Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN: 2581 - 611X)
VOLUME:8, ISSUE:2, AUGUST 2025, PAPER NUMBER:25

Optimizing Mobile Application for Bandwidth-Challenged Networks in Developing Regions

Anitha G [1], Boomika P [2], Devika S [3], Jeyavarthini R [4], Srinivas B. M.Tech.[5]

[1,2,3,4] Students and [5] Faculty
Dept. of Information Technology, V.S.B. Engineering College (Autonomous), Karur, India.
jeyavarthini11@gmail.com

Abstract: *The necessity of optimising mobile applications in low-bandwidth settings is emphasised in this abstract. Customers often have trouble accessing applications in these areas since mobile networks frequently fail to provide sufficient connectivity. The quantity of data required for an app can be decreased by developers with the aid of data compression, caching, and effective data transport protocols. Additionally, introducing offline functionality enables customers to continue using essential features even in the event that internet connectivity is lost. Apart from enhancing the user experience, these solutions also increase the accessibility of mobile applications in places where connectivity is expensive or unreliable. The abstract also looks at how mobile application optimisation affects developing nations' socioeconomic systems. By requiring less data and operating more efficiently on low-bandwidth networks, mobile applications can reach a wider audience. This accentuates the necessity of improving mobile applications in bandwidth-constrained places. Customers often have trouble*
accessing applications in these areas since mobile networks frequently fail to provide sufficient connectivity. The quantity of data required for an app can be decreased by developers with the aid of data compression, caching, and effective data transport protocols. Additionally, introducing offline functionality enables customers to continue using essential features even in the event that internet connectivity is lost. Apart from enhancing the user experience, these solutions also increase the accessibility of mobile applications in places where connectivity is expensive or unreliable. The abstract also looks at how mobile application optimisation affects developing nations' socioeconomic systems.

Keywords: *Mobile applications, code bloat, Bandwidth- networks, Network optimization, Mini-programs*

1. INTRODUCTION:

Optimisation of mobile applications for networks with limited bandwidth in developing nations is a crucial topic covered in this paper. Here, users' inability to use mobile applications is often caused by inadequate network infrastructure, which creates significant barriers to commerce, education, healthcare, and communication. This study explores different approaches to optimise mobile applications for low-bandwidth environments, realising the importance of addressing these issues. Using techniques like data compression, caching, and offline capabilities, developers can improve the user experience for users in these places by lessening the negative effects of limited connectivity. This article delves into the socioeconomic implications of optimising mobile applications, highlighting the potential for enhanced accessibility to digital services to foster community empowerment, stimulate economic growth, and bridge the digital divide. In order to shed light on the significance of mobile application optimisation and its potential advantages in underprivileged areas which could ultimately result in more accessible and inclusive digital ecosystems this research combines technical and socioeconomic studies.

1.1. NETWORKING TEHNOLOGY

In the network generation of technology is differentiated in the technical work process of mobile cellular networks by unique features and speed ranges that directly affect buffer range and data transmission capacities. The Global System for Mobile Communications (GSM) and Code Division Multiple Access (CDMA) are the mainstays of 2G networks, which are the first generation of mobile technology. With

buffer levels that minimise data caching and transmission delays, these networks usually provide data speeds between 10 and 100 Kbps. With the advent of 3G networks and technologies like CDMA2000 and UMTS (Universal Mobile Telecommunications System), data speeds increased significantly to 0.5 to 10 Mbps. Buffer ranges widened to accommodate larger data packets and reduce buffering waits, resulting in faster streaming of low-quality videos and basic web browsing. Data speeds rose even further with the advent of 4G LTE (Long-Term Evolution) technology, typically reaching ranges of 10 Mbps to 100 Mbps and higher. Faster web browsing, online gaming, and streaming of high-definition videos were all made possible by this enhanced speed. More buffer range expansions led to faster buffering times and more seamless data transfer, especially for applications requiring a lot of bandwidth.

Mobile networks are currently being transformed by 5G technology at speeds between hundreds of Mbps and several Gbps. Real-time applications like augmented reality, virtual reality, and remote surgery are made possible by this incredibly rapid connectivity. 5G networks now have significantly longer buffer ranges, allowing uninterrupted data delivery even under circumstances of high demand. Within the technical pipeline of network technologies, generational advancements have mostly focused on increasing data throughput, reducing latency, and extending buffer ranges to handle larger data sets. Users now have more uninterrupted access to a wider range of applications and services, making for a more seamless and responsive mobile experience.

