PDF Malware Detection Using Visualization and Machine Learning

Ching-Yuan Liu, Min-Yi Chiu, Qi-Xian Huang, Hung-Min Sun

PDF Malware Detection Using Visualization and Machine Learning

Ching-Yuan Liu[1], Min-Yi Chiu[2], Qi-Xian Huang[3], and Hung-Min Sun[4⋆]

Department of Computer Science, National Tsing Hua University, Taiwan
joeyliu7878@gapp.nthu.edu.tw
Institute of Information Systems and Applications, National Tsing Hua University, Taiwan
s106065803@m106.nthu.edu.tw
Institute of Information Systems and Applications, National Tsing Hua University, Taiwan
xiangg800906three@gapp.nthu.edu.tw
Department of Computer Science, National Tsing Hua University, Taiwan
hmsun@cs.nthu.edu.tw

Abstract. Recently, as more and more disasters caused by malware have been reported worldwide, people started to pay more attention to malware detection to prevent malicious attacks in advance. According to the diversity of the software platforms that people use, the malware also varies pretty much, for example: Xcode Ghost on iOS apps, FakePlayer on Android apps, and WannaCrypt on PC. Moreover, most of the time people ignore the potential security threats around us while surfing the internet, processing files or even reading email. The Portable Document Format (PDF) file, one of the most commonly used file types in the world, can be used to store texts, images, multimedia contents, and even scripts. However, with the increasing popularity and demands of PDF files, only a small fraction of people know how easy it could be to conceal malware in normal PDF files. In this paper, we propose a novel technique combining Malware Visualization and Image Classification to detect PDF files and identify which ones might be malicious. By extracting data from PDF files and traversing each object within, we can obtain the holistic tree-like structure of PDF files. Furthermore, according to the signature of the objects in the files, we assign different colors obtained from SimHash to generate RGB images. Lastly, our proposed model trained by the VGG19 with CNN architecture achieved up to 0.973 accuracy and 0.975 F1-score to distinguish malicious PDF files, which is viable for personal, or enterprise-wide use and easy to implement.

Keywords: Malware Detection · PDF Malware · Malware Visualization · Machine Learning.

⋆ Corresponding Author

1 Introduction

The goal of malware detection is to identify whether a file is malicious or belongs to a certain malware family. Though various malware detection techniques have been proposed, they can be divided into two categories: static analysis and dynamic analysis. Since static analysis focuses on the content or the signatures of a certain malware, it has the advantage of being more time-efficient while at the cost of inaccuracy. Dynamic analysis observes and records a malware's behavior, which makes it more accurate but usually is time-consuming. Researchers have applied static and dynamic analysis techniques to examine common file formats in several platforms, such as executable files on x64 machines or .apk files on Android phones. In this paper, we only focus on Portable Document Format (PDF), which is supported by most platforms.

PDF, released in 1987 by Adobe, is currently one of the most used file formats worldwide. The main reason is that PDF files can be recognized and human-readable in most settings. Moreover, they can contain texts as well as various types of information, such as images, audios, scripts, or other files. Some PDF files even have forms or buttons inside, making them more interactive and richful. Due to their interactiveness and richness, some PDF files have been exploited by hackers to conduct malicious behaviors such as executing embedded javascript while opening the document, moving the mouse cursor to download and launch the malware to steal sensitive information of users, or encrypt certain types files in the machine [1,2]. Evidence shows that emails with malicious PDF attachments have caused severe damages to businesses and governments [3]. Therefore, the techniques of how to detect malicious PDF files in advance before opening them have been proposed and gained more and more attention nowadays.

Our proposed method is a novel technique that employs the image classification for PDF malware detection. A PDF file usually consists of multiple objects, which can be analogous to nodes in a huge tree structure that each node has a parent node and multiple children nodes. Taking advantage of this characteristic of PDF, we trace the tree-like structure to extract all objects in PDF files by a specific order. After applying the malware visualization technique, we build a dataset containing images of PDF files. With those images obtained from PDF files at hand, we can then train our malware detection model to distinguish the malicious PDF files from the benign ones.

The rest of the paper is organized as follows. In section Sect.2 we will briefly describe the structure and the characteristics of PDF files. Sect.3 will be discussing the related works of malware detection on PDF files. More detail of our experiment and proposed technique will be discussed in Sect.4. We will present our result in Sect.5. And finally, our conclusions will be narrated in Sect.6.

