Comparative analysis of Convolutional Neural network-based counterfeit detection: keras vs. Pytorch

Emmanuel Balogun and Hayden Wimmer [0000-0002-2811-4531]

Georgia Southern University, Statesboro GA, USA
eb20284@georgiasouthern.edu, hwimmer@georgiasouthern.edu*

Abstract. The proliferation of advanced printing and scanning technologies has worsened the challenge of counterfeit currency, posing a significant threat to national economies. Effective detection of counterfeit banknotes is crucial for maintaining the monetary system's integrity. This study aims to evaluate the effectiveness of two prominent Python libraries, Keras and PyTorch, in counterfeit detection using Convolutional Neural Network (CNN) image classification. We repeat our experiments over 2 data sets, one dataset depicting the 1000 denomination of the Colombian peso under UV light and the second dataset of Bangladeshi Taka notes. The comparative analysis focuses on the libraries' performance in terms of accuracy, training time, computational efficiency, and the model behavior towards datasets. The findings reveal distinct differences between Keras and PyTorch in handling CNN-based image classification, with notable implications for accuracy and training efficiency. The study underscores the importance of choosing an appropriate Python library for counterfeit detection applications, contributing to the broader field of financial security and fraud prevention.

Keywords: Convolutional Neural Network, Image classification, MaxPooling, Activation Function, Keras Library, Pytorch Library, Counterfeit Detection
Acknowledgement – *This work was support in part by the National Science Foundation (USA) under Grant no. 2321939*

INTRODUCTION

Currency is a symbol of a nation's economic strength and stability. It has been an essential medium of exchange for centuries, maintaining the human phycological pleasure driven called human want by ensuring smooth business transactions, and establishing trust between the parties. However, as technology has advanced, so too have the methods of forging this very symbol of trust. Innovations in scanning and printing technologies have made it increasingly easier for counterfeiters to produce fake currency notes that are nearly inseparable from the real ones to the naked eye. Every nation, regardless of its economic strength, faces the challenge concerning counterfeit currency. These fake notes, when introduced into the economy, disintegrate the trust in the currency system, resulting to potential inflation, loss of revenue, and destabilization of the whole economy. Additionally, counterfeit money is often linked to other criminal activities, including terrorism, drug trafficking, and money laundering, making it also a significant threat to the national security. Banks and financial institutions invest heavily in sophisticated machines to detect these counterfeit notes. Yet, the adaptability and rapid evolution of counterfeiting methods render many of these machines useless within a short span. Traditional methods of detecting counterfeit money, reliant on human expertise and manual processes, are proving to be inefficient in the face of this growing challenges. This paper will investigate supervised machine-learning approach for banknote genuity detection systems through their images under ultraviolet light. Using a deep learning algorithm, a subset of machine learning comprises tons of artificial neural networks capable of recognizing patterns and characteristics in any given input (images) when trained. This logic will be introduced to distinguish genuine banknotes from counterfeit ones through their features and patterns similarities. This approach will be compared using two different Python programming language libraries Keras and PyTorch library by analyzing their performance to determine the one that is more effective.

The methods include two different phases, the first phase is image classification with CNN. In this phase, necessary libraries will be imported in the python console environment and dataset containing thousand images of genuine and counterfeit 1000 Colombian peso under ultraviolent light will be loaded for preprocessing where resizing to a uniform size, normalizing pixel values to a standard range (typically 0 to 1), and potentially applying data augmentation techniques like labeling the images into batch (2 batches) according to their Genuity (fake or real) to diversify the training set. Following that, CNN Architecture will design with selected parameters for training the model. Validation data will be introduced to validation for the model performance by helping prevent overfitting and ensures the model generalizes well to new data after training the model. Next is evaluating the model performance after training on a completely separate test dataset, this provides an unbiased assessment of its accuracy and effectiveness. The second phase is comparison between TensorFlow Keras and Pytorch libraries by following the phase method using the same parameters in both libraries and record their performance to determine their similarities and differences.

The result is expected to generate the differences between the libraries as we will be training the classification model using the same dataset with the same parameter settings. This proposed comparison will be able to determine the best library to adopt among the two libraries based on their accuracy differences for classification of UV light images. What is new in this research is to investigate the effectiveness of different machine learning libraries in the field of image classification, specifically focusing on UV light images. Previous research has explored image classification using machine learning techniques; however, there is a notable absence of comparative analyses assessing the performance of different libraries in this context. By identifying the most suitable library for processing this unique dataset, it will help to enhance security measures against the circulation of counterfeit banknotes.

RELATED WORKS

Islam, et al. [1] undertook a rigorous analysis comparing the sigmoid and ReLU activation functions to evaluate their effectiveness. The selection of an appropriate activation function is crucial for optimizing the performance of a CNN architecture. According to their methodological approach, the image dataset was first pre-processed and subsequently trained using a CNN architecture rooted in the LeNet design. The CNN was then subjected to both the sigmoid and ReLU activation functions. Their findings indicated that the sigmoid activation function outperformed the ReLU activation function in terms of efficiency [1]. Mrudula, et al. [2] compare the efficiency of different CNN architectures toward plants image classification. Among them are ResNet-50, Alexnet, Inception V3, and VGG-16. The result shows that ResNet-50 succeeded others by proving the best accuracy [1]. Kausar [3] did a comparative study on support vector machine (SVM) and CNN approaches towards image classification, in his method, a dataset consisting of 24,000 images of pets is introduced to both models using 80 percent for training and 20 percent for testing. SVM is said to fail as it doesn't have the capability to train such a huge amount of data. 4000 images are then introduced using the same percentage for training and testing. The result shows that SVM has 61% precision accuracy, making it inferior to CNN, which has 89% precision accuracy [3]. Liu, et al. [4] also confirm this by taking the same approach and using the Modified National Institute of Standards and Technology (MNIST) image dataset to train both classifiers. Their CNN model superseded SVM in precision accuracy, confirming CNN is the right choice for image classification [4].