Most mobile apps are designed to take use of high-speed networks in places with lots of bandwidth, and developers often make updates to improve functionality and security. Recent research indicates that updates are regularly released for 10 out of the top twelve most popular programs.

Facebook: One of the most widely used software, Facebook frequently publishes updates to enhance security and user experience. Generally, the size of the installation package is between 50 and 100 MB. Upgrades can be quickly received by users in places with plenty of bandwidth, especially those connected to 4G and 5G networks. Instagram: Facebook owns Instagram, which is updated frequently as well. The size of the installation package ranges from 30 to 70 MB. These updates are rather easy for users of 4G and 5G networks to obtain.

The Twitter app is updated frequently with new features and bug fixes. The size of an installation package often falls between 40 and 80 MB. These upgrades are readily available to users on 4G and 5G networks. Popular messaging app WhatsApp is updated frequently to enhance efficiency and security. The size of the installation package varies from 30 to 60 MB. These updates can be installed quickly by users with 3G or higher. Snapchat is well-known for its regular updates, which provide new features and filters. Typically, the installation package has a size of 60 to 100 MB. These improvements are easily accessible to users on 4G and 5G networks.

The quality of video streaming and the user experience are enhanced by frequent app upgrades from YouTube. Between 50 and 100 MB make up the installation bundle. The rapid delivery of these updates is made possible by the high bandwidth of 4G and 5G networks. Google Maps: In order to provide precise location and navigation services, Google Maps is updated frequently. Between 50 and 100 MB make up the installation bundle. Rapid delivery of these updates is what users of 4G and 5G networks can anticipate. With new features and improved streaming quality, Netflix has updated its app. Between 60 and 100 MB make up the installation bundle. Fast upgrades are available to users on 4G and 5G networks.

2. RELATED WORK:

Rabia Shaheen, Faheem Babar Dr [1], the revolution in information technology has brought about changes in people's lifestyles and work routines. Because of the wonderful features of mobile applications, it has also totally changed the traffic on the internet. The use of mobile devices has become more popular than ever, and people are looking for answers to more and more everyday issues. This includes the need for processing power, communication flexibility, and information exchange via mobile apps. The architecture, key elements, and function of mobile applications in cutting-edge technology are all examined in this study. Additionally, a thorough comparison of the functionality of different platforms with regard to mobile application performance is provided.

I. Sawad, inas, sawad [2], the goal of this project is to examine how bandwidth considerations might be assessed using simulation while developing or upgrading cellular infrastructure in developing countries. When determining the maximum number of users that a system can accommodate, bandwidth is an important resource that needs to be utilised effectively. Sustaining end-to-end delays below acceptable thresholds and supporting capacity are directly impacted by bandwidth. The method for optimising bandwidth utilisation while accounting for capacity and latency considerations is suggested in this study, along with an identification of critical features that affect system architecture.

Carlos Bernal-Cardenas, Nathan Cooper [3] Crowdsourced app feedback is often obtained through screen recordings of mobile applications, which are easy to obtain and include a wealth of information that is helpful to software developers (e.g., bugs or feature requests). Consequently, these videos are turning into a regular artefact that developers have to deal with. The unique limitations of mobile development such as brief release cycles and rapidly shifting platforms make automated methods for evaluating different kinds of rich software artefacts useful to mobile developers. Sadly, because screen recordings are graphical in nature as opposed to other types of (textual) objects, automatically assessing them presents significant challenges.

M. A. ridwan, n. A. M. radzi [4] ML techniques can provide computational models for solving complicated communication network challenges, hence improving performance. This paper discusses recent trends in the use of machine learning models in communication networks for prediction, intrusion detection, route and path assignment, Quality of Service improvement, and resource management. A survey of the recent literature suggests numerous opportunities for academics to use the benefits of ML in solving difficult network performance issues, particularly with the growth of software-defined networks and 5G.

Obinna Izima, ORCID, Ruairí de Frein [5] The methods used to forecast QoD for video fall into six categories: (1) video quality prediction in QoD impairments; (2) video quality prediction from encrypted video streaming traffic; (3) video quality prediction in HAS applications; (4) video quality prediction in SDN applications; (5) video quality prediction in wireless settings; and (6) video quality prediction in WebRTC applications. A few research issues and directions in this area are included in the survey: (1) machine learning vs deep learning; (2) adaptive deep learning for better video distribution; (3) computational cost and interpretability; and (4) self-healing networks and failure recovery. The survey indicates that the most widely used models for video quality prediction are traditional machine learning methods.

