A Proposed Solution: Identifying Sensitive Information as a Safety Measure Against Privacy Vulnerabilities Associated With Optical Character Recognition

Farah Arafa Mohammed

Abstract

Optical character recognition (OCR) is a technology used to generate machine-readable text from images and documents; some OCR applications store extracted text in cloud storage, which has been proven to be not 100% secure for storing sensitive information. Therefore, items including sensitive information should not be processed and have their text extracted and stored to preserve the user's security, which is not applicable unless sensitive data is identified first. Based on the conducted research about this problem, the previous efforts, and what is currently available, this paper proposes a solution of identifying items including sensitive information, and preventing OCR applications that store extracted text in cloud storage from extracting text out of items including sensitive information. This research also tests the validity of the major part of the proposed solution, which is identifying items including sensitive data in the first place. To test the ability to identify sensitive data, a MobileNet neural network was trained four times to determine whether items include sensitive data. The results of testing MobileNet after the last training session demonstrated the validity of identifying sensitive information at a reliable level of accuracy in a short time, indicating promising results for the proposed solution if applied to a real OCR application in the presence of simple coding, including if-else statements.

1. Introduction:

Optical character recognition (OCR) is a technology used to convert images or scanned documents containing text into machine-readable text [1]. OCR applications use approaches such as rule-based methods and pattern-matching algorithms to identify text and make it machine-readable [2]. The input data is transmitted through an API from the customer's location where the OCR system is located to the vendor's location in the cloud, where the data extraction process takes place, then input and output data can be stored locally at the vendor's location in the cloud [2]. The most common storage options are as follows: The data could be stored in a buffer only during the execution of the data extraction process; the vendor

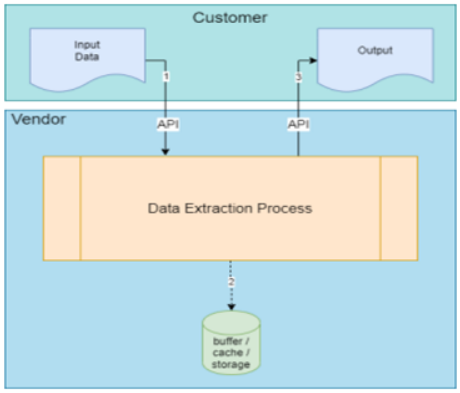

Fig (1) *Data flow in OCR systems hosted in the cloud* [2]

does not save the data after it has sent the output to the customer [2]. The data could also be temporarily stored in a cache; the data retention period depends on the cache memory capacity and configuration. Another possible scenario is the storage of data in a persistent storage layer such as a database or a cloud storage [2]. The longer the data is stored in a system the higher the risk gets. The last option (storage location like a file or a database for instance) has the highest risks because the data can remain stored for a longer duration [2]. Some OCR applications, Microsoft Lens for example, create a file on a cloud storage to store the extracted text, which may lead to unauthorized access to confidential and sensitive information, unlawful repurposing, or an infringement of the data storage limitation principle.

The Cloud Service Providers (CSPs) have promised to ensure the data security over stored data of cloud clients by using methods such as firewalls and virtualization [3]. These mechanisms would not provide complete data protection because of their vulnerabilities' over the network and CSPs have full command of cloud applications, hardware, and client data [3]. Encrypting sensitive data before hosting can deserve data privacy and confidentiality against CSP; however, encryption schemes are usually impractical because of the huge amount of communication overheads over the cloud access patterns [3].

Cloud storage vulnerabilities can be classified into three categories: (i) network/resource-based[5], which includes Denial of Service, Buffer Overflow, Virtual Machine Based Rootkit, Side Channel, Man in the Middle, Replay Attack, and Byzantine Failure; (ii) browser-based, which includes Cross Site Scripting, Malware, and XML Wrapping; and (iii) social networking-based, which includes Sybil Attack, Social Intersection Attacks, and Collusion Attack [4]. All the previous vulnerabilities exist in cloud storage and the majority of their mitigations (if exist) are either reliable in specific scenarios, apply to a single scenario (unreliable), or can be defeated [4], which makes cloud storage unsafe for OCR applications to store sensitive information in.