2 Background

PDF files are normally divided into four sections: Header, Body, 'xref' Table, and Trailer as shown in Fig.1.

Fig. 1: The main structure of PDF.

The ***header*** specifies the PDF version of the file and some hidden characters to inform software to recognize the file as the format of PDF. The ***body***, as the biggest part of a PDF file, contains all the main contents of the file including all the objects. Each object may contain multiple tags, and indirect objects to more detailedly describe the corresponding object. For several files with a special purpose, objects may also attach streams with information of text, images, or even scripts. The main purpose of the ***'xref' table*** is to store the relative address of objects in each section in order to optimize the efficiency of object finding. Lastly, the ***trailer***, which is the starting point to read a PDF file, instructs the object number of the root object, the size of the PDF file, and some additional information including the most important part about the address of 'xref' table.

Because of the flexibility PDF provides, attackers utilize a variety of methods to sneak any kind of malicious scripts or make the target files obfuscate to evade malware detection tools. Such as dividing a file into multiple sections to create multiple 'xref' tables, deleting size or root object's information in trailer, or utilize one or more encoding methods to encrypt scripts inside an object. These alterations will not damage the readability of a PDF file, but make it vulnerable for attackers to steal sensitive information.

To reduce the detection error caused by an attacker's obfuscation, our method focuses on visualizing PDF files to seek the relative pattern of malicious files despite them having been obfuscated.

3 Related Work

More and more researchers utilize machine learning in the field of malware detection. However, only a few specialists take advantage of image and object detection, which results in better efficiency and accuracy.

O'Shaughnessy et al. [4] utilized byte plot technique to fill the bytes of executable files into RGB images with three kinds of Space-Filling Curve patterns, including Z-order, Gray-code, and Hilbert curves. They extracted image features with Local Binary Patterns (LBP), Gabor filters, and Histogram of Gradients (HOG) and trained them with K-Nearest Neighbor (KNN), Random Forest

(RF), and Decision Trees (DT) models. After all, the results of the proposed method were compared with GIST Byte Plot Method [5].

FU et al. [6] focused on analyzing files of PE format. After section division, which divided a file into code section, data section, and other natural sections, they calculated the entropy value, byte value, and relative section size of the sections independently. Besides image malware visualization, they also recorded the ASCII strings that appeared in the file. With the image features and the string features, they trained their method with multiple classifiers in order to achieve high accuracy and performance.

Bhodia et al. [7] transferred binary files into grayscale images directly. After feature extraction they trained the dataset with a simple KNN classifier. Darus et al. [8] also transformed files into grayscale images, but they focused on .apk files for Android applications.

Although Kapoor et al. [9] did not do malware visualization, they extracted the control flow of malwares while dividing codes into multiple basic blocks. By traversing the basic code blocks, extracting features and opcode sequence, they achieve high accuracy with multiple machine learning classifiers. Han et al. [10] also focused on analyzing the control flow of malwares. But after parsing, extracting features, and hashing with specific hash functions, they filled in colors for each pixel accordingly and created a unique RGB image for every malware sample. After the process, they successfully differentiated different malware families in a vast malware dataset.

While researchers mentioned above focused on a wide variety of file formats, such as PE files, normal executable files, mobile applications, etc., few of them noticed the dangers of PDF files [11,12,13].

Filiol et al. [14] and Maiorca et al. [15] did detailed investigation of multiple malicious PDF files detection techniques proposed in recent years. According to the investigation, we can acknowledge that even fewer researchers combine malware visualization with PDF malware detection.

The most inspiring research was done by Corum et al. [16]. Through byte plot and Markov plot they directly transferred raw PDF files into grayscale images and utilized SIFT and ORB to extract keypoints of the images as the main features of the files. Besides keypoint features, they also extract texture features of images including LBP, local entropy, and Gabor filter. By comparing multiple combinations of keypoint features and texture features, their method results in an accurate model for object detection. While directly extracting features from grayscale images without examining the structure or even contents in PDF files, their method lacks a reasonable explanation.

Through the proposed methods mentioned above, we can easily notice that there are multiple malware visualization techniques done to all kinds of file formats, but fewer of them focused on PDF files. With all the inspiration from those researches, this paper contributed a novel malware visualization technique combined with machine learning which is tailored for PDF files.