Yu Su, Shuijie Wang, Qianqian Cheng [6] This work proposes a reinforcement learning based video stream scheduling methodology and a deep learning-based buffer starvation evaluation tool. This method uses machine learning to determine the relationship between traffic load and the buffer starvation probability distribution. The outcome is a set of resource allocation techniques that maximises long-term quality of experience (QoE) and explicit evaluation findings of buffer starvation incidents. The approach integrates an internal incentive system into the scheduling process to mitigate the noise issue caused by the randomly generated environment, thereby enabling the agent to fully investigate the surroundings. Tests have demonstrated that our system is capable of precisely assessing and enhancing the video service quality of 5G-enabled unmanned aerial vehicles.

Jasneet kaur, M. Arif khan [7] The next generation of these systems is always evolving, and fifth-generation (5G) wireless networks are currently being deployed globally. The sixth generation (6G) of wireless technology, which will come after the fifth generation, is already being discussed by academics and business. Using artificial intelligence (AI) and machine learning (ML) for wireless networks will be one of the key elements of 6G systems. All of the building blocks and parts of a wireless system, including the physical, network, and application layers that we now know from our understanding of wireless technologies up to 5G will make use of one or more AI/ML techniques.

Hassan Farsi and Pouriya Etezadifar [8] this work aims to increase PSNR for synthesised video by increasing the channel encoder rate while keeping the transmission rate constant. Artificial neural networks and Huffman coding are used in VLC blocks of the MPEG standard to significantly reduce transmitted data size. Afterwards, a secondary channel encoder is used to recode the compressed data according to how much compression was accomplished using the recommended method. Without adding more data to each frame, the suggested approach can increase channel coding rate. By using this

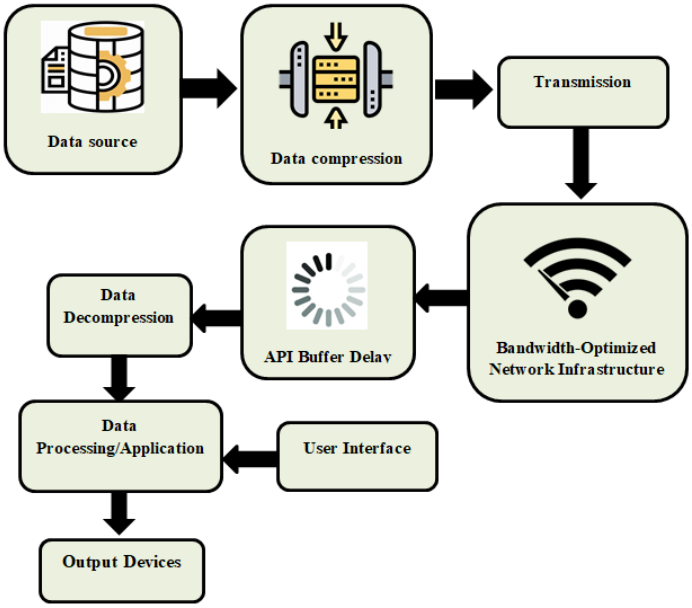
method, video frames are more resilient to channel errors. The suggested approach is evaluated at different channel SNRs and source coding rates, and the outcomes are contrasted with a recently developed technique called the Farooq method.

3. PROPOSED SYSTEM:

The proposed method aims to tackle the problem of code bloat and its impact on network performance, specifically with mobile application video buffering. The system will employ a number of techniques to achieve this goal, including maximising code efficiency, decreasing installation package size, and lowering network data consumption. The programme will be smaller overall because the system will employ code analysis techniques to identify and remove superfluous or unnecessary code.

This will mean optimising resource-intensive processes, eliminating superfluous libraries, and simplifying algorithms in order to boost efficiency and lower the likelihood of video buffering. When streaming videos over the network, the system will apply data compression algorithms to minimise the amount of data transferred.

The technology can reduce bandwidth usage and provide faster loading times and smoother playback even on networks with limited capacity by compressing video and other multimedia files. In addition, the system will include caching techniques to save data that is frequently accessed locally on the user's device. By doing this, you can improve performance and lessen video buffering by relying less on the network and fewer frequent network queries. Additionally, in order to lessen the effect of code bloat on network speed, the system would load essential components first and prioritise the transmission of important information and services. By employing strategies such as selective loading and lazy loading, the system is able to maximise network utilisation and maintain a consistent user experience, even in locations with limited bandwidth. In order to improve efficiency, reduce data usage, and enhance user experience, the proposed solution applies a comprehensive suite of optimisation techniques to address the issues of code bloat and network performance in mobile applications, particularly video buffering.