The visibility and public awareness of safe information and transaction-sharing remain limited; the urgency and behavior of users do not reflect a positive reaction to awareness-related efforts [6]. Therefore, the proposed solution in this research takes the form of a technology tool.

Research goal

Based on the conducted research, OCR applications that store extracted text on cloud storage should not process images including sensitive information to achieve a reliable level of security, and for the applications not to process images including sensitive information, these images

must be identified first. This research aims to propose a solution to potential violations associated with OCR applications and thoroughly test the major part of it, which is identifying sensitive data. The proposed solution is to classify the images by a neural network[7] that could be integrated with OCR applications so that the neural network's page is the first page obligatorily navigated to the user.

In this solution, the neural network's main task is to accurately differentiate between images including sensitive and non-sensitive information, and be integrated with OCR applications to automatically prevent OCRs from processing and extracting text of sensitive data.

2. Background:

2.1.Optical character recognition (OCR)

Optical character recognition is a field with many different applications, including financial, legal, healthcare, and invoice imaging.

2.2. Network Vulnerability

A network vulnerability is a security exposure that results from a flaw in design or implementation. A network vulnerability has the potential to trigger an unanticipated and undesirable action compromising the security of a network infrastructure. Thus, a network vulnerability, even in cases where deployment and implementation are done correctly, makes it impossible to stop an unauthorized user from gaining access to a network, altering it, and compromising data on it. Often, and particularly in situations where the vulnerability is related to software, it is anticipated that the vendor would release patches to address any found vulnerabilities[5].

2.3.Browser-Based Vulnerability

A web browser-based attack uses a web browser's flaws to steal user information or run malicious code. By exploiting flaws in web applications, these attacks may compromise user data, steal confidential information, or interfere with services. The way the website is operated could be used by attackers to target its end users. Through pop-ups and clickjacking, attackers inject websites with malicious code [9], and the end-user systems forward sensitive data to attackers.

Insecure habits, malevolent activities, and technical weaknesses all contribute to the occurrence of browser-based attacks.

2.4.Social Networking-Based Vulnerability

Numerous security and privacy issues are related to the user's shared information particularly when the user uploads private files [10]. Shared information may be maliciously used by the attacker for illegal reasons. Targeting vulnerable sectors of the community increases the hazards [10]. Privacy violations through social media may take many forms including contamination or accessing a computer system, acquiring sensitive and confidential information like username, password, and credit card details of a user through fake websites and emails, stealing identity, and so on. [10].

2.5. Image classification

A key aspect of computer vision is image classification, which is classifying images into one of several predetermined labels. It serves as the foundation for advanced computer vision tasks like segmentation, detection, and localization. Handcrafted features were first obtained from images by feature descriptors, and these were used as input to a trainable classifier [11]. The main problem with this strategy was that the feature extraction stage's design had a significant impact on the classification task's accuracy, and this was often a demanding task. These difficulties have been solved in recent years by deep learning models that take advantage of numerous layers of nonlinear information processing for feature extraction and modification, pattern analysis, and classification [11].

2.6. Neural Networks

Neural networks are a type of artificial intelligence that attempts to imitate the way a human brain works. Neural networks work by creating connections between processing elements, the computer equivalent of neurons. The organization and weights of the connections determine the output [12]. Neural network models are ubiquitous in the image data space. They work well on computer vision tasks like image classification, object detection, and image recognition. [13]. They have hence been widely used in artificial intelligence modeling.

2.6.1. Convolutional Layers

A neural network's primary component is the convolutional layer. It includes

several filters, the parameters of which are to be “learned” during the training. The filter's size is less than that of the original image. The convolutional layers are used as feature extractors; thus, they learn the feature representations of their input images [11]. To create a new feature map[14], inputs are convolved with the learned weights, and the convolved outputs are then sent via a nonlinear activation function.