4 Methodology

4.1 Overview

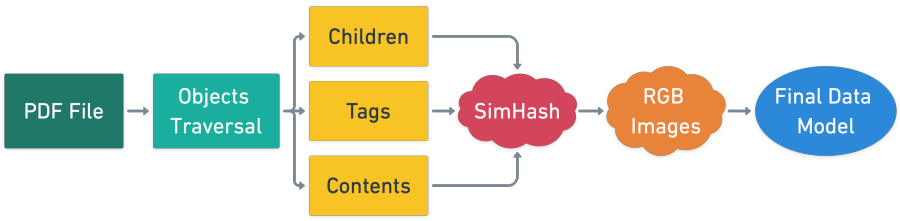

Fig. 2: An overview of the proposed method.

In our observation, an increasing amount of PDFs containing various mutations of malicious scripts has been exploited by hackers nowadays to lure the victims to open or browse the file, which usually comes as an attachment of email. Those malicious PDFs contain similar patterns of scripts, structure of tags and contents in objects, and tree-like composition order.

Our proposed method consisting of several processing steps is shown in Fig.2. First, we traverse the whole PDF file by its structure defined in the PDF specification. Next, we extract three main features including children, tags, and contents. In step three, we apply the SimHash algorithm in order to compare the similarity between image results. After hashing, we assign different colors according to the hash value of every object and generate an RGB image for each PDF file. Finally, we train our malware detection model with the images obtained from the previous step. Details will be described in the following sections.

4.2 PDF objects traversal

According to the PDF structure mentioned in section II, a standard PDF structure can be divided into four parts: header, body, 'xref' table and trailer [17]. The correct way to parse a PDF file should start from the trailer to the header.

From the trailer, we can obtain the detailed information of the file including which object is the root, how many objects are there in the file, and the relationships among the objects as shown in Fig.3. Therefore, attackers will not let such information be easily discovered, they will try to obfuscate or even remove the information which in fact will not affect the readability of the file. Apart from the trailer, we can also get important information by reading the two numbers in the first line of the 'xref' table as shown in Fig.4, which indicate the first object and the size of the section, respectively. Due to the flexibility characteristic of a PDF file, there might be multiple 'xref' tables and trailers, which in turn results in multiple root objects for different sections.

```
trailer
<< /Root 1 0 R /Size 10 >>
startxref
7989
%EOF
```

Fig. 3: An example of trailer.

```
xref
0 10
0000000000 65535 f
0000000017 00000 n
0000000109 00000 n
0000000169 00000 n
0000000267 00000 n
0000000328 00000 n
0000000686 00000 n
0000000789 00000 n
0000006784 00000 n
0000006886 00000 n
```

Fig. 4: Information of the root object and total objects in the section can be observed from the first line of 'xref' Table.

After identifying the root object, we start the process of object traversal. During the traversal, we extract the three main features from every object as shown in Fig.5.

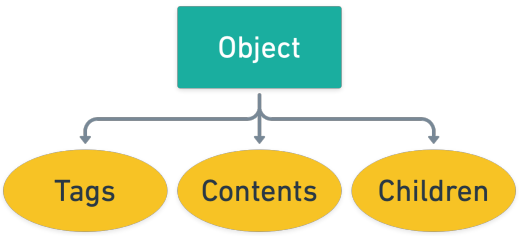

Fig. 5: The three main features we extract from each object.

According to the PDF structure, under an object there might be multiple data types that start with "/" to describe it. We treat them as the tags of an object. Next, some tags can be followed by other objects called indirect objects that contain more information about the tag. Those indirect objects are treated as the object's children as shown in Fig.6.

Finally, many malicious PDF files can contain scripts which will be triggered when opening the files, moving the mouse cursor on the document, or clicking any (in)visible button in the files. In order to take the plain instead of encrypted contents as features, streams with suspicious properties such as "JS",

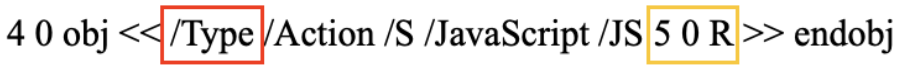

Fig. 6: Sample of tags (red rectangle) and children (yellow rectangle) relatively.