Optimising data transmission is essential for guaranteeing effective communication and resource utilisation in networks with limited capacity. Using bandwidth optimisation and API buffer delay techniques is one possibility. Queuing API requests is a necessary part of API buffer delay management in order to manage the flow of incoming data and prevent network overload. The system may balance incoming and outgoing data flows, prevent bottlenecks, and maximise bandwidth utilisation by purposefully delaying API responses. By guaranteeing that data is transferred at a speed that the network can handle, this delay reduces the possibility of congestion or packet loss. Strategies for optimising bandwidth prioritise important data and adjust traffic to make the most use of available bandwidth, hence improving network performance. This can involve techniques like adaptive bitrate streaming,

traffic prioritisation, and data compression. By reducing the size of data packets, compression methods like LZ77 and Huffman coding enable more data to be transmitted over the available bandwidth. A smoother user experience is achieved by adaptive bitrate streaming, which dynamically adjusts the quality of multimedia content based on available bandwidth. Prioritising critical data ensures that key information is provided without delay. The system can efficiently handle data transmission in networks with limited capacity by merging bandwidth optimisation techniques with API buffer delay control. This guarantees the timely and effective delivery of critical data while also improving network performance and stability a crucial feature in challenging situations such as developing nations with underdeveloped network infrastructures.

4. CODE BLOAT:

Code bloat is becoming a big problem in the rapidly evolving world of mobile applications, especially when it comes to network speed and video streaming. Code bloat is the unnecessary accumulation of superfluous or ineffective code in mobile applications, leading to larger installation packages and increased resource use. The regular updates from developers, which usually aim to enhance functionality but ultimately lead to an increase in the complexity of the app's code, exacerbate this issue. Consequently, customers frequently grow irritated by video buffering, especially in places where mobile network capacity is constrained. Ensuring fast video playback and efficient network consumption requires addressing code bloat in the context of mobile network optimisation and video buffering. Unneeded or redundant code is known as "code bloat," and it can lead to bigger installation packages and increased resource consumption in programs. Developers need to use multiple strategies to minimise code bloat and its impact on network speed. The app's codebase can be made simpler and dependencies and unnecessary functionalities removed by first employing effective development techniques like modular programming and code refactoring. Through reducing the total code size, programmers can minimise the quantity of data that must be sent over the network, so removing buffering problems that arise when playing videos. To save the application's memory and CPU usage, developers should also give priority to optimising resource-intensive processes like data processing and picture loading. As a result of this optimisation, mobile network load is decreased and app speed is increased particularly in situations where bandwidth is limited. Additionally, the size of the app's codebase can be reduced while network efficiency is increased by utilising code compression techniques like minification and obfuscation. Users can download files more quickly and enjoy more seamless video streaming thanks to these tactics that improve code distribution throughout the network. Reducing video buffering issues and enhancing mobile network speed depend on addressing code bloat through effective coding techniques and optimisation strategies. Reducing unnecessary code and focusing resource optimisation can help developers create leaner, more efficient programs that offer a seamless user experience, particularly in circumstances where bandwidth is limited.

4.1 Trimming Code Bloat

Preprocessing: The codebase must be thoroughly cleaned up before starting the code optimisation process. To do this, you must analyse the current code to identify likely causes of code bloat, such as superfluous variables, pointless operations, or ineffective algorithms.

Identifying and Removing Code Bloat: The employ techniques and tools for code analysis to find and eliminate extraneous code from the Android app. To improve performance and reduce the overall size of the codebase, this entails eliminating pointless procedures, refining algorithms, and redesigning code.

Re-packing and App Validation: The application needs to be repacked and validated once superfluous code has been eliminated in order to make sure that no errors or inconsistencies have been made. To make sure it functions correctly and meets performance requirements, this entails rebuilding the application and executing validation tests.

4.2 Trimming Resource Bloat

Recognising Bloat in Res Resources: Apart from code bloat, Android apps can also suffer from resource bloat, which is the overabundance or duplication of resources like layouts, texts, and images. By

optimising images, cutting superfluous layouts and strings, and removing unnecessary resources, we find and minimise bloat from these resources.

