a balanced ratio of both classes which is 55:45(genuine: counterfeit). The target class contains two values: 0 and 1 where 0 represents genuine note and 1 represents fake note.

Variance of Wavelet Transformed Image: Continuous data, Variance finds how each pixel varies from the neighbouring pixels and classifies them into different regions .

Skewness of Wavelet Transformed image: Continuous data, Skewness is the measure of the lack of symmetry .

Kurtosis of Wavelet Transformed image: Continuous data, Kurtosis is a measure of whether the data are heavytailed or light-tailed relative to a normal distribution .

Entropy of image: Continuous data, Image entropy is a quantity which is used to describe the amount of information which must be coded for, by a compression algorithm . Class Integer Class contains two values 0 representing genuine note and 1 representing fake note.

4.3 MODULES

4.3.1 Data Preparation, Train & Split:

The dataset has been divided into train and test data. Approximately 70 percent of the data is taken for training and remaining 30 percent for testing the model.

Data Preparation

Standard Scaling

```
In [8]: from sklearn.preprocessing import StandardScaler
```

```
In [9]: scaler = StandardScaler()
```

```
In [10]: scaler.fit(data.drop('Class',axis=1))
```

```
Out[10]: StandardScaler(copy=True, with_mean=True, with_std=True)
```

```
In [11]: scaled_features = scaler.fit_transform(data.drop('Class',axis=1))
```

```
In [12]: #Converting Scaled Features to a DataFrame
df_feat = pd.DataFrame(scaled_features,columns=data.columns[:-1])
df_feat.head()
```

```
Out[12]:
```

	Image.Var	Image.Skew	Image.Curt	Entropy
0	1.121806	1.149455	-0.975970	0.354561
1	1.447066	1.064453	-0.895036	-0.128767
2	1.207810	-0.777352	0.122218	0.618073
3	1.063742	1.295478	-1.255397	-1.144029
4	-0.036772	-1.087038	0.736730	0.096587

FIG 1.3 Scaling

The target variables are all the variables that the dataset contains. Because every component is important for training the model.

Train Test Split

```
In [14]: x = df_feat
In [15]: y = data['Class']
In [16]: # Using the .as_matrix() method in order for TensorFlow to accept the data in Numpy array form instead of a pandas series.
x = x.as_matrix()
y = y.as_matrix()
In [17]: from sklearn.model_selection import train_test_split
In [18]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)
```

FIG 1.4 Data split

4.3.2 Feature Extraction & Selection

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. A characteristic of these large data sets is a large number of variables that require a lot of computing resources to process. Feature extraction is the name for methods that select and /or combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set.

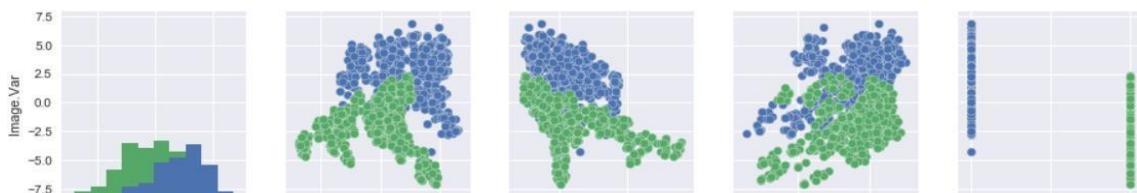


FIG 1.5 Feature Extraction

4.3.3 DNN Training: Tensorflow Estimators (tf.contrib.learn)

It is used for creating feature columns, hidden layers and to configure & train.

Using Tensorflow and Contrib.learn

```
In [19]: import tensorflow as tf

In [20]: feature_columns = tf.contrib.learn.infer_real_valued_columns_from_input(x_train)
WARNING:tensorflow:float64 is not supported by many models, consider casting to float32.

In [22]: #Due to the Warning, Converting into float32
import numpy as np
x_train = np.float32(x_train)
x_test = np.float32(x_test)

In [23]: feature_columns = tf.contrib.learn.infer_real_valued_columns_from_input(x_train)

In [24]: #Choosing a Dense Neural Network Classifier(DNNClassifier) with random hidden layers of [10, 20, 10]
#and classes as 2(Authentic 1 vs Fake 0)
classifier = tf.contrib.learn.DNNClassifier(feature_columns=feature_columns, hidden_units=[10, 20, 10], n_classes=2)

INFO:tensorflow:Using default config.
WARNING:tensorflow:Using temporary folder as model directory: C:\Users\apoor\AppData\Local\Temp\tmp91or3yj2
INFO:tensorflow:Using config: {'_task_type': None, '_task_id': 0, '_cluster_spec': <tensorflow.python.training.server_lib.ClusterSpec object at 0x000001EDBF552518>, '_master': '', '_num_ps_replicas': 0, '_num_worker_replicas': 0, '_environment': 'local', '_is_chief': True, '_evaluation_master': '', '_tf_config': gpu_options {
  per_process_gpu_memory_fraction: 1
}, '_tf_random_seed': None, '_save_summary_steps': 100, '_save_checkpoints_secs': 600, '_log_step_count_steps': 100, '_session_config': None, '_save_checkpoints_steps': None, '_keep_checkpoint_max': 5, '_keep_checkpoint_every_n_hours': 10000, '_model_dir': 'C:\\Users\\apoor\\AppData\\Local\\Temp\\tmp91or3yj2'}

In [26]: classifier.fit(x_train, y_train, steps=200)

WARNING:tensorflow: Casting <dtype: 'int64'> labels to bool.
WARNING:tensorflow: Casting <dtype: 'int64'> labels to bool.
INFO:tensorflow: Create CheckpointSaverHook.
INFO:tensorflow: Restoring parameters from C:\Users\apoor\AppData\Local\Temp\tmp91or3yj2\model.ckpt-200
INFO:tensorflow: Saving checkpoints for 201 into C:\Users\apoor\AppData\Local\Temp\tmp91or3yj2\model.ckpt.
INFO:tensorflow: loss = 0.0193855, step = 201
INFO:tensorflow: global_step/sec: 172.608
INFO:tensorflow: loss = 0.0110293, step = 301 (0.581 sec)
INFO:tensorflow: Saving checkpoints for 400 into C:\Users\apoor\AppData\Local\Temp\tmp91or3yj2\model.ckpt.
INFO:tensorflow: loss for final step: 0.00748629.

Out[26]: DNNClassifier(params={'head': <tensorflow.contrib.learn.python.learn.estimators.head.BinaryLogisticHead object at 0x000001EDBF552438>, 'hidden units': [10, 20, 10], 'feature columns': ( RealValuedColumn(column name='', dimension=4, default value=None, dtype=tf.float32, no
```