"JavaScript", "MacroForm" and "XFA" are decompressed and saved as objects' contents.

By doing depth first search (DFS) on all children objects, we can simulate the tree-like structure of the PDF file. Afterwards, tags and contents of every object in a PDF file will be extracted by the order of the tree-like structure of the PDF file. The complete procedure is shown in Algorithm 1.

Algorithm 1 PDF objects traversal

```
1: Obtain root objects through trailer and 'xref' table
2: for all root objects do
3:     if has contents with malicious tags then
4:         do flatedecode to decompress contents
5:     Put children in traversing queue
6:     Do DFS to traverse all the objects
```

4.3 Malware Visualization

After the preprocessing described above, a dataset containing features such as tags and contents of every object in a single PDF file is created. To find out the similarity of features among PDF files, tags and contents are combined into one string, which is computed by SimHash and stored as strings in binary format.

SimHash is a local sensitive function and is adopted by Google to distinguish duplicate web pages. The main characteristic of SimHash is that though the original texts are similar, the output after hashing will not be completely different like other hash functions.

Through SimHash, we can obtain a number which can be translated into binary representation. To prove that SimHash suits our experiment, we can know that the hamming distance is related to the similarity between two texts by observing the binary values. Afterwards, the binary values will be divided into three sections with equivalent length in order to retrieve the relative values of RGB pixels to form a color image as shown in Fig.7.

To create an image formed by multiple RGB pixel blocks, two parameters are employed: color, which is given by the output value of SimHash; size, the height of an image block. To obtain the size, besides tags' length, we also have to retrieve the correct length of contents. To avoid some extremely lengthy scripts with unused padding words that attackers intend to evade malware detection, we

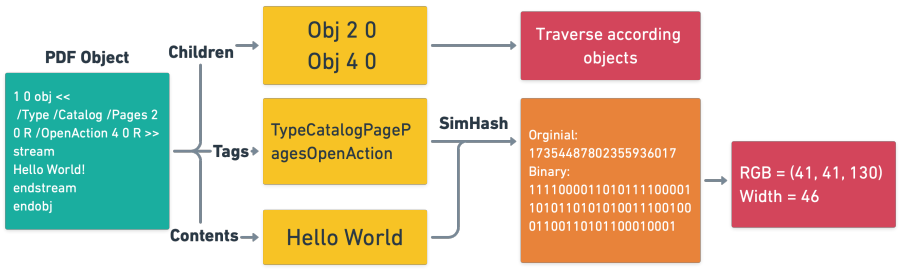

Fig. 7: Detailed example of feature extraction.

reduce the contents' length by a specific ratio which we obtained by averaging all contents' length in our PDF dataset. Then the tags' and contents' length of a single object are accumulated and are made to be related to the height of a color block.

By iterating the steps shown in Algorithm 2, an RGB image can be formed with multiple color blocks with specific order representing different objects and the relative order in a PDF file. Lastly, for better classification, images are scaled to "200 x 150" as shown in Fig.8.

Algorithm 2 Malware Visualization

```
for every traversed object do
    Do SimHash with Tags and Contents
    Acquire output from SimHash and transform into binary
    Divide binary string into 3 equal parts to get relative RGB color
    Reduce content length with specific ratio
    Accumulate Tags Length and Content Length to get the height of color block
Draw images with all the RGB color and Height
Scale images into 200 x 150
```

5 Experimental Result

To prove whether our result matches the assumption that currently most malicious PDFs are generated by certain manner. We utilize Structural Similarity Index Measure (SSIM), which is good at comparing the similarity by recognizing the whole structure of images.

First, we randomly chose a malicious image and a benign image. Then, we compare the chosen images with both malicious and benign datasets independently. Finally, the average SSIM value between each dataset is acquired as shown in Fig.9. From the result, the similarity between malicious PDFs can be

