References

[1] Prasad, Ramjee, and Vandana Rohokale. "Cyber Threats and Attack Overview." In Cyber Security: The Lifeline of Information and Communication Technology, pp. 15-31. Springer, Cham, 2020. ↑1

[2] Lewis, J. A. (2002). Assessing the risks of cyber terrorism, cyber war and other cyber threats (p. 12). Washington, DC: Center for Strategic & International Studies. ↑1

[3] Vyawahare, M., & Chatterjee, M. (2020). Survey on Detection and Prediction Techniques of Drive-by Download Attack in OSN. In Advanced Computing Technologies and Applications (pp. 453-463). Springer, Singapore. ↑2

[4] Categories of Cyber Security. Available at: https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security. ↑3

[5] Sitelock Security Report. Available at: https://www.sitelock.com/download/SiteLock-Website-Security-Insider-Q4-2017.pdf. ↑4 ↑15

[6] Microsoft Edge. Available at: https://www.microsoft.com/en-us/edge. ↑4

[7] Mozilla Firefox. Available at: https://www.mozilla.org/. ↑4

[8] Google Chrome. Available at: https://www.google.com/chrome/. ↑4

[9] Denko, Blaž & Pecnik, Spela & Fister jr, Iztok. (2021). A Comprehensive Comparison of Hybrid Mobile Application Development Frameworks. International Journal of Security and Privacy in Pervasive Computing. 13. 78-90. 10.4018/IJSPPC.2021010105. ↑4

[10] Article on 'App Development Decisions: Native, Web or Hybrid?' Available at: https://medium.com/@Imaginovation/app-development-decisions-native-web-or-hybrid-31c103f9b4e1. ↑4 ↑5

[11] Singhal, Mohit. "Analysis and Categorization of Drive-by Download Malware Using Sandboxing and Yara Ruleset." PhD diss., 2019. ↑4

[12] Symantec Corporation, "Internet Security Threat Report 2020," Symantec, 2020. Available: http://www.symantec.com. ↑4 ↑16

[13] Sihwail, Rami, Khairuddin Omar, and Khairul Akram Zainol Ariffin. "A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis." International Journal on Advanced Science, Engineering and Information Technology 8, no. 4-2 (2018): 1662. ↑4 ↑8

[14] Tahir, Rabia. "A study on malware and malware detection techniques." International Journal of Education and Management Engineering 8, no. 2 (2018): 20. ↑4

[15] Thanh, Cong Truong, and Ivan Zelinka. "A survey on artificial intelligence in malware as next-generation threats." In Mendel, vol. 25, no. 2, pp. 27-34. 2019. ↑4

[16] Market Share of Mobile Platforms globally. Available at: https://gs.statcounter.com/os-market-share/mobile/worldwide. ↑5

[17] Article on 'Android Devices Worldwide'. Available at: https://www.gizchina.com/2021/05/19/there-are-over-3-billion-active-android-devices-worldwide/. ↑5

[18] Android Versions Market Share. Available at: https://gs.statcounter.com/android-version-market-share/mobile-tablet/worldwide. ↑5

[19] Android WebView. Available at: https://developer.android.com/guide/webapps/webview. ↑5 ↑9 ↑49

[20] Acemoglu, Daron, and Pascual Restrepo. 8. Artificial Intelligence, Automation, and Work. University of Chicago Press, 2019. ↑5

[21] Haenlein, Michael, and Andreas Kaplan. "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence." California management review 61, no. 4 (2019): 5-14. ↑5 ↑5

[22] Bory, Paolo. "Deep new: The shifting narratives of artificial intelligence from Deep Blue to AlphaGo." Convergence 25, no. 4 (2019): 627-642. ↑5

[23] Sabouret, Nicolas, and Lizete De Assis. "Understanding Artificial Intelligence." (2020). ↑6

[24] Jackson, Philip C. Introduction to artificial intelligence. Courier Dover Publications, 2019. ↑7

[25] Web Hacking Statistics 2021. Available at: https://patchstack.com/website-hacking-statistics/. ↑8

[26] J.J. Rosen, 2014. The Internet that you can't Google'. Available at: https://www.tennessean.com/story/money/tech/2014/05/02/jj-rosen-popular-search-engines-skim-surface/8636081/. ↑8

[27] Mirea, Mihnea, Victoria Wang, and Jeyong Jung. "The not so dark side of the darknet: a qualitative study." Security Journal 32, no. 2 (2019): 102-118. ↑8

[28] Pure, Ivo. "An automated methodology for validating web related cyber threat intelligence by implementing a honeyclient." ↑8

[29] B. Altay, T. Dokeroglu, and A. Cosar, "Context-sensitive and keyword density-based supervised machine learning techniques for malicious webpage detection," Soft Computing, pp. 1–15, 2018. ↑8

[30] McMahan, Brendan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. "Communication-efficient learning of deep networks from decentralized data." In Artificial Intelligence and Statistics, pp. 1273-1282. PMLR, 2017. ↑9 ↑131 ↑137

[31] Hard, Andrew, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. "Federated learning for mobile keyboard prediction." arXiv preprint arXiv:1811.03604 (2018). ↑9 ↑133

[32] Li, Wenqi, Fausto Milletarì, Daguang Xu, Nicola Rieke, Jonny Hancox, Wentao Zhu, Maximilian Baust et al. "Privacy-preserving federated brain tumor segmentation." In International Workshop on Machine Learning in Medical Imaging, pp. 133-141. Springer, Cham, 2019. ↑9 ↑133 ↑135

[33] Rieke, Nicola, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R. Roth, Shadi Albarqouni, Spyridon Bakas, et al. "The future of digital health with federated learning." NPJ digital medicine 3, no. 1 (2020): 1-7. ↑9

[34] Long, Guodong, Yue Tan, Jing Jiang, and Chengqi Zhang. "Federated Learning for Open Banking." In Federated Learning, pp. 240-254. Springer, Cham, 2020. ↑9