Alnowaini, et al. [5] identified rapid advancements in color printing, scanning, and duplicating technologies as primary culprits behind the surge in counterfeit banknotes, particularly in Yemen. They introduced a robotic counterfeit detection system that leveraged Support Vector Machines (SVM) post feature extraction. Their findings showcased the system's swifter and more accurate performance compared to the Fuzzy Logic Method [5]. Wang, et al. [6] carried out a comprehensive comparison between traditional machine learning and deep learning techniques to ascertain their relative significance in image classification. For this analysis, the MNIST dataset was trained using Support Vector Machines (SVM) as a representative of traditional machine learning algorithms, and CNN as a representative of deep learning algorithms. Authors found when utilizing a large sample MNIST dataset, SVM achieved an accuracy of 0.88, while the CNN achieved an accuracy of 0.98 and for a smaller sample from the COREL1000 dataset, SVM's accuracy stood at 0.86, compared to CNN's 0.83. Overall, the results showcased that traditional machine learning algorithms tend to exhibit superior performance with smaller datasets. In contrast, deep learning frameworks demonstrate heightened recognition accuracy when dealing with larger datasets [6].

Kumar, et al. [7] propose banknote denomination detection and authenticity detection using machine learning and deep learning respectively, to tackle the challenges of hike in the circulation of counterfeit notes in India. According to their method, different images of genuine Indian banknotes ranging from 10 - 500 in denomination was used for their model, different machine learning algorithm including K-Nearest Neighbor, Support Vector Machine, Decision Tree Classifier, and Random Forest Classifier were employed to detect the currency with their denomination and then pass it to CNN to learn and predict the authenticity. Comparative analysis was conducted on the machine learning algorithms and Support Vector Machine prove to have an overall detection accuracy with 99% accuracy. For the authenticity detection, CNN proved to be effective with 98% accuracy [7]. Kim, et al. [8] tackled the challenge of distinguishing flying drones from birds using a deep learning object detection approach. Their proposed method leveraged the state-of-the-art YOLOV8, considered by [9] as the foremost CNN image detection algorithm. To enhance the detection of minuscule objects, a Multi-scale Image Fusion (MSIF) technique was integrated, along with the addition of a P2 layer to the YOLOV8 design. Post-implementation, the Average Precision (AP) of object detection was assessed on testing videos, setting a threshold of 0.5 for Intersection over Union (IoU) to deem a detection accurate. With the enhanced YOLO-V8-M model, frame rates of 17.6 fps and 45.7 fps were achieved at image resolutions of 1280 and 640 respectively, covering all processes from preprocessing to the Non-Maximum Suppression (NMS) stage. This method offers valuable insights, particularly regarding the enhancement of object detection performance for small objects [8].

Singh, et al. [10] underscored the severe repercussions of currency counterfeiting on both micro and macro-economic scales. While prevailing countermeasures involve intricate hardware, often inaccessible to the average person, Singh and his team presented a novel, hardware-independent authentication system for identifying counterfeit Indian banknotes. This system employs image processing in YCrCB, LUV, and HSV color spaces, capitalizing on the visibility of security features in these models. A clustering algorithm is then introduced for feature detection and classification, using template matching based on the HOG descriptor. Their dataset, comprising 20 counterfeit and 40 genuine banknotes, underwent training, with 20 counterfeit and 20 genuine notes reserved for testing. Their results highlighted a 90% accuracy based on the security thread and a flawless 100% accuracy using the latent image. This research is pertinent to my own, especially considering the neural network-based classification and feature extraction through template matching [10]. Desai, et al. [11] emphasized the growing challenge of banknote recognition, particularly with the advent of sophisticated printing techniques that render counterfeit bills virtually indifferent to the naked eye. Addressing this, Desai proposed a Generative Adversarial Network (GAN)-based approach for classifying counterfeit and genuine Indian banknotes. The three-phase method begins with preprocessing, followed by generator learning, and culminating in discriminator learning. When compared with the conventional CNN model, GAN displayed superior result in the area of creating a fake note to maximize the dataset. Incorporating GAN could potentially augment the performance of my ongoing project, given their similarities [11]. Just like [11], Addressing the limited access to diverse banknote image datasets, a challenge that plagues counterfeit detection research, Khemiri, et al. [12] proposed a Semi Supervised Generative Adversarial Network (SSGAN) approach. The SSGAN, an unsupervised learning model, comprised two neural networks. By introducing a dataset containing Tunisian banknotes, GAN-generated fake images were produced. The subsequent neural network then assessed authenticity based on the actual and generated images. Comparative analysis underscored the superiority of this model over other machine learning algorithms [12].