FIG 1.6 Jupyter Notebook

4.3.4 DNN Performance of the model

Following measures have been used to measure the performance of the models implemented.

- Accuracy – The accuracy of the test is its ability to differentiate the genuine and fake note test cases correctly. $\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN})$
- Sensitivity - The sensitivity of a test is its ability to determine the genuine note cases correctly. $\text{Sensitivity} = \text{TP} / (\text{TP} + \text{FN})$
- Specificity - The specificity of a test is its ability to determine the fake note cases correctly. $\text{Specificity} = \text{TN} / (\text{TN} + \text{FP})$
- Precision - The precision of a test is its ability to determine the number of notes that classifier labelled as genuine is actually genuine $\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$ Where,
- True Positive (TP) = the number of cases correctly identified as genuine notes.
- True negative (TN) = the number of cases correctly identified as fake notes.
- False positive (FP) = the number of cases incorrectly identified as genuine notes.
- False negative (FN) = the number of cases incorrectly identified as fake notes.

Model Evaluation

```
In [29]: note_prediction = list(classifier.predict(x_test, as_iterable=True))
INFO:tensorflow:Restoring parameters from C:\Users\apoor\AppData\Local\Temp\tmp91or3yj2\model.ckpt-400

In [30]: from sklearn.metrics import classification_report, confusion_matrix

In [31]: print(confusion_matrix(y_test, note_prediction))
[[227  0]
 [ 0 185]]

In [32]: print(classification_report(y_test, note_prediction))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	227
1	1.00	1.00	1.00	185
avg / total	1.00	1.00	1.00	412

FIG 1.7 Model Evaluation

4.3.5 Testing & Validation with INTEGRATED UI

We have given the test data as input to the Bank Note Authentication Web Application, which in turn detects whether the given data constitutes to a genuine bank note feature or not.

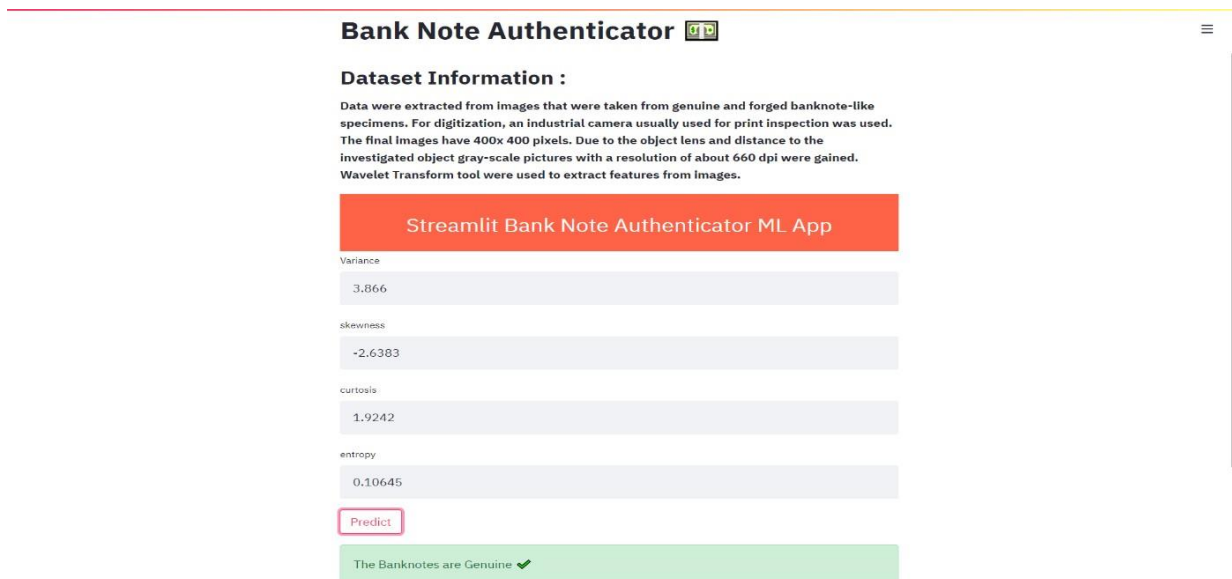


Fig 1.8 Homepage

Bank Note Authenticator



Dataset Information :

Data were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images.

Streamlit Bank Note Authenticator ML App

Variance

1.0552

skewness

1.1857

curtosis

-2.6411

entropy

0.11033

Predict

The Banknotes are Forged 🚫

Fig 1.9 Parameters as inputs

CHAPTER 5

RESULTS AND DISCUSSION

As a result of this system, the bank notes which are fake can be identified this model, a website is developed in order to make a simpler version to detect fake notes and separate the good ones from forged. At the bank end we used Deep Neural Network Classifier to test the data and split of data. In this system we have to countplot the dataset using seaborn. The analysis was done for the given data set.

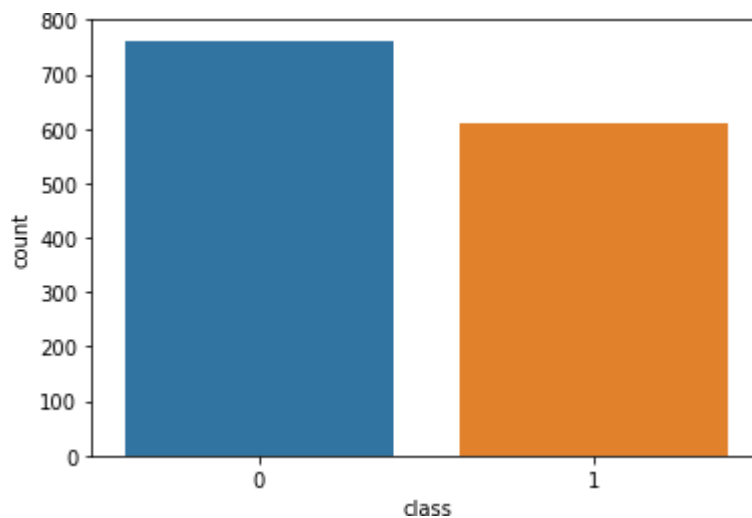


Fig 1.10 plotting the dataset

In the learning phase, the models are trained with 80% data i.e., 1097 samples out of which 609 samples were of genuine notes and 488 were of fake notes. For testing the models, remaining 275 samples have been used, where 153 samples were of genuine notes and 122 samples were of fake notes. The system evaluates the performance of two models and results have been shown in screenshots. The experiments performed using this dataset as input has resulted in a system providing high recognition rate of banknotes. The goal to get high accuracy of prediction is fulfilled by DNN. As shown in table 3, DNN correctly predicts the genuine and counterfeit notes. At the same time, SVM lacks in predicting the genuine notes and forged notes and gives sensitivity and specificity of 98.69% and 99.18% respectively. These results have been obtained by performing hold-out operation on the data. The results of training and test data do not show much difference except for sensitivity of SVM. The training sensitivity of DNN classifier is 98.68% and the test sensitivity is 98.69%. DNN gives 100% detection rate and SVM gives 98.90% success rate.