[35] Kohonen, Teuvo. "The self-organizing map." Proceedings of the IEEE 78, no. 9 (1990): 1464-1480.↑10

[36] B Wainakh, Aidmar, Alejandro Sanchez Guinea, Tim Grube, and Max Mühlhäuser. "Enhancing privacy via hierarchical federated learning." In 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 344-347. IEEE, 2020. ↑11

[37] J.J. Rosen, 2014. The Internet that you can't Google'. Available at: https://www.tennessean.com/story/money/tech/2014/05/02/jj-rosen-popular-search-engines-skim-surface/8636081/. ↑15

[38] G. K. Jayasinghe, J. S. Culpepper, and P. Bertok, "Efficient and effective realtime prediction of drive-by download attacks," 2014. ↑16

[39] Y. Cao, X. Pan, Y. Chen, and J. Zhuge, "JShield: Towards Real-time and Vulnerability-based Detection of Polluted Drive-by Download Attacks," in Proceedings of the 30th Annual Computer Security Applications Conference., 2014, pp. 466–475. ↑16 ↑16

[40] S. Sarwade and P. D. D. Patil, "Document-based and URL-based Features for Automatic Classification of Cross-Site Scripting in Web Pages," vol. 3, pp. 1–10, 2013. ↑17 ↑18

[41] L. Invernizzi, S. Benvenuti, M. Cova, C. Kruegel, and G. Vigna, "EVILSEED : A Guided Approach to Finding Malicious Web Pages," in IEEE Symposium on Security & Privacy (SP), 2012, pp. 428–442. ↑18 ↑21 ↑23 ↑31

[42] D. Canali, M. Cova, G. Vigna, C. Kruegel, "Prophiler: a fast filter for the large-scale detection of malicious web pages, "2011. ↑19 ↑21 ↑75 ↑76

[43] P. S. Rohit, R. Krishnaveni, "Deep Malicious Website Detection," vol. 2, Issue 4, pg. 517-522, 2013. ↑19

[44] N. Provos, P. Mavrommatis, M. A. Rajab, and F. Monrose, "All Your iFrames Point to Us," in USENIX Security Symposium, 2008. ↑19

[45] Y.-T. Hou, Y. Chang, T. Chen, C.-S. Laih, and C.-M. Chen, "Malicious web content detection by machine learning," Expert Systems with Applications, vol. 37, no. 1, pp. 55–60, Jan. 2010. ↑21

[46] Pham, Kien, Aècio Santos, and Juliana Freire. "Understanding Website Behavior based on User Agent." Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. ACM, 2016. ↑21

[47] P. Likarish and E. Jung, "A targeted web crawling for building malicious javascript collection," in Proceeding of the ACM DSMM, 2009, vol. 21, no. 4, pp. 23–26. ↑21

[48] H. Y. I Jo, E Jung, "Interactive Website Filter for Safe Web Browsing," Journal of Information Science, vol. 131, pp. 115–131, 2013. ↑21

[49] M. T. Qassrawi and H. Zhang, "Detecting Malicious Web Servers with Honeyclients," vol. 6, no. 1, pp. 145–152, 2011. ↑21

[50] T. H. and F. C. Ikinci, Ali, "Monkey-Spider : Detecting Malicious Websites with Low-Interaction Honeyclients," vol. 8, 2008. ↑21 ↑75 ↑76

[51] Malware Domain List. Available at: http://www.malwaredomainlist.com/. ↑23

[52] JSoup- JSoup Java Library. Available: http://www.jsoup.org. ↑23 ↑28 ↑168

[53] "HTML Unit". Available: http://htmlunit.sourceforge.net/. ↑24 ↑24 ↑27 ↑28 ↑59 ↑168

[54] "Rhino-Mozilla". Available:https://developer.mozilla.org/docs/Mozilla/Projects/Rhino. ↑24 ↑24 ↑26 ↑26 ↑28 ↑59 ↑169

[55] N. Z. Univeristy of Waikato, "WEKA.". Available: http:/ml/weka. ↑24 ↑28 ↑60 ↑66 ↑170

[56] Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993. ↑24 ↑28 ↑58 ↑58

[57] F. Karbalaie, A. Sami, and M. Ahmadi, "Semantic Malware Detection by Deploying Graph Mining," vol. 9, no. 1, pp. 373–379, 2012. ↑25

[58] S. Kaplan, C. Siefert, B. Livshits, B. Zorn, and C. Curtsinger, NO FUS : Automatically Detecting Obfuscated JavaScript Code, 2011. ↑25 ↑26

[59] B. S. Pintol and R. Barnete, "A novel algorithm for obfuscated code analysis," in Information Forensics and Security (WIFS), 2011 IEEE International Workshop, 2011, pp. 1–5. ↑25

[60] MalCrawler Code Hosted Online on GitHub. Available at: https://github.com/aksingh2411/MalCrawler. ↑28

[61] Google Safe Browsing API. Available: https://developers.google.com/safe-browsing. ↑29 ↑34 ↑35 ↑51 ↑54 ↑63 ↑67 ↑99 ↑126 ↑127 ↑132 ↑132 ↑137 ↑164 ↑168

[62] Dataset: "Android Malicious Dataset 2017 (CICAndMal2017)". Available: https://www.unb.ca/cic/datasets/andmal2017.html. ↑32 ↑50 ↑160 ↑166 ↑119

[63] Dataset: "Android Application Dataset for Malware Detection". Available: https://github.com/cskamil/Android-Application-Dataset-for-Malware-Detection. ↑32

[64] Dataset: "Android Antimalware Dataset". Available: https://github.com/ VT-Magnum-Research/antimalware/tree/master/arff. ↑32

[65] Utility for Downloading APKs from Google Play: "APK Combo". Available: https://apkcombo.com/en-in/. ↑32 ↑50 ↑160 ↑120