2.7.MobileNet

The foundation of MobileNets is a streamlined architecture that creates lightweight deep neural networks using depth-wise separable convolutions [15]. The standard convolution filters feature based on the convolutional kernels and combine them to produce a new representation. However, depth-wise separable convolutions split the filtering and combination steps into two steps, which results in a tangibly reduced computational cost for the MobileNet neural network [15]. MobileNets possess high adaptability towards the tendency of human interaction that puts the effectiveness of computer vision the most into consideration [16]. Also, it is known to achieve high accuracy scores with substantially fewer parameters than those in standard models. Due to their high efficiency, MobileNets are used in solving a variety of problems including face attribute classification, fine-grained recognition, large-scale geolocalization, object detection, face embedding, and more [15].

2.8.Feature map

A feature map is a set of neurons that are connected to an input and work to generate an output that encodes features or patterns present in the input. This allows the neurons to extract particular features from the input [14]. Multiple feature maps are usually stacked together to create a hierarchy of features [14].

3. Methodology

In order to differentiate between images including sensitive information and non-sensitive information at a high accuracy and low computational cost and with a relatively small amount of data[15], [16], the MobileNet neural network (which lies under the Supervised learning category[17]) was used. The MobileNet model is based on depthwise separable convolutions [18] which factorize a standard convolution into a depthwise convolution [19] and a 1×1 convolution called pointwise convolution[20] [15]. A standard convolution both filters and combines the images into a new set of outputs in one step, and then the depthwise separable convolution splits this into two layers [15].

Figure (2) shows how a standard convolution is factorized into a depthwise convolution (3) and a 1 × 1 pointwise convolution (4).

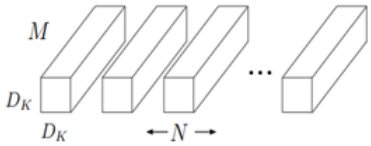

Fig (2) *Standard Convolution*[15]

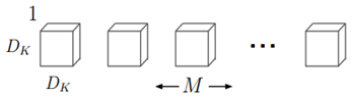

Fig (3) *Depthwise Convolutional Filters* [15]

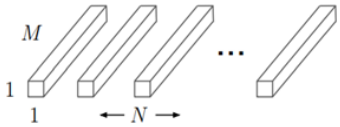

Fig (4) *1×1 Convolutional Filters called called Pointwise Convolution* [15]

Depthwise separable convolutions are included in MobileNet for their substantial reduction in computational cost, they also apply a single filter [21] per each input channel (input depth [22]). Then, a linear combination of the output of the depthwise layer is created by Pointwise convolution, a simple 1×1 convolution [15]. Depthwise convolution with one filter per input depth is computed as shown in the following equation:

$$\hat{G}_{k,l,m} = \sum_{i,j} \hat{K}_{i,j,m} \cdot F_{k+i-1,l+j-1,m} \quad [15]$$

where **K̂** is the depthwise convolutional kernel[23] of size **Dk** × **Dk** × **M** where the **m**th filter in **K̂** is applied to the **m**th channel in feature map **F** to produce the **m**th channel of the filtered output feature map **Ĝ**. In the training process of MobileNet, features of the raw training data (images of items including sensitive information and items that do not include sensitive information) were identified in a feature extraction process and then mapped to the intended labels (“sensitive information” and “non-sensitive information”) so that the model could recognize similar sets of features in the testing data and classify it accordingly. The MobileNet neural network was trained four times with a different amount of training data in each trial: In the first trial, the neural network was trained by 190 images labeled “Sensitive Information” and 190 labeled “Non-sensitive Information”. In the second trial, the neural network was trained by 290 images labeled “Sensitive Information” and 290 labeled “Non-sensitive Information”. In the third trial, the neural network was trained by 390 images labeled “Sensitive Information” and 390 labeled “Non-sensitive Information”. In the fourth trial, the neural network was trained by 490 images labeled “Sensitive Information” and 490 labeled “Non-sensitive Information”. The neural network did not show notable progress after the fourth trial, therefore no more training sessions were run. The data labeled “Sensitive information” included images of credit cards, debit cards, bank statements, income statements, transcripts, and login pages with credentials, while the “Non-sensitive Information” included images of everyday items, in addition to the images that the model could confuse with sensitive data (images including tables and numbers):

Calendars, school public schedules, and boarding schools and hotels meal plans. Both types, sensitive and non-sensitive, included images in different positions, lightings, and backgrounds. A web application interface was built using JavaScript and HTML to integrate the model with it and simulate real-life experience, especially since OCR applications that store text in cloud storage are web-based.