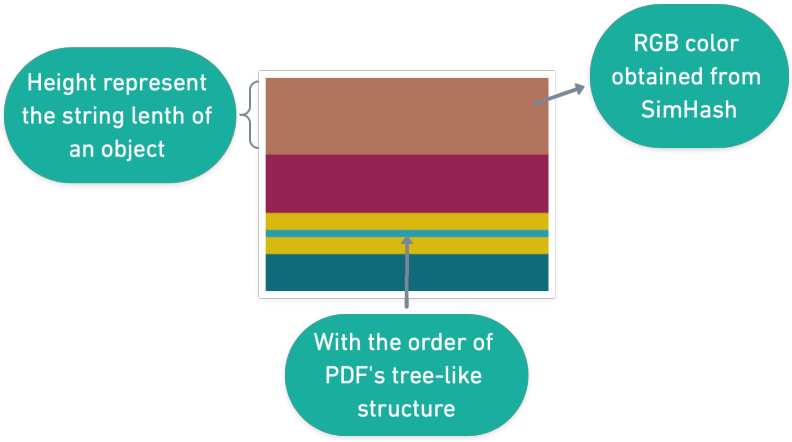

Fig. 8: Sample of the resulting image.

acknowledged, while benign images seem to have more diversity related to malicious images. We can tell that malicious PDFs are much more similar than benign images, which implies our assumption and method are reasonable. The failure of the malware detection may be caused by the certain amount of similarities among the malicious and benign images.

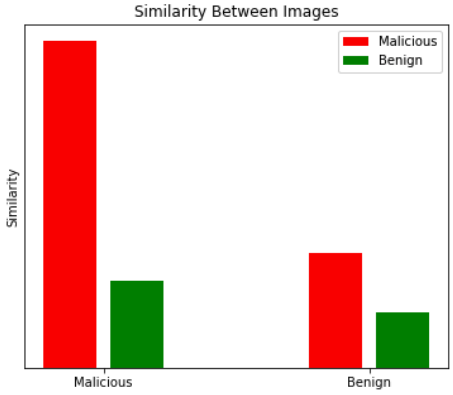

Fig. 9: SSIM comparison between malicious and benign images.

Our datasets consisting of 9000 malicious and benign PDF files each are from Contagio, which is a collection of malware samples and benign samples for comparison. After the malware visualization process described above, two image

datasets transferred from malicious and benign PDF files are created. To express our method concretely, we compare images from malicious PDF files which belong to the same CVE as shown in Fig.10. Due to the limitation of the datasets, most of the malwares are not tagged with CVE numbers while the rest that tagged with CVE are outdated. However, we can still recognize that there exist obvious differences in patterns among malicious images.

Fig. 10: Patterns between malicious PDF files from the same CVE can be easily observed.

To further examine the proposed method, both the image datasets, which contain more than 9000 files of benign and malicious PDF files each, are further divided into training and validation sets with the ratio of 60% and 40% respectively. In order to prove the feasibility of our proposed method, the images will be trained with a built-in model of Tensorflow library, which is the VGG19 model.

The VGG19 model [18], which was proposed by Visual Geometry Group from Oxford University, is specialized in image classification. With multiple smaller convolution layers instead of larger ones, VGG19 results in higher non-linearity and fewer parameters required, making it more accurate and more efficient.

With the characteristics that VGG19 offered, we utilize it to prove our method further. The performance of our method is measured by using two primary metrics: accuracy and F1-score.

The accuracy can be easily observed through the confusion matrix shown as Fig.11. To sum up, our proposed method results in high accuracy of 0.973 and relative F1-score of 0.975.

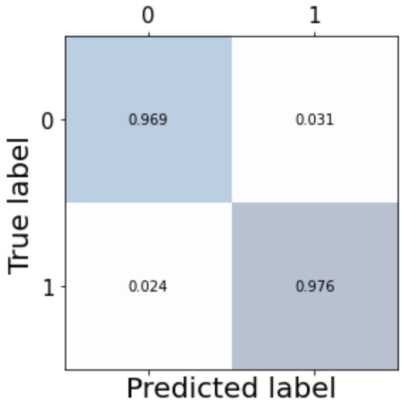

Fig. 11: Confusion matrix of the experimental result.

6 Conclusion

With portability, flexibility, and stability, Portable Document Format (PDF) becomes one of the most used file formats worldwide. People utilize PDF files to store documents, submit resumes or even transport sensitive information. The advantages and popularity also turn PDF files into huge targets for malicious attackers. Our proposed method is based on the idea of maximizing the utilization of PDF format's characteristics. First of all, after objects traversing and features extraction, we applied SimHash to retrieve relative RGB color values. When finishing the preprocessing to every PDF file, image datasets of benign and malicious PDF files are created. At last, the datasets are trained by the VGG19 model to achieve 0.973 accuracy and 0.975 F1-score. Our results show that the proposed method received relatively excellent performance. It indicates that there are some patterns for sure to recognize benign or malicious PDF files by turning them into images with reasonable methods.