[66] JADX Disassembler for APKs. Available: https://github.com/skylot/jadx. ↑32 ↑119 ↑120 ↑168

[67] Singh, A. K., and Navneet Goyal. "Malcrawler: A crawler for seeking and crawling malicious websites." In International Conference on Distributed Computing and Internet Technology, pp. 210-223. Springer, 2017. https://doi.org/10.1007/978-3-319-50472-8_17. ↑34 ↑35 ↑156 ↑59 ↑62 ↑63 ↑64 ↑98

[68] Dataset, Visualization Code and Output, and Pre-processing Code are hosted on Mendeley Data. [Online] Available at: http://dx.doi. org/10.17632/gdx3pkwp47.1. ↑35 ↑45 ↑159

[69] Code for Visualization of Dataset hosted on Kaggle. [Online] Available at: https://www.kaggle.com/aksingh2411/visualisation-of-webpages-dataset. ↑35 ↑45

[70] Kohonen, Teuvo. "The self-organizing map." Proceedings of the IEEE 78, no. 9 (1990): 1464-1480. ↑46

[71] Code and Results of SOM-based Analysis of Webpages Dataset hosted online on Kaggle. Available at: https://www.kaggle.com/aksingh2411/som-model-of-webpages-dataset. ↑48

[72] A. K. Singh and N. Goyal, "A Comparison of Machine Learning Attributes for Detecting Malicious Websites," 11th International Conference on Communication Systems & Networks (COMSNETS 2019), Bengaluru, India, 2019, pp. 352-358. ↑156 ↑75 ↑76 ↑123 ↑125

[73] GeoIP Database. [Online] Available at: https://www.maxmind.com/en /geoip2-databases. ↑61 ↑63 ↑156 ↑167

[74] JavaScript Auto De-Obfuscator (JSADO). [Online] Available at: https://github.com/lucianogiuseppe/ JS-Auto-DeObfuscator. ↑98 ↑157 ↑168

[75] Selenium for Python. [Online] Available at: https://pypi.org/project/selenium. ↑157

[76] Code for De-obfuscation. [Online] Available at: https://github.com/ lucianogiuseppe/JS-Auto-DeObfuscator/blob/master/jsado.py. ↑157

[77] TLD Library. [Online] Available at: https://pypi.org/project/tld. ↑157

[78] WHOIS API. [Online] Available at: https://whois.whoisxmlapi.com. ↑158 ↑170

[79] Pre-processing code on Kaggle. [Online] Available at: https://www.kaggle.com/ aksingh2411/ sample-preprocessing-of-web-content-dataset-prepa/. ↑159

[80] Dataset: "Android Application Dataset for Malware Detection". Available: https://github.com/cskamil/Android-Application-Dataset-for-Malware-Detection. ↑50 ↑160 ↑166

[81] Dataset: "Android Antimalware Dataset". Available: https://github.com/ VT-Magnum-Research/antimalware/tree/master/arff. ↑50 ↑160 ↑166

[82] Dataset for "Machine Learning Analysis of Hybrid Apps". Available: www.kaggle.com/aksingh2411/hybrid-apps-security-analysis. ↑50 ↑56

[83] Visualization of Hybrid Apps Dataset: Live Run on Kaggle. Available at: https://www.kaggle.com/aksingh2411/visualization-of-hybrid-apps-dataset. ↑50 ↑56

[84] Disassembling code (in Java) using JADX disassembler: Available at: https://gist.github.com/aksingh2411/2867548195159cf1c8be3de149ecd161. ↑50 ↑160 ↑166

[85] Pre-processing code (in Python) for Hybrid Apps Dataset. Available at: https://gist.github.com/aksingh2411/b661ad5d65b2667fd365455aa99d0fae. ↑50 ↑166

[86] Google Chrome Custom Tabs. Available: https://developer.chrome.com/ multidevice/android/customtabs. ↑51 ↑161 ↑126 ↑127

[87] Song, Wei, Qingqing Huang, and Jeff Huang. "Understanding JavaScript Vulnerabilities in Large Real-World Android Applications." *IEEE Transactions on Dependable and Secure Computing* (2018). ↑52 ↑160 ↑160

[88] Custom Tabs Builder. Available at: https://developer.android.com/reference/androidx/browser/customtabs/CustomTabsIntent.Builder. ↑161

[89] JavaScript Settings in WebView. Available at: https://developer.android.com/reference/android/webkit/WebSettings#setJavaScriptEnabled(boolean). ↑161

[90] JavaScript Interface. Available at: https://developer.android.com/reference/android/webkit/JavascriptInterface. ↑161

[91] Manifest Permission. Available at: https://developer.android.com/reference/android/Manifest.permission. ↑162

[92] Blanc, G., Miyamoto, D., Akiyama, M., & Kadobayashi, Y. (2012, March). Characterizing obfuscated JavaScript using abstract syntax trees: Experimenting with malicious scripts. In 2012 26th International Conference on Advanced Information Networking and Applications Workshops (pp. 344-351). IEEE. ↑163

[93] WebViewClient Settings. Available at: https://developer.android.com/reference/android/webkit/WebViewClient. ↑163

[94] Loading URLs in Android WebView. Available at: https://developer.android.com/reference/android/webkit/WebView#loadUrl(java.lang.String,%20java.util.Map%3Cjava.lang.String,%20java.lang.String%3E). ↑164

[95] Evaluate JavaScript Method of Android WebView. Available at: https://developer.android.com/reference/android/webkit/WebView#evaluateJavascript(java.lang.String,%20android.webkit.ValueCallback%3Cjava.lang.String%3E. ↑165

[96] The shouldOverrideUrlLoading() Method of Android WebView. Available at: https://developer.android.com/reference/android/webkit/WebViewClient#shouldOverrideUrlLoading(android.webkit.WebView,%20android.webkit.WebResourceRequest). ↑165