Testing the neural network's accuracy:
After each training session, the model was tested on 50 images including sensitive information, and 50 including non-sensitive information, and the number of images the model classified incorrectly in each trial was documented. Figures (5), (6), (7), (8) represent the model displaying its highest probability to some of the testing data.

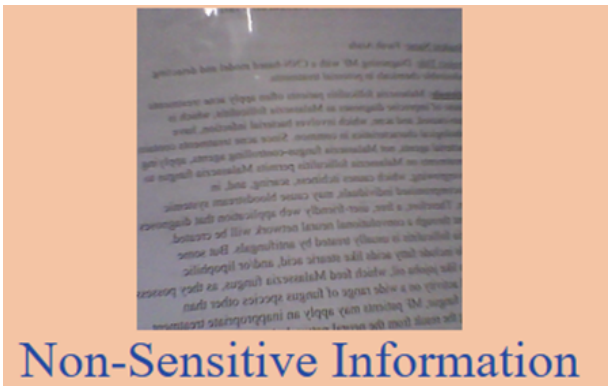

Fig (5) *Classifying an essay*

Fig (6) *Classifying a credit card*

Fig (7) *Classifying a baseball image*

Fig (8) *Classifying a calendar on a phone*

The time in which the model classified testing data was recorded by a stopwatch. The accuracy of identifying sensitive data and non-sensitive data (either printed, in

real life, or from a phone screen) in each trial was calculated by dividing the number of images the model classified correctly by 50 and then multiplying by one hundred as shown in the following equation: [(number of images classified correctly/50)*100].

4. Results:

table **(1)**

Throughout the testing process, the number of training images for both classes was manipulated four times. As demonstrated in the following table, the recognition speed and accuracy of the model directly varied according to the amount of training data to a certain extent. In each trial, the model was tested on 50 images containing sensitive data and 50 containing non-sensitive data. It is worth noting that the change in the time spent in recognition according to the amount of training data was relatively minuscule.

Trial	1	2	3	4
Amount of sensitive data	190	290	390	490
Amount of non-sensitive data	190	290	390	490
Time of identifying sensitive data	3 Sec ± 0.5	2 Sec ± 0.5	2 Sec ± 0.5	2 Sec ± 0.5
Time of identifying non-sensitive data	1 Sec ± 0.5	1 Sec ± 0.5	1 Sec ± 0.5	1 Sec ± 0.5
No. of sensitive data classified correctly/50	43	45	48	49
No. of non-sensitive data classified correctly/50	44	48	50	50
Accuracy of identifying sensitive data	86%	90%	96%	98%
Accuracy of identifying non-sensitive data	88%	96%	100%	100%

5. Discussion:

The results show that the accuracy of the model generally increased by increasing the number of training images, which is true with most prediction models[24] since increasing the training data lessens the likelihood of overfitting [25].

In the final trial, the model classified 50 out of 50 items including non-sensitive information as “Non-Sensitive Information” and none as “Sensitive Information”, classifying each item in approximately 1 second and, out of the 50 images including sensitive information, the model classified 48 as “Sensitive Information” and 2 as “Non-Sensitive Information” classifying each image in approximately 2 seconds, indicating that identifying sensitive information at high efficiency is applicable. Below are some reasons that could explain these observations.

First, the training data sets consisted of sensitive and non-sensitive data in various positions, lightings, backgrounds, and situations, which may have allowed the model to accurately classify most of the testing data.

Second, the model did not classify uploaded images, instead, it classified testing data live, which only displays a final prediction once the testing item is stable, whereas if the model were to classify captured images, the user may upload shaky or low-quality images, altering the dimensions of images' components and accordingly the accuracy.