CHAPTER 6

CONCLUSION

After analysing various techniques used to detect forged banknotes, this paper presents banknote authentication for recognizing the banknote as genuine or fake by using two supervised learning techniques. Extensive experiments have been performed on banknotes dataset using both the models to find the best model suitable for classification of the notes. ROC and other metrics have been calculated to compare the performances of both the techniques. The result shows that back-propagation neural network outperforms support vector machine and gives 100% success rate. These techniques are an efficient way of solving the problem for all banking machines that accept all types of notes. We need to quantify how good our model is for you to implement in your network. Given that we just have two outcomes, real and forged, we proceeded to cluster the data around two points, where each point that belongs to a cluster shares similar features.

FUTURE WORK

In future, this work can be extended by categorizing the notes into different categories as Genuine, Low-Quality forgery, High-Quality forgery, Inappropriate ROI. And also in this work can be continued by implementing image processing of bank notes. A system which can take the input as images of notes and gives the output as fake or genuine. This work can be done by using Deep Neural network.

REFERENCES

- [1]. The 7th python in Science Conference 2007, Aric Hagberg, Daniel Schult and Pieter Swart. Exploring network structure, dynamics, and function using network. Proceedings.
- [2]. Journal of Machine Learning Research, 2010. Tom Schaul, Justin Bayer, Daan Wierstra, Yi Sun, Martin Felder, Frank Sehnke, Thomas Rückstieß, and Jurgen Schmidhuber, Pybrain.
- [3]. Ali Feizollah, Nor Badrul Anuar, Fair Amalina, Comparative study of k-means and mini batch k-means clustering algorithms in android malware detection using network traffic analysis, published in 2014 International Symposium on Biometrics and Security Technologies (ISBAST)
- [4]. Sculley D (2010), "Web-scale k-means clustering", Proceedings of the 19th international conference on World Wide Web, Raleigh, North Carolina, USA, pp. 1177-1178.
- [5]. Python (2014), "python", Available at: www.python.org (Accessed: 1st March 2014).
- [6]. Scikit-learn (2013), "sklearn. Cluster. MiniBatchKMeans", Available at the link given below as: <http://scikitlearn.org/stable/modules/generated/sklearn.Cluster.MiniBatchKMeans.html> (Accessed: 1st March 2014).
- [7]. Maulik U, Bandyopadhyay S: Genetic algorithm based clustering technique. Pattern Recognition. 2000, 33: 1455-1456. 10.1016/S0031-3203(99)00137-5.
- [8] Prachi Damodhar Shahare and Ram Nivas Giri, "Comparative Analysis of Artificial Neural Network and Support Vector Machine Classification for Breast Cancer Detection", International Research Journal of Engineering and Technology, Dec-2015.
- [9] Fumiaki Takeda, Lalita Sakoobunthu and Hironobu Satou, "Thai Banknote Recognition Using Neural Network and Continues Learning by DSP Unit", International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, 2003.
- [10] Costas Nastoulis, Apostolos Leros, and Nikolaos Bardis, "Banknote Recognition Based On Probabilistic Neural Network Models", Proceedings of the 10th WSEAS

International Conference on SYSTEMS, Vouliagmeni, Athens, Greece, July 10-12, 2006.

[11] Sigeru Omatu, Michifumi Yoshioka and Yoshihisa Kosaka, "Bank Note Classification Using Neural Networks", IEEE, 2007.

.

APPENDIX
PLAIGARISM REPORT

fp paper (3).docx

ORIGINALITY REPORT

9%

SIMILARITY INDEX

8%

INTERNET SOURCES

1%

PUBLICATIONS

0%

STUDENT PAPERS

PRIMARY SOURCES

1

mafiadoc.com

Internet Source

5%

2

www.ijcaonline.org

Internet Source

3%

3

"Multimedia Content Representation,
Classification and Security", Springer Science
and Business Media LLC, 2006

Publication

1%

Exclude quotes On

Exclude matches Off

Exclude bibliography On