Zhang and Yan [13] explained the inherent limitations of manual currency authentication. They championed a deep learning-driven image recognition technique for currency validation, which they argued surpasses human visual assessment in accuracy. Their method processed video-captured images into a machine-readable format and employed the Single Shot MultiBox Detector (SSD) for real-time object detection. The final CNN model was tasked with extracting security features of the currency for classification. Their results underscored the importance of accurate positioning in classification and identification, offering insights into the significance of dataset positioning in model accuracy [13]. Lee, et al. [14] developed an image classifier to detect counterfeit printouts from Korean home-based laser jet and inkjet printers, focusing on their unique halftone feature patterns. The proposed model, which achieved a perfect accuracy score, offers a promising avenue for future research [14]. Like [14], Lee and Lee [15] employed a CNN image classifier model to discern genuine Korean banknotes from counterfeit printouts from various printer models. Emphasizing the importance of filtering and maxpooling in the convolutional layer of the CNN architecture, the model demonstrated a flawless accuracy in distinguishing genuine banknotes from counterfeits [15].

METHODS

Convolutional Neural Networks (CNN) for Image Classification

CNN are a class of deep neural networks primarily known for its widely utilization and its efficiency in analyzing visual imagery. They have a specialize architecture optimized for processing grid-like data structures, such as images. CNNs process images in layers, with each layer responsible for detecting different features, from simple edges to complex patterns. The importance of CNN in image similarity lies in its ability to extract hierarchical features from images:

- Low-level Features: In the initial layers, CNNs identify basic patterns like edges, textures, and gradients.
- Mid-level Features: As it moves deeper, the network begins to recognize more complex structures like shapes, repetitive patterns, and object parts.
- High-level Features: In the final layers, the network can detect entire objects or specific features pattern that define the uniqueness of an image.

Through the extraction and analysis of intricate features, CNNs are proficient at determine the similarities and discrepancies between two images. This capability is particularly crucial in scenarios where differences might not be immediately apparent to the human eye. For example, in the case of currency notes, both genuine and counterfeit notes may appear identical at a cursory glance. However, they possess distinct componential variations, such as unique watermark characteristics, holographic strip patterns, and specific microprinting details. CNNs leverage these intricate feature patterns, effectively differentiating between genuine and counterfeit notes by analyzing and comparing the nuanced similarities and variances present in the image data. Such precise detection is achieved through CNNs' advanced pattern recognition abilities, which are integral in ensuring the accuracy and reliability of counterfeit detection systems. Below diagram illustrates the CNN architecture system with the different layers it comprised.

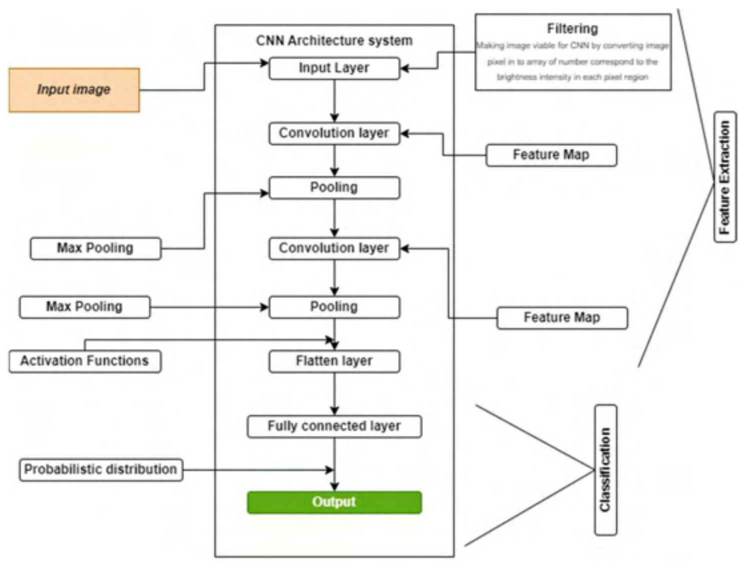


Fig. 1. CNN Architecture System

Comparing CNN across Different Libraries

Deep learning is a dynamic field, with multiple libraries and frameworks available for researchers and developers. Each of these libraries offers different optimizations, tools, and functionalities for implementing CNNs. By comparing CNN's performance across two different libraries, we can ascertain which one offers better accuracy, speed, and efficiency in detecting counterfeit notes. For instance, a library might be optimized for faster processing but might compromise slightly on accuracy. Another might offer incredibly detailed detection but could be resource-intensive. By conducting a comparative study, we can determine which library offers the best balance of speed, accuracy, and efficiency. This knowledge is crucial for financial institutions and governments, as it allows them to invest in the most effective tools for combating counterfeit currency.

The Python programming language was selected for this project due to its widespread adoption in the scientific and machine learning communities, coupled with its extensive libraries and straightforward syntax. This makes Python an ideal choice for developing complex machine learning models efficiently. We employed the Keras and PyTorch libraries, renowned for their robustness and popularity in the Python ecosystem, particularly in the development of neural networks. For the development and training of the CNN model, Google Colab's Integrated Development Environment (IDE) was utilized. Google Colab is particularly advantageous for building CNN networks due to its provision of a high-performance computing environment accessible via cloud. This includes the availability of GPUs and TPUs, which significantly accelerate the training process of deep learning models, making it an ideal platform for handling computationally intensive tasks inherent in CNN training.