[97] Cheeseman, Peter C., Matthew Self, James Kelly, Will Taylor, Don Freeman, and John C. Stutz. "Bayesian Classification." In AAAI, vol. 88, pp. 607-611. 1988. ↑58 ↑58

[98] Quinlan, J. Ross. "Induction of decision trees." Machine learning 1, no. 1 (1986): 81-106. ↑58

[99] Li, Bin, J. Friedman, R. A. Olshen, and C. J. Stone. "Classification and regression trees (CART)." Biometrics 40, no. 3 (1984): 358-361. ↑58

[100] Ho, Tin Kam. "Random decision forests." In Proceedings of 3rd international conference on document analysis and recognition, vol. 1, pp. 278-282. IEEE, 1995. ↑58

[101] Drucker, Harris, Chris JC Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. "Support vector regression machines." Advances in neural information processing systems 9 (1997): 155-161. ↑58 ↑58

[102] Ma J, Saul L.K., Savage S. and Voelker, 2009, June. Beyond blacklists: learning to detect malicious websites from suspicious URLs. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*(pp. 1245-1254). ACM. ↑60 ↑63

[103] Wressnegger C., Yamaguchi F., Arp D. and Rieck K., 2015. Analyzing and Detecting Flash-based Malware using Lightweight Multi-Path Exploration. *University of Göttingen, Germany*. ↑60 ↑64 ↑65

[104] Mavrommatis N.P.P. and Monrose, 2008. All your iframes point to us. ↑60 ↑64

[105] Ganesh N., Di Troia F., Corrado V.A., Austin, T.H. and Stamp M., 2016, March. Static analysis of malicious Java applets. In *Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics* (pp. 58-63). ACM. ↑60 ↑64 ↑65

[106] Mao J., Tian W., Li P., Wei T. and Liang Z., 2017. Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity. *IEEE Access*. ↑60

[107] Marco Cova, Christopher Kruegel, and Giovanni Vigna. 2010. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In *Proceedings of the 19th international conference on World wide web* (WWW' 10). ACM, New York, NY, USA, 281-290. ↑60 ↑64 ↑75 ↑76

[108] Gorji A. and Abadi M., 2014, March. Detecting obfuscated JavaScript malware using sequences of internal function calls. In *Proceedings of the 2014 ACM Southeast Regional Conference* (p. 64) ACM. ↑60 ↑65 ↑65

[109] Hall M.A. and Holmes G., 2003. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions on Knowledge and Data engineering*, *15*(6), pp.1437-1447. ↑60 ↑67

[110] Malware Domain List: 2017. [online] Available at: https://www.malwaredomainlist.com/. Accessed: 2017- 10- 07. ↑62 ↑67

[111] A. P. Felt, R. Barnes, A. King, C. Palmer, C. Bentzel, and P. Tabriz, "Measuring HTTPS adoption on the web," in 26th USENIX Security Symposium, 2017, pp. 1323–1338. ↑63

[112] K. A. Messabi, M. Aldwairi, A. A. Yousif, A. Thoban, and F. Belqasmi, "Malware detection using DNS records and domain name features," in Proceedings of the 2nd International Conference on Future Networks and Distributed Systems. ACM, 2018, p. 29. ↑63

[113] M. Klatt, B. W. Roberts, and T. C. Helming, "Domain reputation evaluation process and method," Apr. 6, 2017, US Patent App. 14/872,191. ↑63

[114] H. Mekky, R. Torres, Z.-L. Zhang, S. Saha, and A. Nucci, "Detecting malicious HTTP redirections using trees of user browsing activity," in INFOCOM, 2014 Proceedings IEEE. IEEE, 2014, pp. 1159–1167. ↑63 ↑65

[115] M. Hirotomo, Y. Nishio, M. Kamizono, Y. Fukuta, M. Mohri, and Y. Shiraishi, "Efficient method for analyzing malicious websites by using multi-environment analysis system," in 2017 12th Asia Joint Conference on Information Security (AsiaJCIS). IEEE, 2017, pp. 48–54. ↑64

[116] R. Wang, Y. Zhu, J. Tan, and B. Zhou, "Detection of malicious web pages based on hybrid analysis," Journal of Information Security and Applications, vol. 35, pp. 68–74, 2017. ↑64

[117] C. Wressnegger and K. Rieck, "Looking back on three years of flash-based malware," in Proceedings of the 10th European Workshop on Systems Security. ACM, 2017, p. 6. ↑64 ↑65

[118] B. Altay, T. Dokeroglu, and A. Cosar, "Context-sensitive and keyword density-based supervised machine learning techniques for malicious webpage detection," Soft Computing, pp. 1–15, 2018. ↑64 ↑64 ↑64

[119] N. Jagpal, E. Dingle, J.-P. Gravel, P. Mavrommatis, N. Provos, M. A. Rajab, and K. Thomas, "Trends and lessons from three years fighting malicious extensions." in USENIX Security Symposium, 2015, pp. 579- 593. ↑65 ↑65

[120] S. Morishige, S. Haruta, H. Asahina, and I. Sasase, "Obfuscated malicious javascript detection scheme using the feature based on divided URL," in Communications (APCC), 2017 23rd Asia-Pacific Conference on. IEEE, 2017, pp. 1-6. ↑65 ↑66 ↑66

[121] O. Sivan and Y. Lavi, "Web page and web browser protection against malicious injections," Jul. 18, 2017. ↑66

[122] H. Kikuchi, D. Yu, A Chander - US Patent 9, 686, and U. 2017, "Method and apparatus for constructing security policies for web content instrumentation against browser-based attacks," 2017. [Online]. Available: https://patents.google.com/patent/US9686288B2/en. ↑66 ↑66