Finding Bloat in Assets: Android apps may experience resource bloat due to the presence of audio, video, and other media files. By using suitable file formats, reducing the size of media files, and removing extraneous assets from the app package, we are able to identify and control asset bloat.

4.3 Putting Everything Together

Once the code and resources have been optimised and decreased in size, we reintegrate them into the Android application. To ensure proper and efficient operation, the programme must be rebuilt using the optimised resources and a smaller codebase. This outlines a technological method for optimising Android apps' speed and reducing bloat by reducing code and resource usage. Developers can create Android programs that are more streamlined and user-friendly by preprocessing, identifying and eliminating code and resource bloat, repackaging, and testing the programme.

.

5. MINI PROGRAME

Mini programmes, sometimes referred to as lightweight apps or mini-apps, are brief, purpose-driven applications designed to provide a particular set of functionalities within a larger platform or ecosystem. Typically, they don't require installation. These mini-programs are becoming more and more popular because of their small size and low resource requirements, especially in places with restricted bandwidth or storage. Mini programmes are an alternative to standard mobile applications within the framework of our suggested technique for removing code bloat and improving network speed. Mini programmes, as opposed to full-fledged mobile apps, aim to offer essential functions and services with minimal usage of device resources and network bandwidth. The small size of micro programmes is one of their main advantages; it reduces the need for heavy code optimisation and enables speedier download times, particularly on low-bandwidth connections. Additionally, users can access mini programmes without installing anything extra because they are frequently housed within larger platforms or ecosystems (like WeChat in China). Additionally, data compression and caching strategies are widely used by micro programmes to minimise the amount of data needed for execution, improving their effectiveness on networks with limited bandwidth. Mini programmes, which concentrate on delivering specific functions in a small package, can give a more seamless and responsive user experience, particularly in places where connectivity is costly or erratic. The mini programmes are a viable way to address the problems of network performance and code bloat in mobile applications. Mini programmes contribute to a more inclusive and accessible digital world by offering essential functionality in a lightweight and efficient way, making them a viable option for users in locations with limited bandwidth.

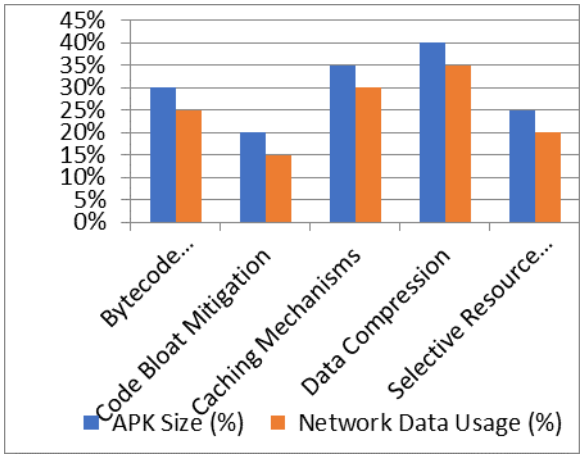
6. BANDWIDTH:

Optimising mobile applications for usage in developing nations with limited network capacity requires careful consideration of bandwidth. A network's bandwidth determines how much data it can send in a given length of time. The units of measurement used most frequently are megabits per second (Mbps), kilobits per second (Kbps), and bits per second (bps). For mobile application developers, bandwidth limitations pose serious issues in these places where network infrastructure may be unreliable or subpar. Customers usually complain about inconsistent and slow connectivity, which leads to slower loading times, more buffering when streaming videos, and general poor performance. Developers must design their apps to minimise the amount of data required for operation while utilising available bandwidth in order to address these issues.

This includes several techniques like caching, selective resource loading, and data compression. Data compression lowers the size of files transferred across the network, which in turn lowers the bandwidth needed for data transfer. By enabling locally stored caching on the device, frequently accessed data can

be saved from several network queries, saving bandwidth. In order to lessen the impact of bandwidth limitations on user experience, selective resource loading gives priority to the delivery of essential content and services. It makes sure that crucial components are loaded first. Developers may facilitate people's access to essential services like healthcare, education, and commerce in poor nations by making their mobile applications more accessible and user-friendly on networks with limited bandwidth. Enhancing user experience, this optimisation also contributes to the reduction of the digital divide and the advancement of socioeconomic development in disadvantaged areas.