This analysis utilizes two distinct datasets to assess the model's performance. UV light is commonly employed for detecting counterfeit notes, and thus, we incorporated a dataset of banknotes exposed to UV light. Additionally, a dataset under normal conditions was tested. Dataset 1 comprises over 2,500 images, encompassing both authentic and counterfeit Colombian peso notes with a denomination of 1000, captured under ultraviolet light. This dataset was sourced from Mendeley-Data, a reputable online repository for research datasets. Conversely, Dataset 2 comprises a more extensive collection of approximately 8,500 images. These images represent various denominations of Bangladeshi Taka, featuring both genuine and counterfeit notes. This dataset was sourced from Kaggle, a prominent online platform known for its comprehensive dataset collections. Prior to introducing the dataset to the CNN model, the images were uniformly resized, a critical step to ensure consistency in input size. This uniformity is essential for the CNN to effectively learn and extract features, as CNNs require fixed-size input tensors. Consequently, all images in the dataset were resized to a standardized dimension of 28x28 pixels. In this model, approximately 75% of the dataset was allocated for training, while the remaining 25% was reserved for testing and evaluation. The architecture incorporated two convolutional layers and two max pooling layers for feature mapping. Following image preprocessing, which rendered the images suitable for the CNN network, they underwent a series of training and testing iterations across different epochs. The outcomes were systematically recorded based on their corresponding library and epoch duration. A comparative analysis was subsequently conducted, and confusion matrix was plotted, focusing on the performance variance between the libraries. This model consists of seven principal components, each integral to the model's overall functionality, and are detailed in the following sections.

This is the initial engineering stage where images are prepared for analysis. Preprocessing steps include image resizing, labeling, normalization, augmentation, or color conversion. The Dataset is resized to 64 by 64 dimension giving the model a less computational power, which is needed to process the data faster, speeding up the training and inference stages and reduce the hardware requirements. It also reduces the noise in the images to improve the robustness of the feature extraction process, leading to more accurate classifications. Normalization converts pixel intensities into numerical values within a standardized range, typically 0 to 255. By scaling the pixels based on their intensity, the process not only facilitates more efficient network training but also accelerates convergence during the learning process. Prepossessing ensures that the input image is in a consistent format suitable for feeding into a CNN.

IMAGE CLASSIFICATION USING CNN.

CNN learn information from images through layering system, wherein each layer is design to identify complex feature sets within the image data. This is achieved by employing a kernel or filter (a predefined matrix of weights) across the normalized input image. The kernel traverses the image's entirety, performing convolution operations that mathematically combine the kernel values with the underlying pixel intensities to produce feature maps. These feature maps represent distilled versions of the original image, highlighting salient features essential for pattern recognition. This process, known as feature mapping, systematically reduces the spatial dimensions of the image while preserving critical structural details, thus preparing the data for the following layers to perform more complex analyses.

FEAUTURE MAPPING

2D convolution use 2-dimension filter, each filter is designed to detect specific types of features, such as edges, corners, or textures. The filter moves across the input image in a sliding window manner, from left to right and top to bottom, one pixel at a time. At each position, the convolution operation performs a dot product between the filter values and the underlying pixel values of the image. The result of each convolution operation is a matrix that forms a feature map.

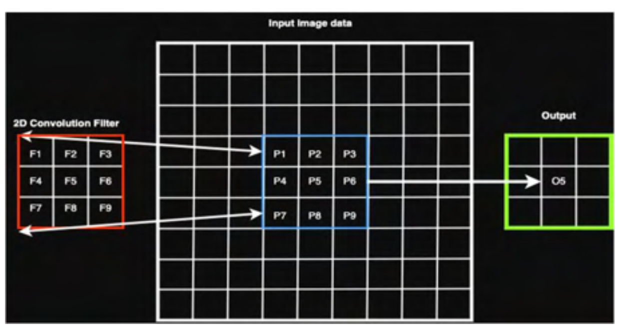


Fig. 3. 2D convolution, **output** is derived from the dot product of **Input** and **filter.**

Output O = Filter F [] * Input image P []

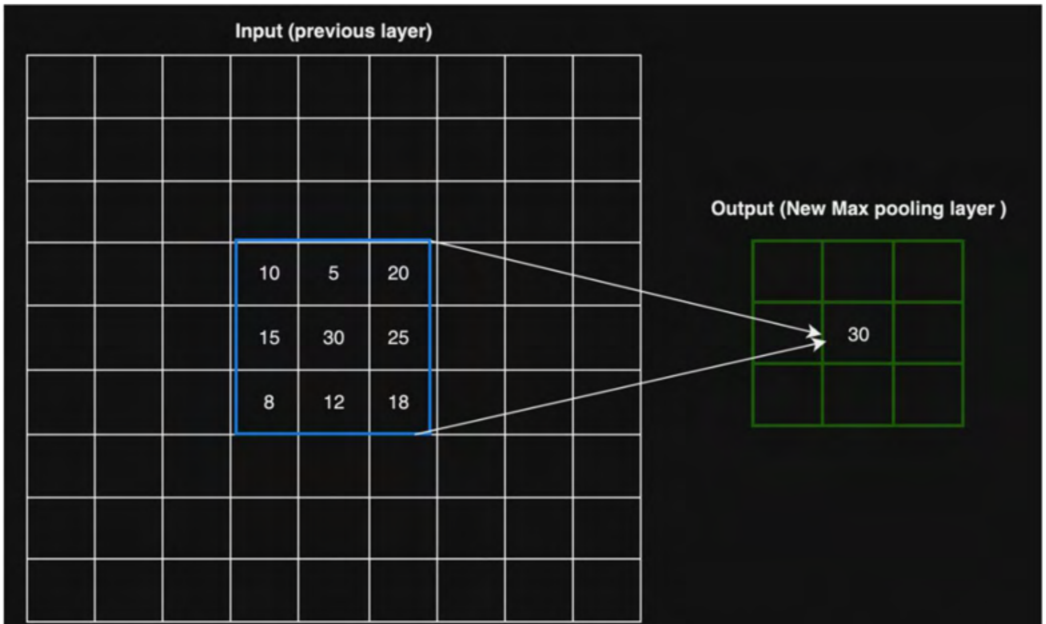
$$O\,[i, j] = \sum_m \sum_n I[i + m, j + n] \times F[m, n]$$