ADAPTIVE AUTHENTICATION OF BANK NOTES USING DEEP LEARNING

T.SRAVANA LAKSHMI[1],T.D.D.JYOTHIRMAI[2],KAMALESH[3]

[1][2] UG Student, Dept. Of Computer Science and Engineering, Sathyabama Institute of Science and Technology, Chennai

sravanithallapaka@gmail.com dhanajyothirmai415@gmail.com

[3] Professor,Dept. of Computer Science and Engineering, Sathyabama Institute of Science and Technology, Chennai.

ABSTRACT

Banknotes are one of a country's most valuable assets. Some criminals introduce fake notes that look similar to the original notes in order to cause money discrepancies in the financial market. Humans find it difficult to distinguish between genuine and counterfeit banknotes, especially because they share many characteristics. Because fake notes are made with precision, there is a need for an efficient algorithm that predicts whether a banknote is genuine or not. This project includes a feasibility report of deep neural networks for the classification and authentication of banknotes in accordance with central bank requirements on ATMs, and bank employees can use this model to detect forged and genuine notes.

1. INTRODUCTION:

The undertaking discusses the certified receipts and their fraud. These days, programmed machines which acknowledge banknotes are created to address different issues and capacities. These machines perceive banknotes took care of to them by distinguishing the plan or estimation of the banknotes. It is critical for such machines to validate the banknote to recognize it among genuine and fake notes.

2.LITERATURE SURVEY:

Protecting validity of higher category printed Banknotes is one of the basic issues.It has the significant job in monetary exercises of each

When all is said in done, validation is more troublesome than acknowledgment, since the distinctions in plans or qualities are intentionally intended to be promptly recognized, while phonies are intentionally made for expected purposes to be vague from veritable banknotes.

The goals of this research are:

- To examine how to order the fake banknotes and what is the current conceivable answer for recognize it and to track down the best strategy to distinguish fake banknotes location technique.
- To create wanted model of banknote verification to produce the improved aftereffect of genuine or fake notes utilizing the Deep Neural Network method.
- To test gave information proposed, by utilizing the Deep Neural Network method in a carafe web application to get the advanced outcome. At that point, the web application will be utilized to dissect the class of information that is given from the UCI Machine Learning Repository data set and it will confirm if those banknotes' information is fake information.

country.This study assesses the profound learning classifier is utilized for confirmation of bank notes.And additionally checks that Deep learning is ideal to characterize a bank note.The highlights of the certified receipt are separated

as 4 credits i.e variance,skewness,entropy and kurtosis.

Existing system:

Banknotes are perhaps the main resources of a country. A few lowlives present phony notes which bear a similarity to unique note to make inconsistencies of the cash in the monetary market. It is hard for people to distinguish valid and counterfeit banknotes particularly in light of the fact that they have a ton of comparable highlights. Counterfeit notes are made with exactness, subsequently there is need for a proficient calculation which precisely predicts if a banknote is authentic.

The base paper utilizes an administered arrangement calculation, arbitrary timberland classifier, k-implies grouping, SVM for separating certified banknotes from counterfeit ones.

Proposed system:

This paper contains a plausibility investigation of profound neural organizations for the classification and verification of banknotes concerning prerequisites of national banks on the ATM and fast arranging industry. Consider somebody needs to store cash in the bank. The notes that are to be saved are given to an individual to check for their genuineness. As the phony notes are set up with exactness, it is hard to separate them from certifiable ones. An acknowledgment framework should be introduced to recognize authenticity of the note.A framework has been proposed in for acknowledgment of banknotes. The examination shows that three-layered perceptron is demonstrated to order the banknotes into a specific class precisely by taking the banknote pictures as sources of info. The dataset used to prepare the models is taken from UCI AI storehouse. Information were removed from certifiable and fake banknote pictures. The dataset has 1372 occurrences. There are 5 credits out of which 4 are the highlights and one is the objective attribute.TensorFlow's significant level AI API (tf.contrib.learn) makes it simple to design, train, and assess an assortment of AI models.

TensorFlow's DNN is valuable since it can scale issues with no restriction hubs in a

diagram can stumble into an appropriated network. The rationale in TF is special, depending on both a machine's CPU and its GPU. Including the graphical processor unit gives TF significantly more force per machine.

3. METHODOLOGY:

This paper is an investigation about the Banknote Authentication utilizing Deep Neutral Network . The exploration is to introduce a development that will give strategies, that are useful for banknote confirming or ensuring physical. In this exploration, the information were extricated from pictures that were taken for the assessment of a confirmation methodology for banknotes.

3.1EXISTING DATA DESCRIPTION

Attribute Name	Value Type	Description
Variance	Continuous	Variance finds how each pixel varies from the neighboring pixels and classifies them into different regions
Skewness	Continuous	Skewness is the measure of the lack of symmetry.
Kurtosis	Continuous	Kurtosis is a measure of whether the data are heavy-tailed or light-tailed relative to a normal distribution
Entropy	Continuous	Image entropy is a quantity which is used to describe the amount of information which
Class	Integer	Class contains two values 0 representing genuine note and 1 representing fake note

Figure 1:Description of the Attributes of notes

3.2MODULES

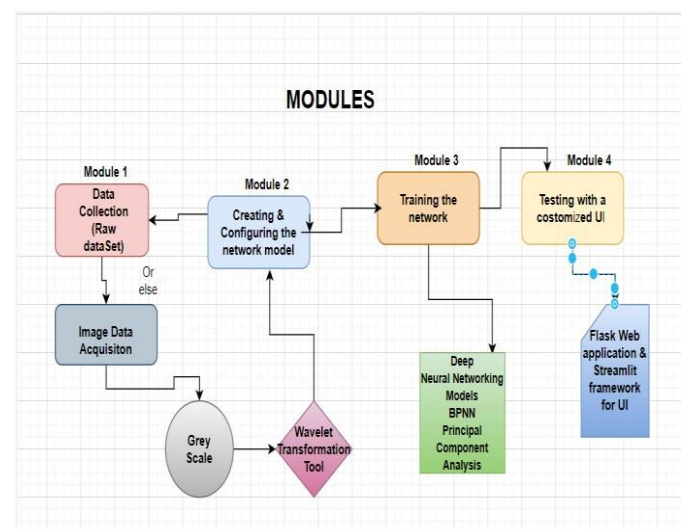


Figure 2:Modules Flowchart

3.3 WORK FLOW OF DATA

3.3.1 Collect data:

The information were extricated from the data set and also can be taken from the pictures that were taken from veritable and manufactured banknote-like examples. The last pictures have 400x 400 pixels. Wavelet Transform device were utilized to remove highlights from pictures. The information ascribes data is fluctuation, skewness, curtosis, entropy and class.

3.3.2 Create the network

Profound neural organizations with a few layers have as of late become an exceptionally fruitful also, famous exploration theme in AI because of their magnificent presentation in numerous benchmark issues and applications. A vital thought in profound learning is to learn not just the nonlinear planning between the sources of info and yields yet additionally the underlying design of the information (input) vectors. The DNNClassifier calls tf.estimator. DNNClassifier from the Tensorflow Python API. This order constructs a feedforward multi-facet neural organization that is prepared with a bunch of named information to perform grouping on comparable, unlabeled information.

3.3.3 Configure the network

Assign all the parameters of network name, input data, target data, training function, properties, transfer functions and etc. to create appropriate Neural Network 'net'. Initial values of the parameters were automatically generated by the command.

3.3.4 Train the network

Boundaries related with the training like error objective, greatest number of epochs (cycles), and so