[123] A. K. Singh, "Dataset for Comparison of ML Attributes for Detecting Malicious Websites," 2018. [Online]. Available: http://dx.doi.org/10.17632/stctg82wsf.1. ↑67

[124] Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. "SMOTE: synthetic minority over-sampling technique." Journal of artificial intelligence research 16 (2002): 321-357. ↑67

[125] Google Transparency Report. [Online] Available at: https://transparencyreport.google.com/. ↑75

[126] Kapravelos, Alexandros, Chris Grier, Neha Chachra, Christopher Kruegel, Giovanni Vigna, and Vern Paxson. "Hulk: Eliciting malicious behavior in browser extensions." In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pp. 641-654. 2014. ↑75

[127] Eshete, Birhanu, Adolfo Villafiorita, and Komminist Weldemariam. "Binspect: Holistic analysis and detection of malicious web pages." In *International Conference on Security and Privacy in Communication Systems*, pp. 149-166. Springer, Berlin, Heidelberg, 2012. ↑75 ↑76

[128] Akiyama, Mitsuaki, Makoto Iwamura, Yuhei Kawakoya, Kazufumi Aoki, and Mitsutaka Itoh. "Design and implementation of high interaction client honeypot for drive-by-download attacks." *IEICE transactions on communications* 93, no. 5 (2010): 1131-1139. ↑76

[129] Yoo, Suyeon, Sehun Kim, Anil Choudhary, O. P. Roy, and T. Tuithung. "Two-phase malicious web page detection scheme using misuse and anomaly detection." *International Journal of Reliable Information and Assurance* 2, no. 1 (2014): 1-9. ↑76

[130] Wang, Rong, Yan Zhu, Jiefan Tan, and Binbin Zhou. "Detection of malicious web pages based on hybrid analysis." *Journal of Information Security and Applications* 35 (2017): 68-74. ↑76 ↑90

[131] David, Eli, Nadav Maman, and Guy Caspi. "Methods and systems for detecting malicious webpages." US Patent Application 15/641,851, filed January 10, 2019. ↑77

[132] Shrivastava, Vishal, Shashank Satish Damodaran, and Megha Kamble. "Adalward: a deep-learning framework for multi-class malicious webpage detection." *Journal of Cyber Security Technology* (2020): 1-43. ↑77

[133] Vinayakumar, R., K. P. Soman, and Prabaharan Poornachandran. "Evaluating deep learning approaches to characterize and classify malicious URL's." *Journal of Intelligent & Fuzzy Systems* 34, no. 3 (2018): 1333-1343. ↑77 ↑90

[134] Wang, Huan-huan, Long Yu, Sheng-wei Tian, Yong-fang Peng, and Xin-jun Pei. "Bidirectional LSTM Malicious webpages detection algorithm based on convolutional neural network and independent recurrent neural network." *Applied Intelligence* 49, no. 8 (2019): 3016-3026. ↑77

[135] Zhang, Qingchen, Laurence T. Yang, Zhikui Chen, and Peng Li. "A survey on deep learning for big data." *Information Fusion* 42 (2018): 146-157. ↑77

[136] Schmidhuber, Jürgen. "Deep learning in neural networks: An overview." *Neural networks* 61 (2015): 85-117. ↑78

[137] Le Roux, Nicolas, and Yoshua Bengio. "Deep belief networks are compact universal approximators." *Neural computation* 22, no. 8 (2010): 2192-2207. ↑78

[138] Profanity Check Computation. [Online] Available at: https://pypi.org/project/profanity-check. ↑37

[139] Singh, AK; Goyal, Navneet (2020), "Dataset of Malicious and Benign Webpages", MendeleyData, v1. [Online] Available at: http://dx.doi.org/10.17632/gdx3pkwp47. ↑144

[140] Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* 15, no. 1 (2014): 1929-1958. ↑80 ↑99

[141] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014). ↑81

[142] Nielsen, Michael A. *Neural networks and deep learning*. Vol. 2018. San Francisco, CA, USA: Determination press, 2015. ↑81

[143] Johnson, Justin M., and Taghi M. Khoshgoftaar. "Survey on deep learning with class imbalance." *Journal of Big Data* 6, no. 1 (2019): 27. ↑85

[144] Amin, Adnan, Sajid Anwar, Awais Adnan, Muhammad Nawaz, Newton Howard, Junaid Qadir, Ahmad Hawalah, and Amir Hussain. "Comparing oversampling techniques to handle the class imbalance problem: A customer churn prediction case study." *IEEE Access* 4 (2016): 7940-7957. ↑85 ↑120

[145] Qiu, Qiang, and Zichen Song. "A nonuniform weighted loss function for imbalanced image classification." In *Proceedings of the 2018 international conference on image and graphics processing*, pp. 78-82. 2018. ↑86

[146] A Receipe for Training Neural Networks. [Online] Available at: http://karpathy.github.io/2019/04/25/recipe/#2-set-up-the-end-to-end-trainingevaluation-skeleton--get-dumb-baselines. ↑86

[147] TensorFlow. [Online] Available at: https://www.tensorflow.org. ↑87 ↑104 ↑145 ↑170

[148] Keras. [Online] Available at: https:// https://keras.io. ↑87 ↑104 ↑145 ↑169

[149] Software Code published on Kaggle. [Online] Available at: https://www.kaggle.com/aksingh2411/malicious-webpage-detection- with-unstructured-data. ↑87

[150] TensorFlow.js. [Online] Available at: https://www.tensorflow.org/js. ↑91 ↑109 ↑154

[151] Live Internet Statistics. Available at: https://www.internetlivestats.com /total-number-of-websites. ↑92

[152] Fang, Yong, Yang Li, Liang Liu, and Cheng Huang. "DeepXSS: Cross site scripting detection based on deep learning." In Proceedings of the 2018 International Conference on Computing and Artificial Intelligence, pp. 47-51. 2018. ↑77