Figure 3 shows the visual representation of 2D convolution, and the mathematical formulation is given by the above formula, where O[i, j] is the value of the output image at position (i j), I(i + m, j + n) represents the pixels of the input image and F(m, n) is the filter value at position (m, n).
Multiple filters are used to extract different features, and the resulting feature maps are stacked along the depth dimension to form the complete output of the convolution layer. CNNs will learn features in a hierarchical manner.

Simple features like edges are learned in the initial layers, and more complex features like textures and patterns are learned in deeper. In this module, two convolutional layer is created allowing it to handle noises in the image for a better precision result.

POOLING LAYER

In pooling layers, pooling function filter reduces the spatial size of the image data to reduce the number of parameters and computation in the network. This operation also can also be called subsampling or downsampling, it leverages detection of features somewhat invariant to scale and orientation changes.




```
# Define a 3x3 matrix (an array of arrays)
matrix = [
    [10, 5, 20],
    [15, 30, 25],
    [8, 12, 18]
]
# Initialize max_value to the minimum possible value
max_value = float('-inf')
# Iterate through the rows and columns of the matrix
for row in matrix:
    for element in row:
        # Update max_value if a larger value is found
        if element > max_value:
            max_value = element
# Print the maximum value
print(max_value)

Output>>>>>>30
```

Fig. 4. Visual and Code-line representation of CNN MaxPooling

Max pooling function takes the maximum value in each feature map as the output. This is done to reduce the risk of overfitting, creating more computational efficiency because of the reduction in image size because of this. It also helps to create an exact position of features into a more general form, which helps the model in recognizing patterns more broadly and aids in generalization to new, unseen data.

The fully connected layers learn global or overall patterns in the input data set by taking the high-level filtered images from previous layers. This layer is where the actual classification takes place. Based on the features learned during the convolutional and pooling stages, the FC layer determines the probabilities of the input image belonging to each class or label of the input dataset making it so essential to the module. While convolutional layers handle local feature patterns, fully connected layers learn global or overall patterns in the input data set. Each neuron is connected to every element in the flattened vector from the previous layer, weighted sum of its inputs, adds a bias, and then passes this sum through an activation function which introduce non-linearity to the model.

Real-world data is complex and non-linear, meaning that the relationship between the input variables and the output variable cannot be accurately follow a straight-line pattern, especially in a situation like this where there are many images containing thousands array of number values. Activation functions introduce non-linearity to this model, allowing it to learn and perform more complex tasks beyond what a linear model could do. There are different types of activation functions and in this model two most common is selected.

ReLU is first introduced to the model because it allows only positive values to pass through it and turns the negative value to zero preparing the model for a binary classification. It is currently the most widely used activation function due to its computational efficiency and the ability to alleviate the vanishing gradient problem. Below is the graphical demonstration of ReLU.

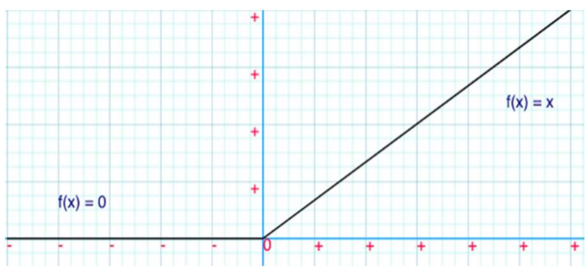


Fig. 5. ReLU Activation Function

The sigmoid activation function transforms any given input into a value within the 0 to 1 range, rendering it particularly useful for binary classification tasks. This characteristic facilitates the interpretation of the neuron's output as a probability, thereby making it a right choice for the model. Below is the graphical demonstration of Sigmoid function.

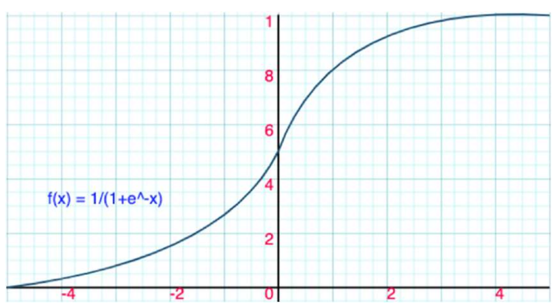


Fig. 6. Sigmoid Activation Function

In this model, backpropagation is used to calculate the gradient after each forward pass through the network. The Gradient is the derivatives of the loss function with respect to each parameter in the network (weights and biases). The gradient is then used to update the weights in the opposite direction of the gradient since it wants to minimize the loss, not to increase it. This is done using an optimization algorithm called gradient descent. The loss function quantifies the difference between the predicted outputs of the network and the actual target values. It provides a single scalar value that the training algorithms use as a signal to guide the optimization of the weights. Cross-entropy loss and Adam optimizer is used in this model to perform this task. The goal is to adjust the weights and biases to reduce the loss function's value, which corresponds to improving the model's predictions.