on, are characterized. At that point, the preparation calculation is called. In this interaction, the creator chose to utilize 90% information for preparing, 5% for testing and 5% for approval.

3.3.4 Validate the network After the neural organization has been resolved, the outcome is first tried by mimicking the yield of the neural organization with the deliberate info information. This is contrasted and the deliberate yields. Overall testing & validation must be carried out with remaining data and flask UI.

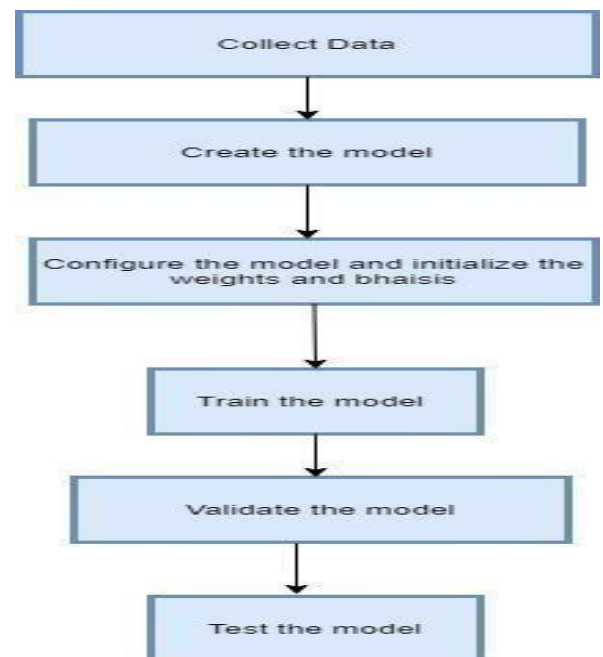


Figure 3:Steps of the model creation

4 EXPERIMENTS AND RESULTS:

4.1 Deep Neural Network with tensorflow:

Significant learning is a piece of Machine Learning and an unprecedented kind of Artificial Neural Network framework (ANN) that resembles the multifaceted human discernment system. Significant learning is at present getting a lot of thought for its utilization with tremendous social protection data. Significant taking in, a methodology with its foundation in ANN, is ascending actually as a basic

Profound neural organizations with a few layers have as of late become an exceptionally fruitful. Furthermore, well known exploration subject in AI because of their magnificent presentation in numerous benchmark issues and applications. A critical thought in profound learning is to learn not just the nonlinear planning between the sources of info and yields yet additionally the underlying construction of the information (input) vectors. The DNNClassifier calls tf.estimator. DNNClassifier from the Tensorflow Python API. This order assembles a feedforward multi-facet neural organization that is prepared with a bunch of marked information to perform characterization on comparable, unlabeled information.

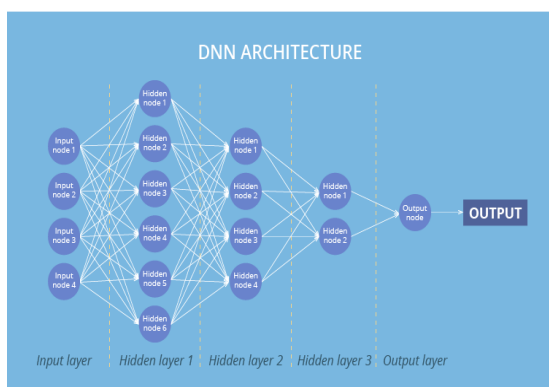


Figure 4:DNN Architecture

4.2 Back Propagation Neural Network (BPNN) :

The back ending neural framework estimation is an essential one of the neural framework. The computation is the readiness or learning estimation rather than the actual framework. Back inducing neural framework computation is a controlled learning system used for planning ANN. Getting ready is ordinarily finished by iterative invigorating of burdens reliant on the mix-up banner. By then the bumble banner is back multiplied to the lower layers. Back inducing is a drop computation which attempts to restrict the goof rate at each accentuation. The back expansion technique is a method used in planning multi-facet neural frameworks in a controlled manner. The back causing strategy, in any case got back to the

misstep multiplication estimation, relies upon the screw up change learning rule [5]. It contains two goes through the unmistakable layers of the framework: a forward pass and a backward pass. In the forward pass, a development configuration is associated with the information centers of the framework, and its effect spreads through the framework layer by layer. Finally, a great deal of yields are conveyed as the genuine response of the framework. In the midst of the forward pass the synaptic heaps of the frameworks are out and out settled. In the midst of the backward pass, the synaptic burdens are by and large adjusted according to a mix-up correction rule. The genuine response of the framework is deducted from an ideal response to make a screw up banner. This bungle banner is then caused backward through the framework. The synaptic burdens are changed as per make the real response of the framework move closer to the ideal response from a quantifiable perspective. The heap alteration is made by the summarized delta rule [8] to restrict the error.

4.3 Principal Component Analysis (PCA):