6. Limitations:

The language in any text-including image, whether in the training or the testing data, is English; the same performance is not guaranteed with other languages.

The model was only trained on American and European formats of bank statements and income statements, which will likely result in a weaker performance with other formats.

Non-sensitive information data sets included images with cells and numbers, such as calendars, which the model could confuse with sensitive information (see 4.); nevertheless, there could be more confusing data to the model that was not considered.

The time the model spent to classify testing data could be recorded with a more advanced method to eliminate errors further.

If applied in real life, this model could only identify and help prevent sensitive information coming through OCR applications from being processed and stored on cloud storage; yet, a user can manually store an image of a debit card on the cloud. That is, the model will never be able to prevent all violations associated with cloud storage.

7. Conclusion:

In this paper, based on available information and previous attainments, I proposed a solution of classifying items into sensitive and non-sensitive information before scanning them with OCR applications that store extracted text on cloud storage for robust protection against privacy violations associated with the cloud. I also tested the main part of this proposed solution, which is differentiating between sensitive and non-sensitive information at a reliable level of accuracy, and kept records of

the outcomes of all trials until a high level of efficiency was reached. Sensitive information in the testing data was identified in approximately 2 seconds per item with an accuracy that reached 98% in the fourth trial, demonstrating the validity of the major part of the proposed solution. This indicates that, if integrated with the MobileNet trained in this study with simple coding, including if else statements [26], OCR applications should be able to identify and prevent sensitive information from being processed and stored on the cloud and accordingly offer a reliable level of security for OCR applications users.

8. Recommendations:

For future work, I suggest implementing a user-friendly format for this solution: printing a message that states that the user is trying to scan sensitive information and that they should try with another item whenever the user tries to scan an item including sensitive information or swiftly proceeding to processing and extracting text out of the item in case the user tries scanning an item that does not include sensitive information. It is also recommended to have strong approaches to the limitations mentioned: include training data in different languages, train the model on different formats of bank statements and income statements, include more non-sensitive information that the model could conflate with sensitive information, and record the time spent in classifying each testing data with a more advanced method.

9. References

[1] N. A. M. Isheawy and H. Hasan, “Optical Character Recognition (OCR) System,” *Opt. Character Recognit.*, vol. 17, no. 2, Apr. 2015.

[2] I. Barberá, “AI Possible Risks & Mitigations - Optical Character Recognition,” *Opt. Character Recognit.*, Sep. 2023.

[3] N. Vurukonda and B. T. Rao, “A Study on Data Storage Security Issues in Cloud Computing,” *Procedia Comput. Sci.*, vol. 92, pp. 128–135, 2016, doi: 10.1016/j.procs.2016.07.335.

[4] N. C. Rajasekar and C. O. Imafidon, “Exploitation of Vulnerabilities in Cloud-Storage,” *GSTF Int. J. Comput.*, vol. 1, no. 2, 2011, doi: 10.5176/2010-2283_1.2.41.

[5] O. Awodele, E. E. Onuiri, and S. O. Okolie, “Vulnerabilities in Network Infrastructures and Prevention/Containment Measures”.

[6] H. De Bruijn and M. Janssen, “Building Cybersecurity Awareness: The need for evidence-based framing strategies,” *Gov. Inf. Q.*, vol. 34, no. 1, pp. 1–7, Jan. 2017, doi: 10.1016/j.giq.2017.02.007.

[7] A. D. Dongare, R. R. Kharde, and A. D. Kachare, “Introduction to Artificial Neural Network,” vol. 2, no. 1, 2012.

[8] A. Singh, K. Bacchuwar, and A. Bhasin, “A Survey of OCR Applications,” *Int. J. Mach. Learn. Comput.*, pp. 314–318, 2012, doi: 10.7763/IJMLC.2012.V2.137.

[9] P. Shital and C. R., “Web Browser Security: Different Attacks Detection and Prevention Techniques,” *Int. J. Comput. Appl.*, vol. 170, no. 9, pp. 35–41, Jul. 2017, doi: 10.5120/ijca2017914938.