RESULT AND DISCUSSION:

Both datasets are preprocessed by resizing them uniformly, making them viable to CNN, then we pass them through our constructed CNN model in both libraries. The study results shown below were derived from analyses conducted across the libraries, focusing on variables that impact the model's efficacy over a range of epochs, from 10 to 200. Central to this comparative investigation was the examination of training and evaluation times for each model. These timeframes are pivotal in determining the practical viability and resource efficiency in real-world scenarios. Additionally, the models' proficiency in categorizing new, unseen images was assessed, employing the confusion matrix as the primary evaluative metric. This measure is crucial as it provides an in-depth view of the model's accuracy, particularly highlighting its capability in correctly identifying images and its limitations. The insights from this research offer a comprehensive understanding of the comparative strengths and weaknesses of each library in these essential aspects.

Dataset 1 Colombia currency under ultraviolet light

Table 1

Epoch #	Training Time (Minute)		Evaluation training time (Minute)	
	Keras	PyTorch	Keras	PyTorch
Epoch 10	8	11	2	4
Epoch 20	8	14	2	2
Epoch 50	9	35	1	4
Epoch 100	16	34	1	2
Epoch 200	32	117	2	2

Table 2

Epoch #	Predicted Result		Confusion Matric (%)	
	Keras	PyTorch	Keras	PyTorch
Epoch 10	Positive	Positive	**49**	**100**
Epoch 20	Positive	Positive	**51**	**100**
Epoch 50	Positive	Positive	**49**	**100**
Epoch 100	Positive	Positive	**49**	**100**
Epoch 200	Positive	Positive	**49**	**100**

There is a consistent increase in training time as the number of epochs grows for both Keras and PyTorch models. Notably, the PyTorch model takes a considerably longer time to train at higher epochs, suggesting a difference in computational efficiency between the two frameworks. Evaluation Training Time shows the time taken to evaluate the training at the 30th epoch. There is little variation across different total epochs, with times for both Keras and PyTorch staying mostly consistent or showing slight increases. This indicates that evaluation time does not significantly increase with the number of epochs for either framework.

Test accuracy is 100% for both frameworks across all epochs. While this suggests excellent performance, it may also indicate overfitting, as perfect accuracy is uncommon in real-world applications and can be a sign that the model has memorized the training data rather than learned to generalize from it. The predicted test results yield positive results, signifying that the models correctly classify genuine currency as authentic and counterfeit currency as fake when feed in both fake and genuine currency across all epochs for both Keras and PyTorch. This consistency is expected given the 100% test accuracy reported. Confusion Matrix shows that there is a stark contrast between the two libraries here. Keras shows approximately 49%-51% accuracy across epochs, which is no better than random guessing in a balanced binary classification task. Conversely, PyTorch maintains a 100% accuracy, aligning with the test accuracy and reinforcing the concern about potential overfitting.

Dataset 2 Bangladesh currency

Table 3

Epoch #	Training Time (Minute)		Evaluation training time (Minute)	
	Keras	PyTorch	Keras	PyTorch
Epoch 10	15	23	3	4
Epoch 20	59	50	10	11
Epoch 50	42	76	3	4
Epoch 100	66	152	3	4
Epoch 200	88	207	3	5

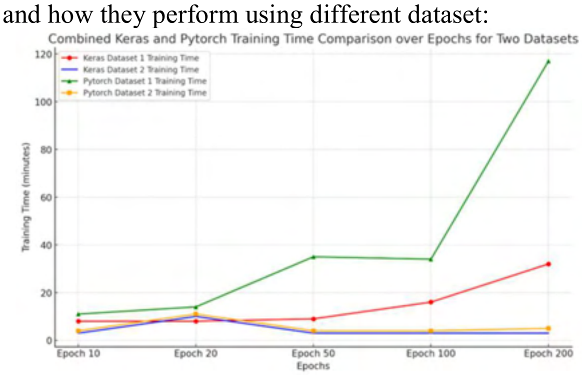
Table 4

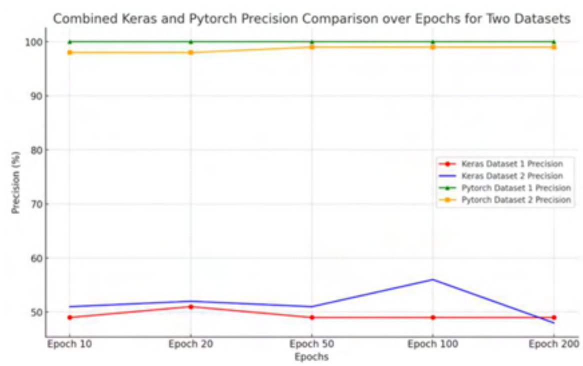
Epoch #	Predicted Result		Confusion Matric (%)	
	Keras	PyTorch	Keras	PyTorch
Epoch 10	Positive	Positive	**51**	**98**
Epoch 20	Positive	Positive	**52**	**98**
Epoch 50	Positive	Positive	**51**	**99**
Epoch 100	Positive	Positive	**56**	**99**
Epoch 200	Positive	Positive	**48**	**99**

Notable differences were observed in their predictive accuracy, training efficiency, and evaluation times. The model in PyTorch consistently outperformed the Keras model in precision across various epochs, maintaining an accuracy of 98-99%. In contrast, the model in Keras exhibited fluctuating accuracy percentages, ranging from 48% to 56%, which raises concerns about its learning stability, also potential overfitting issues. Regarding training efficiency, the Keras model demonstrated a clear advantage, requiring significantly less time to complete training across all epochs compared to PyTorch. This efficiency could be pivotal in scenarios where rapid model development and iteration are essential. Both models exhibited positive prediction when unknown newly test images are introduce and relatively stable evaluation times, suggesting good scalability in the evaluation and test phases.