[153] Vinayakumar, R., K. P. Soman, and P Poornachandran. "Evaluating deep learning approaches to characterize and classify malicious URL's." *Journal of Intelligent & Fuzzy Systems* 34, (2018). ↑94

[154] Wang, H Huan, L Yu, S Tian, Y Peng, and Pei. "Bidirectional LSTM Malicious webpages detection algorithm based on convolutional neural network and independent recurrent neural network." *Applied Intelligence* 49, no. 8 (2019): 3016-3026. ↑94

[155] Shrivastava, Vishal, S Damodaran, and M Kamble. "Adalward: a deep-learning framework for multi-class malicious webpage detection." *Journal of Cyber Security Technology* (2020): 1-43. ↑94

[156] Peters, Matthew, M Neumann, M Iyyer, M Gardner, K Lee, and L Zettlemoyer. "Deep contextualized word representations." *arXiv preprint*. ↑95

[157] Vaswani, Ashish, N Shazeer, N Parmar, J Uszkoreit, L Jones, Aidan N. Gomez, Kaiser, and I Polosukhin. "Attention is all you need." In *Advances in neural information processing systems*, 2017. ↑95 ↑109

[158] Devlin, Jacob, M Chang, and K Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint*. ↑95 ↑108 ↑109

[159] Cer, Daniel, Y Yang, S Kong, N Hua, N Limtiaco, R St John, N Constant, et al. "Universal sentence encoder." *arXiv preprint* (2018). ↑95

[160] TensorFlow Hub. Available at: https://tfhub.dev. ↑96

[161] Dataset of Random Web Content and JavaScript. Available at: https://www.kaggle.com/aksingh2411/webjavascripttext-size50. ↑97

[162] Code for De-obfuscation. Available at: https://github.com/aksingh2411 /JSADO. ↑98

[163] Singh, A. K., and Navneet Goyal. (2020). "Dataset of Malicious and Benign Webpages". *Elsevier Data in Brief*. ↑144

[164] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014). ↑99

[165] Nielsen, Michael A. *Neural networks and deep learning*. Vol. 2018. San Francisco, CA, USA: Determination press, 2015. ↑101

[166] A Recipe for Training Neural Networks. Available at: http://karpathy.github.io/2019/04/25/recipe/#2-set-up-the-end-to-end-trainingevaluation-skeleton--get-dumb-baselines. ↑104

[167] Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." *Journal of machine learning research* , (2012). ↑104

[168] Skikit-learn. Available at: https://scikit-learn.org. ↑105 ↑119 ↑121 ↑123 ↑170

[169] Jouppi, Norman P., C Young, et al. "In-datacenter performance analysis of a tensor processing unit." In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pp. 1-12. 2017. ↑105

[170] Code for Stage-I: Pre-training Autoencoders. Available at: https://www.kaggle.com/aksingh2411/unsupervised-learning-web-content-javascript. ↑105

[171] Code for Stage-II: DNN Classifier. Available at: https://www.kaggle.com/aksingh2411/stage-ii-dnn-model-for-webcontent-classification. ↑105

[172] Code for Grid Search of Hyperparameters. Available at: https://www.kaggle.com/aksingh2411/grid-search-tuning-of-hyper-parameters. ↑105

[173] Details of Number of Android Devices and Versions Active. Available: https://www.androidpolice.com/2019/05/07/android-distribution-numbers-finally-back-for-may-2019-after-6-months/. ↑113

[174] Hybrid Apps Beats Native Apps in Survey. Available: https://adtmag.com/articles/2017/07/28/hybrid-beats-native.aspx. ↑114

[175] Chin, Erika, and David Wagner. "Bifocals: Analyzing WebView vulnerabilities in android applications." In International Workshop on Information Security Applications, pp. 138-159. Springer, Cham, 2013. ↑115

[176] Bao, Wenying, Wenbin Yao, Ming Zong, and Dongbin Wang. "Cross-site Scripting attacks on Android hybrid applications." In *Proceedings of the 2017 International Conference on Cryptography, Security and Privacy*, pp. 56-61. ACM, 2017. ↑115

[177] Wei, Fengguo, Yuping Li, Sankardas Roy, Xinming Ou, and Wu Zhou. "Deep ground truth analysis of current android malware." In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 252-276. Springer, Cham, 2017. ↑115

[178] Rizzo, Claudio, Lorenzo Cavallaro, and Johannes Kinder. "BabelView: Evaluating the Impact of Code Injection Attacks in Mobile Webviews." In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pp. 25-46. Springer, Cham, 2018. ↑115

[179] Hu, Jiajun, Lili Wei, Yepang Liu, Shing-Chi Cheung, and Huaxun Huang. "A tale of two cities: How WebView induces bugs to Android applications." In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pp. 702-713. ACM, 2018. ↑115

[180] Song, Wei, Qingqing Huang, and Jeff Huang. "Understanding JavaScript Vulnerabilities in Large Real-World Android Applications." *IEEE Transactions on Dependable and Secure Computing* (2018). ↑115

[181] Sexton, Julian, Andrey Chudnov, and David A. Naumann. "Spartan Jester: end-to-end information flow control for hybrid Android applications." In *2017 IEEE Security and Privacy Workshops (SPW)*, pp. 157-162. IEEE, 2017. ↑115

[182] Li, Tongxin, Xueqiang Wang, Mingming Zha, Kai Chen, XiaoFeng Wang, Luyi Xing, Xiaolong Bai, Nan Zhang, and Xinhui Han. "Unleashing the walking dead: Understanding cross-app remote infections on mobile WebViews." In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 829-844. ACM, 2017. ↑115

[183] Mohsen, Fadi, and Mohamed Shehab. "Proposing and Testing New Security Cue Designs for OAuth-WebView-Embedded Mobile Applications." In *2017 IEEE 3rd International Conference on Collaboration and Internet Computing (CIC)*, pp. 443-448. IEEE, 2017. ↑116