[10] A. K. Jain, S. R. Sahoo, and J. Kaubiyal, “Online social networks security and privacy: comprehensive review and analysis,” *Complex Intell. Syst.*, vol. 7, no. 5, pp. 2157–2177, Oct. 2021, doi: 10.1007/s40747-021-00409-7.

[11] W. Rawat and Z. Wang, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review,” *Neural Comput.*, vol. 29, no. 9, pp. 2352–2449, Sep. 2017, doi: 10.1162/neco_a_00990.

[12] M. Islam, G. Chen, and S. Jin, “An Overview of Neural Network,” *Am. J. Neural Netw. Appl.*, vol. 5, no. 1, p. 7, 2019, doi: 10.11648/j.ajnna.20190501.12.

[13] M. M. Srivastava and P. Kumar, “Machine Learning approaches to do size-based reasoning on Retail Shelf objects to classify product variants,” Oct. 07, 2021, *arXiv*: arXiv:2110.03783. Accessed: Aug. 15, 2024. [Online]. Available: http://arxiv.org/abs/2110.03783

[14] D. Huang, Q. Bu, Y. Qing, Y. Fu, and H. Cui, “Feature Map Testing for Deep Neural Networks,” Jul. 21, 2023, *arXiv*: arXiv:2307.11563. Accessed: Aug. 15, 2024. [Online]. Available: http://arxiv.org/abs/2307.11563

[15] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” 2017, *arXiv*. doi: 10.48550/ARXIV.1704.04861.

[16] B. Khasoggi, E. Ermatita, and S. Samsuryadi, “Efficient mobilenet architecture as image recognition on mobile and embedded devices,” *Indones. J.*

Electr. Eng. Comput. Sci., vol. 16, no. 1, p. 389, Oct. 2019, doi: 10.11591/ijeecs.v16.i1.pp389-394.

[17] V. Nasteski, "An overview of the supervised machine learning methods," *HORIZONS.B*, vol. 4, pp. 51–62, Dec. 2017, doi: 10.20544/HORIZONS.B.04.1.17.P05.

[18] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2016, *arXiv*. doi: 10.48550/ARXIV.1610.02357.

[19] Y. Guo, Y. Li, L. Wang, and T. Rosing, "Depthwise Convolution Is All You Need for Learning Multiple Visual Domains," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 01, pp. 8368–8375, Jul. 2019, doi: 10.1609/aaai.v33i01.33018368.

[20] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise Convolutional Neural Networks," 2017, *arXiv*. doi: 10.48550/ARXIV.1712.05245.

[21] K. N. Plataniotis, D. Androutsos, and A. N. Venetsanopoulos, "Multichannel filters for image processing," *Signal Process. Image Commun.*, vol. 9, no. 2, pp. 143–158, Jan. 1997, doi: 10.1016/S0923-5965(96)00021-5.

[22] T. Van Dijk and G. De Croon, "How Do Neural Networks See Depth in Single Images?," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South): IEEE, Oct. 2019, pp. 2183–2191. doi: 10.1109/ICCV.2019.00227.

[23] C. Campbell, "Chapter 7 An Introduction to Kernel Methods".

[24] A. R. Ajiboye, R. Abdullah-Arshah, H. Qin, and H. Isah-Kebbe, "EVALUATING THE EFFECT OF DATASET SIZE ON PREDICTIVE MODEL USING SUPERVISED LEARNING TECHNIQUE," *Int. J. Comput. Syst. Softw. Eng.*, vol. 1, no. 1, pp. 75–84, Feb. 2015, doi: 10.15282/ijsecs.1.2015.6.0006.

[25] X. Ying, "An Overview of Overfitting and its Solutions," *J. Phys. Conf. Ser.*, vol. 1168, p. 022022, Feb. 2019, doi: 10.1088/1742-6596/1168/2/022022.

[26] S. Nurollahian, M. Hooper, A. Salazar, and E. Wiese, "Use of an Anti-Pattern in CS2: Sequential if Statements with Exclusive Conditions," in *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, Toronto ON Canada: ACM, Mar. 2023, pp. 542–548. doi: 10.1145/3545945.3569744.