Comparison between the Datasets

Below graphical representation indicates the relationship between training times, model precision across all epochs by using the result from the mode. This help to understand behavior and the model computation ability of each model and how they perform using different dataset:





Training Time and precision for both libraries

For Keras, there's a noticeable difference in training times between the two datasets, especially at higher epochs. Dataset 2 generally takes longer to train than Dataset 1, particularly evident at Epoch 200. Pytorch shows a similar trend, but the difference in training times between the datasets becomes more pronounced at higher epochs. Notably, Dataset 2 shows a steep increase in training time after Epoch 50. In general, there's a linear relationship between the training time and number of epochs during the training as increase in epochs result in increase in training time for both frameworks and datasets. Because of their volumes and complexity, training time in dataset 2 which is larger in volume doubles training time in dataset 1. This is expected as more epochs require more computational resources. In the

PyTorch framework, the model exhibits enhanced performance with Dataset 2 as compared to Dataset 1, achieving a consistent precision rate of 98-99% across various epochs. This high level of accuracy, sustained uniformly over time. Unlike Dataset 1 it does not indicate any signs of overfitting. Furthermore, the data suggests a potential linear correlation between the size of the dataset and the precision of the model in PyTorch. Conversely, a significant gap is observed in the precision trends when comparing the two libraries. The Keras framework demonstrates a precision range of 48-56% for both datasets. This relatively lower and narrower precision band suggests that the Keras model might benefit from tailored model configurations or adjustments in hyper-parameters to optimize learning effectiveness for each specific dataset.

DISCUSSION

The variance in training times between the libraries underscores the importance of choosing the most suitable model based on specific needs like computational resources and real-time analysis requirements. The adaptability of these models across various training epochs showcases their potential to evolve in response to new and emerging counterfeit types, thereby maintaining the effectiveness of detection methods. The result from both datasets gives insight into how complexity and volume of dataset can impact the effectiveness of CNN model, making this analysis useful for researchers in this domain against potential overfitting especially when dealing with low volume of dataset. This analysis not only demonstrates the significant impact of the choice of Python library on the efficiency and accuracy of CNN models in counterfeit detection but also contributes to ongoing efforts to enhance security measures against counterfeit currency. This analysis will also create a cost-effective security measure for small businesses, especially those that deal with handling currency on a daily basis. They can adopt high precision CNN models as a cost-effective means of counterfeit detection, reducing the need for expensive hardware or manual inspection. By employing effective counterfeit detection mechanisms like CNN, it will enhance trust and reliability in the domain of small businesses among their customers and partners. Accurate detection of counterfeit currency directly translates to the prevention of potential financial losses, an aspect particularly crucial for small businesses operating within margins. High precision CNN model will significantly increase the accuracy of counterfeit detection, reducing the circulation of counterfeit currency in the market, and the knowledge that advanced detection systems are in use can act as a deterrent to those considering the creation and distribution of counterfeit currency. Additionally, this analysis offers insights into how different frameworks perform with varied datasets, will give researchers insights, especially in the machine learning and deep learning domains, on improving model architectures for better accuracy in similar tasks, and will guide researchers in choosing or customizing models for specific image recognition challenges.

CONCLUSION

This paper embarked on a comparative analysis of two Python libraries, Keras and PyTorch, in the context of counterfeit detection using CNN image classification. By evaluating two different dataset of Colombian peso notes under ultraviolet light and Bangladeshi Taka, significant insights were collected regarding the performance of these libraries. Findings revealed noteworthy differences in their accuracy, training time, and computational efficiency between the libraries and datasets when applied to CNN-based image classification for counterfeit detection. The differences observed in the performance of Keras and PyTorch have profound implications for researchers in the field of image classification approach for counterfeit detection. The efficiency and accuracy of a CNN model in differentiating genuine currency from fake ones is important, and the choice of the library can significantly influence these aspects. The research highlights that while both libraries are capable, their suitability may vary depending on specific requirements like computational resources and desired accuracy levels. The study also opens avenues for further research. An exploration into more diverse datasets, including different currencies and varied lighting conditions, could provide more comprehensive insights. In conclusion, the comparative analysis between Keras and PyTorch in the realm of counterfeit detection using CNN image classification has demonstrated that the choice of the Python library can significantly impact the model's effectiveness. This study contributes to the ongoing efforts to enhance security measures against counterfeit currency, a critical step in maintaining economic stability and trust in financial institutions globally.

REFERENCES

[1] M. A. Islam, H. Wimmer, and C. M. Rebman, "Examining Sigmoid vs ReLu Activation Functions in Deep Learning," *Interdisciplinary Research in Technology and Management; CRC Press: Boca Raton, FL, USA,* pp. 432-437, 2021.

[2] K. Mrudula, H. R. Jain, J. Bhansali, and S. Sejal, "Plant-Wise Comparison of Various CNN Algorithms for Detection of Pest Infestation," in *2021 IEEE Pune Section International Conference (PuneCon)*, 2021: IEEE, pp. 1-7.