[184] Imamura, Yuta, Hiroyuki Uekawa, Yasuhiro Ishihara, Masaya Sato, and Toshihiro Yamauchi. "Web access monitoring mechanism for Android WebView." In *Proceedings of the Australasian Computer Science Week Multiconference*, p. 1. ACM, 2018. ↑116

[185] The Webkit project. Available at: https://www.webkit.org. ↑116

[186] The Chromium Project. Available at: https://www.chromium.org. ↑117

[187] WebView & Chrome Similarity & Interoperability. Available: https://developer.chrome.com/multidevice/webview/overview. ↑117

[188] Working with Android WebView. Available: https://androidride.com/ android-webview-example-tutorial-kotlin-java-download-source-code/. ↑117

[189] WebView Developer Resource. Available: https://developer.android.com/guide/webapps/webview#java. ↑117

[190] Google Play Store Download of "WebView Tools". Available: https://play.google.com. ↑119

[191] Source Code of "WebView Tool". Available: https://github.com/ aksingh2411/WebViewTool. ↑119 ↑153

[192] Dataset: "Android Application Dataset for Malware Detection". Available: https://github.com/cskamil/Android-Application-Dataset-for-Malware-Detection. ↑119

[193] Dataset: "Android Antimalware Dataset". Available: https://github.com/ VT-Magnum-Research/antimalware/tree/master/arff. ↑119

[194] Hall, Mark A., and Geoffrey Holmes. "Benchmarking attribute selection techniques for discrete class data mining." *IEEE Transactions on Knowledge and Data engineering* 15, no. 6 (2003): 1437-1447. ↑123

[195] Source Code of "WebView Monitor" Android App. Available: https://github.com/aksingh2411/Web-View-Monitor. ↑128 ↑153

[196] UIWebView Component. Available at: https://developer.apple.com/ documentation/uikit/uiwebview. ↑128 ↑155

[197] Konečný, Jakub, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. "Federated learning: Strategies for improving communication efficiency." *arXiv preprint arXiv:1610.05492* (2016). ↑129

[198] Statistics of Smart Phones globally. Available at: https://www.statista.com /statistics/330695/number-of-smartphone-users-worldwide. ↑130

[199] Lindell, Yehida. "Secure multiparty computation for privacy preserving data mining." In Encyclopedia of Data Warehousing and Mining, pp. 1005-1009. IGI global, 2005. ↑131 ↑134

[200] Gentry, Craig. "Fully homomorphic encryption using ideal lattices." In Proceedings of the forty-first annual ACM symposium on Theory of computing, pp. 169-178. 2009. ↑131 ↑131 ↑134

[201] Dwork, Cynthia, and Adam Smith. "Differential privacy for statistics: What we know and what we want to learn." Journal of Privacy and Confidentiality 1, no. 2 (2010). ↑131 ↑134

[202] Dwork, Cynthia, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating noise to sensitivity in private data analysis." Journal of Privacy and Confidentiality 7, no. 3 (2016): 17-51. ↑131 ↑131 ↑134

[203] Bonawitz, Keith, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon et al. "Towards federated learning at scale: System design." arXiv preprint arXiv:1902.01046 (2019). ↑131 ↑133 ↑135

[204] Differential Privacy Team. Learning with privacy at scale. Apple Machine Learning Journal, 1(8), 2017. [Online] Available at: https://machinelearning. apple.com/2017/12/06/learning-with-privacy-at-scale.html. ↑131 ↑133

[205] Liu, Lumin, Jun Zhang, S. H. Song, and Khaled B. Letaief. "Client-edge-cloud hierarchical federated learning." In ICC 2020-2020 IEEE International Conference on Communications (ICC), pp. 1-6. IEEE, 2020. ↑131 ↑133 ↑133 ↑142 ↑142

[206] Chakraborty, Anirban, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. "A survey on adversarial attacks and defences." CAAI Transactions on Intelligence Technology (2021). ↑132

[207] Hard, Andrew, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. "Federated learning for mobile keyboard prediction." arXiv preprint arXiv:1811.03604 (2018). ↑132 ↑133

[208] Virus Total API. [Online] Available at: https://developers.virustotal.com/reference. ↑132

[209] Yang, Timothy, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. "Applied federated learning: Improving google keyboard query suggestions." arXiv preprint arXiv:1812.02903 (2018). ↑133

[210] Sheller, Micah J., Brandon Edwards, G. Anthony Reina, Jason Martin, Sarthak Pati, Aikaterini Kotrotsou, Mikhail Milchenko et al. "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data." Scientific reports 10, no. 1 (2020): 1-12. ↑133

[211] U´ lfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: Randomized aggregatable privacy preserving ordinal response. In ACM CCS, 2014. ISBN 978-1-4503-2957-6. doi: 10.1145/2660267.2660348. ↑133

[212] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Francoise Beaufays. Applied federated learning: Improving Google keyboard query suggestions. arXiv preprint 1812.02903, 2018. ↑133

[213] Mingqing Chen, Rajiv Mathews, Tom Ouyang, and Franc¸oise Beaufays. Federated learning of out-of-vocabulary words. arXiv preprint 1903.10635, 2019. ↑133

[214] Swaroop Ramaswamy, Rajiv Mathews, Kanishka Rao, and Franc¸oise Beaufays. Federated learning for emoji prediction in a mobile keyboard. arXiv preprint 1906.04329, 2019. ↑133

[215] Your chats stay private while Messages improves suggestions, 2019. [Online] Available at: URL https://support.google.com/messages/ answer/9327902. ↑133

[216] Apple. Private Federated Learning (NeurIPS 2019 Expo Talk Abstract). https://nips.cc/ExpoConferences/2019/schedule?talk_id=40, 2019. ↑133

[217] Apple. Designing for privacy (video and slide deck). Apple WWDC, https://developer.apple.com/videos/play/wwdc2019/708, 2019. ↑133