Principal Component Analysis (PCA) procedure [6] [7] is for the most part used for dimensionality decline and recorded an inconceivable execution in face affirmation. PCA based systems can be apportioned in two phases: planning and gathering. In the planning stage, an eigenspace is worked from the readiness tests using PCA system. In gathering stage, a data face is expected to the identical eigenspace and described by an appropriate classifier, for instance, Euclidean partition or Bayesian [3]. PCA encodes information in a balanced direct space while the Linear Discriminant Analysis (LDA) procedure encodes uncalled for information in a straight separable space of which bases are not actually even. In this paper, the PCA and LDA strategies are used for dimensionality diminishing and feed forward neural framework (FFNN) classifier is used for gathering of appearances. The proposed procedures are called PCA-NN and LDA-NN exclusively. The methods include two phases which are the PCA or LDA

preprocessing stage, and the neural framework game plan stage. The proposed structures demonstrate upgrade for the affirmation rates over the normal LDA and PCA stand up to affirmation systems that usage Euclidean Distance based classifier. Head part examination (PCA) and direct discriminant assessment (LDA) [7] and discrete cosine change (DCT) are three essential systems used for data decline and feature extraction in the appearance-based procedures.

4.4 Output through Web Application

To test information provided, by utilizing the Deep Neural Network strategy in a jar web application with Streamlit as the UI, to get the enhanced outcome. The testing information is taken from "Banknotes Authentication Data Set" and the source is from the Center for Machine Learning and Intelligent System. At that point, the web application will be utilized to investigate the class of information that is given from the UCI Machine Learning Repository data set and it will confirm if those banknotes information is fake information.

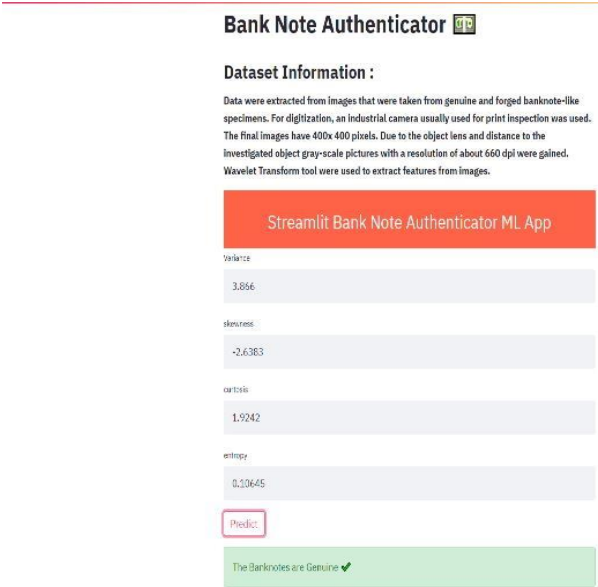


Figure 5:Screenshot of webpage

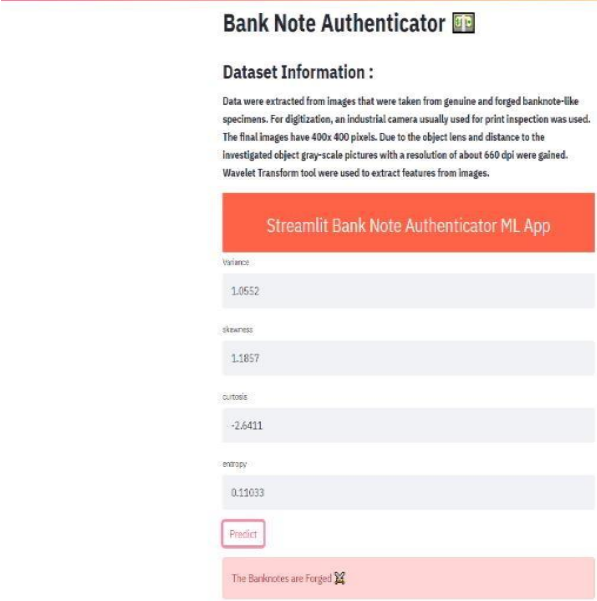


Figure 6:Screenshot of output

CONCLUSION

The tests performed utilizing this dataset as info has brought about a framework giving high acknowledgment pace of banknotes. The objective to get high precision of expectation is satisfied by TensorFlow, BPN. As demonstrated, DNN accurately predicts the authentic and fake notes. Simultaneously, SVM needs foreseeing the certified notes and manufactured notes and gives affectability and particularity.

Subsequent to investigating different methods used to distinguish manufactured banknotes, this paper presents banknote validation for perceiving the banknote as veritable or phony by utilizing two regulated learning procedures. ROC and different measurements have been determined to think about the exhibitions of both the strategies. These methods are a proficient method of tackling the issue for all banking machines that acknowledge a wide range of notes. This model can be taken ahead by using image processing with the notes and then predicting the notes.

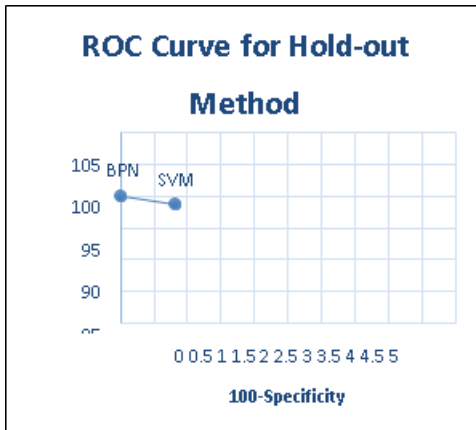


Figure 7: ROC graph plot picture comparing SVM & BPN

REFERENCES:

- [1]. The 7th python in Science Conference 2007, Aric Hagberg, Daniel Schult and Pieter Swart. Exploring network structure, dynamics, and function using network. Proceedings.
- [2]. Journal of Machine Learning Research, 2010. Tom Schaul, Justin Bayer, Daan Wierstra, Yi Sun, Martin Felder, Frank Sehnke, Thomas Rückstieß, and Jürgen Schmidhuber, Pybrain.
- [3]. Sculley D (2010), "Web-scale k-means clustering", Proceedings of the 19th international conference on World Wide Web, Raleigh, North Carolina, USA, pp. 1177-1178.
- [4]. Python (2014), "python", Available at: www.python.org (Accessed: 1st March 2014).
- [5]. Scikit-learn (2013), "sklearn. Cluster. MiniBatchKMeans", Available at the link given below as: <http://scikitlearn.org/stable/modules/generated/sklearn.Cluster.MiniBatchKMeans.html> (Accessed: 1st March 2014).
- [6]. Maulik U, Bandyopadhyay S: Genetic algorithm based clustering technique. Pattern Recognition. 2000, 33: 1455-1456. 10.1016/S0031-3203(99)00137-5.