[3] A. Kausar, "Comparitive Analysis of Support Vector Machine and Convolutional Neural Network in Image Classification," in *2023 International Conference on Network, Multimedia and Information Technology (NMITCON)*, 2023: IEEE, pp. 1-6.

[4] T. Liu, J. Chen, and X. Li, "Research on Image Classification Based on Convolutional Neural Network," in *2022 International Conference on Automation, Robotics and Computer Engineering (ICARCE)*, 2022: IEEE, pp. 1-4.

[5] G. Alnowaini, A. Alabsi, and H. Ali, "Yemeni paper currency detection system," in *2019 first international conference of intelligent computing and engineering (ICOICE)*, 2019: IEEE, pp. 1-7.

[6] P. Wang, E. Fan, and P. Wang, "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning," *Pattern Recognition Letters,* vol. 141, pp. 61-67, 2021.

[7] C. P. Kumar, M. G. Yadav, K. Praneetha, M. Rushikesh, and R. Shreya, "Classification and Detection of Banknotes using Machine Learning," in *2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2023: IEEE, pp. 764-768.

[8] J.-H. Kim, N. Kim, and C. S. Won, "High-Speed Drone Detection Based On Yolo-V8," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023: IEEE, pp. 1-2.

[9] X. Jia, Y. Wang, and T. Chen, "Forest Fire Detection and Recognition Using YOLOv8 Algorithms from UAVs Images," in *2023 IEEE 5th International Conference on Power, Intelligent Computing and Systems (ICPICS)*, 2023: IEEE, pp. 646-651.

[10] M. Singh, P. Ozarde, and K. Abhiram, "Image processing based detection of counterfeit Indian Bank notes," in *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2018: IEEE, pp. 1-5.

[11] S. Desai, A. Rajadhyaksha, A. Shetty, and S. Gharat, "Cnn based counterfeit Indian currency recognition using generative adversarial network," in *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, 2021: IEEE, pp. 626-631.

[12] W. Khemiri, W. Jaafar, A. Tarifa, and J. B. Abderrazak, "Counterfeit money detection: A hybrid semi-supervised GAN-based approach," in *2022 International Interdisciplinary Conference on Mathematics, Engineering and Science (MESIICON)*, 2022: IEEE, pp. 1-6.

[13] Q. Zhang and W. Q. Yan, "Currency detection and recognition based on deep learning," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2018: IEEE, pp. 1-6.

[14] J. Lee, H. Kim, and T. Y. Kang, "Classification algorithm using halftone features of counterfeit bills and CNN," *Journal of Forensic Sciences,* vol. 67, no. 1, pp. 345-352, 2022.

[15] S.-H. Lee and H.-Y. Lee, "Detecting counterfeit bills and their forgery devices using CNN-based deep learning," in *Proc. 13th Int. Multi-Conf. Comput. Global Inf. Technol.*, 2018, pp. 16-20.

Utilization of Personalized PageRank for Protein Protein Interaction Analysis and Similarity-Based Complex Network Analysis: A Brief Review

Arjab Sengupta, Srijita Chakraborty, Malay Gangopadhyay

Department of Electronics and Communication, Institute of Engineering and Management, Kolkata 700091

Email: arjab.sengupta2022@iem.edu.in

Abstract

Getting solutions that are stringent and noteworthy to complex mathematical procedures through algorithms is not something new and has been a key for finding answers to tough questions. Such an algorithm extensively in use today is the PageRank algorithm. Proposed by Larry Page and Sergey Brin in 1996 and originally finding its roots in the Graph Theory, it has become a significant and solution method in cases where complex chain analysis is involved. The algorithm makes use of basic yet intricate graph theory concepts, such as in-degrees and out-degrees corresponding to each node to give weights to a particular webpage and find its consequent rank or weight. This paper discusses the very basics of the algorithm. The paper also reviews the work done by Gabor Ivan and Vince Grolmusz, who, in 2010, used PageRank to determine the protein interaction chains in our body and rank them according to their corresponding importance. In addition, this paper also has provided a brief idea on another original research based on similarity-based methods for link prediction in complex networks. The driving formula for finding the PageRank value of a webpage is: $W_i = d + \Sigma_{I=1, I \neq j} I_{ij} (W_i/n_i)$.

Brief elaboration on the significance of "d" or the damping factor in the above-mentioned formula by taking appropriate examples and through some light on the basic and structural working of the PageRank algorithm have also been done.

Keywords: Nodes, in-degrees, out-degrees, edges, protein-protein interaction chains (PPI).

Introduction

The problem of identifying nodes that are important in relatively large/big networks was prevalent in several fields and still is, but the answers to those problems till date used to appear in conjunction with that of the World Wide Web or commonly attributed as www. There are certain terms related to the graph theory which are: Nodes or vertices, in-degree, out-degree, edges or branches. Each of these terms serve to propose a significant meaning. Nodes basically refer to each point or junction of the meeting of two branches/edges. The in-degree refers to the number of incoming branches in each node whereas the out-degree refers to the total number of outgoing branches from each node. Each of the lines joining two nodes are known as vertices or branches. These uses are historically well established, mostly in matters where scientometry is involved. However, in cases involving the web graph, these degrees are easy to manipulate by simple adding infinite number of referring edges into the graph.

Also, the problem of link prediction is not something new and serves to be an axial field in complex network analysis. It also aims to establish new connections between nodes in a given network. As discussed later, such task may have a tremendous range of activities and outcome or applications, such as suggestion of friends in social media networks.