[218] David Leroy, Alice Coucke, Thibaut Lavril, Thibault Gisselbrecht, and Joseph Dureau. Federated learning for keyword spotting. arXiv preprint arXiv:1810.05512, 2018. ↑133

[219] Wainakh, Aidmar, Alejandro Sanchez Guinea, Tim Grube, and Max Mühlhäuser. "Enhancing privacy via hierarchical federated learning." In 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 344-347. IEEE, 2020. ↑133 ↑142

[220] Wang, Jiayi, Shiqiang Wang, Rong-Rong Chen, and Mingyue Ji. "Local Averaging Helps: Hierarchical Federated Learning and Convergence Analysis." arXiv preprint arXiv:2010.12998 (2020). ↑133 ↑142

[221] Singh, A. K., and Navneet Goyal. "Understanding and Mitigating Threats from Android Hybrid Apps Using Machine Learning." In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 1-9. IEEE, 2020. ↑134

[222] Milosevic, Nikola, Ali Dehghantanha, and Kim-Kwang Raymond Choo. "Machine learning aided Android malware classification." Computers & Electrical Engineering 61 (2017): 266-74. ↑134

[223] Li, Jin, Lichao Sun, Qiben Yan, Zhiqiang Li, Witawas Srisa-An, and Heng Ye. "Significant permission identification for machine-learning-based android malware detection." IEEE Transactions on Industrial Informatics 14, no. 7 (2018): 3216-3225. ↑134

[224] Ma, Zhuo, Haoran Ge, Yang Liu, Meng Zhao, and Jianfeng Ma. "A combination method for android malware detection based on control flow graphs and machine learning algorithms." IEEE Access 7 (2019): 21235-21245. ↑134

[225] Al-Rubaie, Mohammad, and J. Morris Chang. "Privacy-preserving machine learning: Threats and solutions." IEEE Security & Privacy 17, no. 2 (2019): 49-58. ↑134

[226] Kairouz, Peter, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz et al. "Advances and open problems in federated learning." arXiv preprint arXiv:1912.04977 (2019). ↑135

[227] Yang, Qiang, Yang Liu, Tianjian Chen, and Yongxin Tong. "Federated machine learning: Concept and applications." ACM Transactions on Intelligent Systems and Technology (TIST) 10, no. 2 (2019): 1-19. ↑135

[228] Zhuang, Fuzhen, et al. "A comprehensive survey on transfer learning." Proceedings of the IEEE 109.1 (2020): 43-76. ↑110 ↑139

[229] Implementation of LSTM based Autoencoder. [Online] Available at: https://www.kaggle.com/aksingh2411/stage-i-unsupervisedlearning-webcontent-javascript. ↑139

[230] Bonawitz, Keith, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. "Practical secure aggregation for federated learning on user-held data." arXiv preprint arXiv:1611.04482 (2016). ↑140

[231] Abadi, Martin, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep learning with differential privacy." In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 308-318. 2016. ↑140

[232] Code for Federated Learning based Android Web Security Experiment and simulation. [Online] Available at: https://colab.research.google.com/drive/135eNF4si3F4zT_Cxhhc3A_8SqmdHuxKT?usp=sharing. ↑141 ↑145

[233] Chen, Bryant, et al. "Detecting backdoor attacks on deep neural networks by activation clustering." arXiv preprint arXiv:1811.03728 (2018). ↑141

[234] Tran, Brandon, Jerry Li, and A Madry. "Spectral signatures in backdoor attacks." Advances in Neural Information Processing Systems. 2018. ↑141

[235] Cao, Xiaoyu, Jinyuan Jia, and Neil Zhenqiang Gong. "Provably Secure Federated Learning against Malicious Clients." arXiv preprint arXiv:2102.01854 (2021). ↑141

[236] Adversarial Robustness Toolbox (ART). [Online] Available at: https://adversarial-robustness-toolbox.org/. ↑142 ↑167

[237] Source Code of Adversarial Robustness Checks on FL Model. Available at: https://github.com/aksingh2411/Adversarial-Robustness-on-Federated-Learning. ↑142

[238] Source code of Hierachical Federated Learning based Android Web Security. Available at: https://gist.github.com/aksingh2411/6f6091fd3105e30d93298b4d72edfdea. ↑143

[239] Zhao, Yue, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. "Federated learning with non-iid data." arXiv preprint arXiv:1806.00582 (2018). ↑144

[240] Tensorflow Federated. [Online] Available at: https://www.tensorflow.org/federated/api_docs/python/tff. ↑145 ↑170

[241] Source Code Hosted on Git Hub. [Online] Available at: https://github.com/aksingh2411/JSADO/blob/master/Copy_of_Federated_Deep_Learning_Approach_to_Websecurity_Run_with_TFF_Wrapper.ipynb. ↑145

[242] The Android SDK. [Online] Available at: https://developer.android.com/. ↑167

[243] Android Studio IDE. [Online] Available at: https://developer.android.com/studio. ↑167

[244] Google Colab. [Online] Available at: https://colab.research.google.com/. ↑167

[245] JupyterLab IDE. [Online] Available at: https://jupyter.org/. ↑168

[246] Kaggle. [Online] Available at: https://www.kaggle.com/. ↑168

[247] NetBeans IDE and Profiler. [Online] Available at: https://netbeans.apache.org/. ↑169

[248] NumPy Library. [Online] Available at: https://numpy.org/. ↑169

[249] PostgreSQL. [Online] Available at: https://www.postgresql.org/. ↑169

[250] Seaborn Data Visualization Library. [Online] Available at: https://seaborn.pydata.org/. ↑170

[251] Tensor Processing Unit. [Online] Available at: https://cloud.google.com/tpu. ↑170

[252] TensorBoard Visualization Library. [Online] Available at: https://www.tensorflow.org/tensorboard. ↑170