For further research, besides optimizing our proposed method, distinguishing different malware families between a huge dataset of malwares will also be our main focus.

References

1. SentinelOne: Malicious PDFs - Revealing the Techniques Behind the Attacks, https://www.sentinelone.com/blog/malicious-pdfs-revealing-techniques-behind-attacks/. Last accessed 27 Mar 2019
2. Cybersecurity Insiders: Cyber Attack with Ransomware hidden inside PDF Documents, https://www.cybersecurity-insiders.com/cyber-attack-with-ransomware-hidden-inside-pdf-documents/.

3. Kaspersky: Top 4 dangerous file attachments, https://www.kaspersky.com/blog/top4-dangerous-attachments-2019/27147/. Last accessed 31 May 2019
4. O'Shaughnessy, S. (2019, October). Image-based Malware Classification: A Space Filling Curve Approach. In 2019 IEEE Symposium on Visualization for Cyber Security (VizSec) (pp. 1-10). IEEE. https://doi.org/10.1109/VizSec48167.2019.9161583
5. Nataraj, L., Karthikeyan, S., Jacob, G., & Manjunath, B. S. (2011, July). Malware images: visualization and automatic classification. In Proceedings of the 8th international symposium on visualization for cyber security (pp. 1-7). https://doi.org/10.1145/2016904.2016908
6. Fu, J., Xue, J., Wang, Y., Liu, Z., & Shan, C. (2018). Malware visualization for fine-grained classification. IEEE Access, 6, 14510-14523. https://doi.org/10.1109/ACCESS.2018.2805301
7. Bhodia, N., Prajapati, P., Di Troia, F., & Stamp, M. (2019). Transfer learning for image-based malware classification. arXiv preprint arXiv:1903.11551.
8. Darus, F. M., Ahmad, N. A., & Ariffin, A. F. M. (2019, November). Android Malware Classification Using XGBoost On Data Image Pattern. In 2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS) (pp. 118-122). IEEE. https://doi.org/10.1109/IoTaIS47347.2019.8980412
9. Kapoor, A., & Dhavale, S. (2016). Control Flow Graph Based Multiclass Malware Detection Using Bi-normal Separation. Defence Science Journal, 66(2). https://doi.org/10.14429/dsj.66.9701
10. Han, K., Kang, B., & Im, E. G. (2014). Malware analysis using visualized image matrices. The Scientific World Journal, 2014. https://doi.org/10.1155/2014/132713
11. Laskov, P., & Šrndić, N. (2011, December). Static detection of malicious JavaScript-bearing PDF documents. In Proceedings of the 27th annual computer security applications conference (pp. 373-382). https://doi.org/10.1145/2076732.2076785
12. Maiorca, D., Ariu, D., Corona, I., & Giacinto, G. (2015, February). A structural and content-based approach for a precise and robust detection of malicious PDF files. In 2015 international conference on information systems security and privacy (icissp) (pp. 27-36). IEEE.
13. Smutz, C., & Stavrou, A. (2012, December). Malicious PDF detection using metadata and structural features. In Proceedings of the 28th annual computer security applications conference (pp. 239-248). https://doi.org/10.1145/2420950.2420987
14. Blonce, A., Filiol, E., & Frayssignes, L. (2008, March). Portable document format (pdf) security analysis and malware threats. In Presentations of Europe BlackHat 2008 Conference.
15. Maiorca, Davide, Biggio, Battista.: (2017). Digital Investigation of PDF Files: Unveiling Traces of Embedded Malware. In: IEEE Security and Privacy. 17. https://doi.org/10.1109/MSEC.2018.2875879
16. Corum, A., Jenkins, D., & Zheng, J. (2019, June). Robust PDF malware detection with image visualization and processing techniques. In 2019 2nd International Conference on Data Intelligence and Security (ICDIS) (pp. 108-114). IEEE. https://doi.org/10.1109/ICDIS.2019.00024
17. Whitington, J. (2011). PDF Explained: The ISO Standard for Document Exchange (1st ed.). O'Reilly Media.
18. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. https://doi.org/abs/1409.1556