- [7] Prachi Damodhar Shahare and Ram Nivas Giri, "Comparative Analysis of Artificial Neural Network and Support Vector Machine Classification for Breast Cancer Detection", International Research Journal of Engineering and Technology, Dec-2015.

SOURCE CODE & OUTPUT

Direct link to web app:

FLASK APP:

from flask import Flask, render_template, redirect, request	
	import numpy as np
	import pandas as pd
	import pickle
	app = Flask(__name__)
	pickle_in = open('classifier.pkl', 'rb')
	classifier = pickle.load(pickle_in)
	@app.route("/")
	def predict_note_authentication():
	variance = request.args.get('variance')
	skewness = request.args.get('skewness')
	curtosis = request.args.get('curtosis')
	entropy = request.args.get('entropy')
	prediction = classifier.predict([[variance, skewness, curtosis, entropy]])
	return "The predicted values is " + str(prediction)
	@app.route("/predict_file", methods=["POST"])
	def predict_file():
	df_test = pd.read_csv(request.files.get('file'))
	prediction = classifier.predict(df_test)
	return "The predicted values for the csv is " + str(list(prediction))
	if __name__ == '__main__':

	app.run()

from fastapi import FastAPI, File, UploadFile	
	import uvicorn
	import pandas as pd
	import pickle
	app = FastAPI()
	PATH = 'classifier.pkl'
	with open(PATH, 'rb') as file:
	classifier = pickle.load(file)
	@app.post("/predict/{variance,skewness,curtosis,entropy}")
	def predict(variance: float, skewness: float, curtosis: float, entropy: float):
	labels = [variance, skewness, curtosis, entropy]
	features = [[variance, skewness, curtosis, entropy]]
	to_predict = pd.DataFrame(features, columns=labels)
	prediction = classifier.predict(to_predict)
	return {"predicted value:": int(prediction)}
	@app.post("/predict_file")
	async def predict_file(file: UploadFile = File(...)):
	dataframe = pd.read_csv(file.file)
	prediction = classifier.predict(dataframe)
	print(prediction)
	return {"prediction": str(prediction)}

	if __name__ == '__main__':
	uvicorn.run(app, host="0.0.0.0", port=8000)

STREAMLIT APP:

import streamlit as st	
	import pickle
	# Save and Load Model
	pickle_in = open("classifier.pkl", "rb")
	classifier = pickle.load(pickle_in)
	# A custom function for predicting the values
	def predict_note_authentication(variance, skewness, curtosis, entropy):
	prediction = classifier.predict([[variance, skewness, curtosis, entropy]])
	print(prediction)
	return prediction
	def main():
	st.title("Bank Note Authenticator 🏧 ")
	st.markdown("""
	## **Dataset Information : **
	Data were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images.
	""", True)

	html_temp = ""
	<div style="background-color:tomato;padding:10px">
	Streamlit Bank Note Authenticator ML App
	</div>
	"""
	st.markdown(html_temp, unsafe_allow_html=True)
	variance = st.text_input("Variance", "Variance")
	skewness = st.text_input("skewness", "skewness")
	curtosis = st.text_input("curtosis", "curtosis")
	entropy = st.text_input("entropy", "entropy")
	result = ""
	if st.button("Predict"):
	result = predict_note_authentication(variance, skewness, curtosis, entropy)
	if result == 0:
	st.success('The Banknotes are Genuine ✔')
	else:
	st.error('The Banknotes are Forged ❌')
	#st.success('The output is {}'.format(result))
	if st.button("About"):
	st.markdown("""**Built with ❤ by Sravani and Jyothirmai**""")
	if __name__ == '__main__':
	main()

Bank Note Authenticator



Dataset Information :

Data were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images.

Streamlit Bank Note Authenticator ML App

Variance

1.0552

skewness

1.1857

curtosis

-2.6411

entropy

0.11033

Predict

The Banknotes are Forged 🚫

FIG 1.11 :Forged notes detection using parameters.

Bank Note Authenticator



Dataset Information :

Data were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images.

Streamlit Bank Note Authenticator ML App

Variance

3.866

skewness

-2.6383

curtosis

1.9242

entropy

0.10645

Predict

The Banknotes are Genuine ✅

FIG 1.12 :Genuine notes detection using parameters as input.