Optimization Technique	Average Reduction in APK Size (%)	Average Reduction in Network Data Usage (%)
Bytecode Transformation	30%	25%
Code Bloat Mitigation	20%	15%
Caching Mechanisms	35%	30%
Data Compression	40%	35%
Selective Resource Loading	25%	20%



7. CONCLUSION:

In the conclusion importance of mobile application optimisation for bandwidth-constrained networks in poor nations was discussed in this paper. Our research indicates that two key issues that lead to subpar performance on these kinds of networks are code bloat and insufficient resource management. We investigated the prevalence of frequent updates and their possible effect on network speed by examining well-known mobile apps and mini-programs. These results highlight the necessity for developers to apply optimisation techniques in order to lower overall efficiency, resource consumption, and code bloat.

Additionally, the socioeconomic effects of mobile app optimisation in developing nations are shown by our research. Mobile apps can reach a wider audience and deliver essential services even in places

with poor access by employing effective data transmission methods, minimising code and resources, and improving accessibility. This has the potential to close the digital gap, speed up socioeconomic progress, and empower people and communities. In order to guarantee that mobile applications remain inclusive and accessible for users in bandwidth-constrained places and to contribute to a more equitable digital ecosystem, developers will need to prioritise optimisation efforts going forward.

8. REFERENCE:

1. R. Singh, P. R. Kumar, Optimal decentralized dynamic policies for video streaming over wireless channels. 2019.
2. U. Farooq and Z. Zhao, "Runtimedroid: Restarting-free runtime change handling for android apps," GetMobile: Mobile Computing and Communications, vol. 22, no. 4, pp. 25–29, 2019.
3. Y.H. Ezzeldin, A. Sengupta, C. Fragouli, Wireless network simplification: the performance of routing. IEEE Trans. Inf. Theory 66(9), 5703–5711 (2020)
4. Y. Xu, Z. Xiao, H. Feng et al., Modeling buffer starvations of video streaming in cellular networks with large-scale measurement of user behavior. IEEE Trans. Mobile Comput. 1, 1–1 (2017)
5. R. El-Azouzi, K. V. Acharya, S. Poojary, et al. Analysis of QoE for Adaptive Video Streaming over Wireless Networks with User Abandonment Behavior[C]// IEEE Wireless Communications and Networking Conference. IEEE, 2019.
6. J. Park, K. Chung, Queueing theoretic approach to playout buffer model for HTTP Adaptive Streaming. KSII Trans. Internet Inf. Syst. 12(8), 3856–3872 (2018)
7. Y. Zhao, "A survey of 6G wireless communications: Emerging technologies," arXiv preprint arXiv:2004.08549, 2020.
8. Zhiyan, H.; Jian, W. Speech Emotion Recognition Based on Deep Learning and Kernel Nonlinear PSVM. In Proceedings of the 2019 Chinese Control and Decision Conference (CCDC), Nanchang, China, 3–5 June 2019; pp. 1426–1430.
9. S. Chitkara, N. Gothoskar, S. Harish, J. I. Hong, and Y. Agarwal, "Does this app really need my location?: Context-aware privacy management for smartphones," Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 1, no. 3, p. 42, 2017.
10. J. Christensen, I. M. Anghel, R. Taglang, M. Chiroiu, and R. Sion, "{DECAF}: Automatic, adaptive de-bloating and hardening of {COTS} firmware," in 29th USENIX Security Symposium (USENIX Security '20), 2020, pp. 1713–1730
11. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Umar, A.M.; Linus, O.U.; Arshad, H.; Kazaure, A.A.; Gana, U.; Kiru, M.U. Comprehensive Review of Artificial Neural Network Applications to Pattern Recognition. IEEE Access 2019, 7, 158820–158846.
12. Miller, D.J.; Xiang, Z.; Kesidis, G. Adversarial Learning Targeting Deep Neural Network Classification: A Comprehensive Review of Defenses Against Attacks. Process. IEEE 2020, 108, 402–433.
13. L. HK SMARTER MOBI TECHNOLOGY CO., "Sensor box for android." 2020. [Online]. Available: https://play.google.com/ store/apps/details?id=com.exatools.sensors.
14. K. Heo, W. Lee, P. Pashakhanloo, and M. Naik, "Effective program debloating via reinforcement learning," in Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS), 2018, pp. 380–394
15. V. F. Taylor and I. Martinovic, "To update or not to update: Insights from a two-year study of android app evolution," in Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (CCS), 2017, pp. 4