VIDEO HIGHLIGHT GENERATOR UTILIZING A CONVOLUTIONAL NEURAL NETWORK (CNN) COMBINED WITH A LONG SHORT-TERM MEMORY (LSTM) NETWORK FOR MOBILE LEGENDS: BANG BANG

A Thesis Project
Presented to the Faculty of
Computer Studies Department
College of Science
Technological University of the Philippines
Ayala Blvd., Ermita, Manila

by

ADUVISO, ARABELLA MAE M.

CADUCIO, LUIS POCHOLO

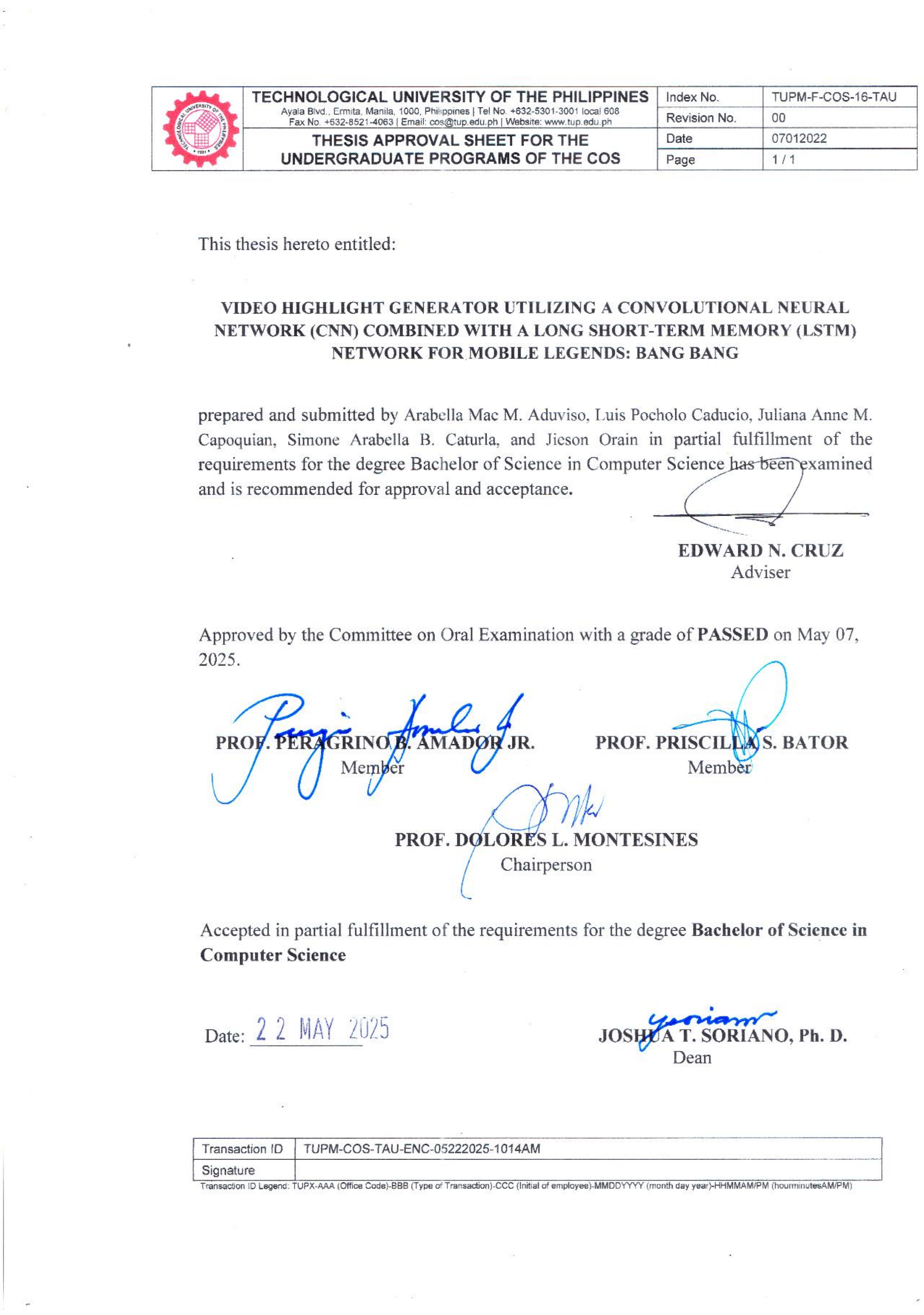
CAPOQUIAN, JULIANA ANNE M.

CATURLA, SIMONE ARABELLA B.

ORAIN, JIESON R.

In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

May 2025



	TECHNOLOGICAL UNIVERSITY OF THE PHILIPPINES Ayala Blvd., Ermita, Manila, 1000, Philippines | Tel No. +632-5301-3001 local 608 Fax No. +632-8521-4063 | Email: cos@tup.edu.ph | Website: www.tup.edu.ph	Index No.	TUPM-F-COS-16-TAU
		Revision No.	00
	THESIS APPROVAL SHEET FOR THE UNDERGRADUATE PROGRAMS OF THE COS	Date	07012022
		Page	1 / 1

This thesis hereto entitled:

VIDEO HIGHLIGHT GENERATOR UTILIZING A CONVOLUTIONAL NEURAL NETWORK (CNN) COMBINED WITH A LONG SHORT-TERM MEMORY (LSTM) NETWORK FOR MOBILE LEGENDS: BANG BANG

prepared and submitted by Arabella Mae M. Aduviso, Luis Pocholo Caducio, Juliana Anne M. Capoquian, Simone Arabella B. Caturla, and Jieson Orain in partial fulfillment of the requirements for the degree Bachelor of Science in Computer Science has been examined and is recommended for approval and acceptance.

EDWARD N. CRUZ
Adviser

Approved by the Committee on Oral Examination with a grade of **PASSED** on May 07, 2025.

PROF. PERAGRINO B. AMADOR JR.
Member

PROF. PRISCILLA S. BATOR
Member

PROF. DOLORES L. MONTESINES
Chairperson

Accepted in partial fulfillment of the requirements for the degree **Bachelor of Science in Computer Science**

Date: 2 2 MAY 2025

JOSHUA T. SORIANO, Ph. D.
Dean

Transaction ID	TUPM-COS-TAU-ENC-05222025-1014AM
Signature	

Transaction ID Legend: TUPX-AAA (Office Code)-BBB (Type of Transaction)-CCC (Initial of employee)-MMDDYYYY (month day year)-HHMMAM/PM (hourminutesAM/PM)

ABSTRACT

To respond to the increase in demand for efficient content creation in Esports, this study presents a Video Highlight Generator for *Mobile Legends: Bang Bang* using two deep learning architectures. The system merges a Convolutional Neural Network (CNN) and a Recurrent Neural Network (RNN) specifically a Long Short-Term Memory Network (LSTM) into a modified Long-term Recurrent Convolutional Network (LRCN) to detect temporal gameplay patterns such as team fights, and YOLOv8 for real-time in-game banner detection with an 80% confidence threshold. Detected banners are assigned excitement scores to prioritize highlight-worthy moments, considering only events lasting at least three seconds to ensure relevance. The system compiles these into a reviewable highlight reel, allowing manual review and arrangement based on priority or chronology. Developed and trained in Google Colaboratory (Colab) and deployed in a Python-based frontend, the system was evaluated across multiple hardware setups and demonstrated high acceptability in the product quality aspect of the ISO 25010. The study validated the effectiveness of the hybrid deep learning model's integration in generating automated and customizable gameplay highlights, providing a significant tool for Esports players, teams, and content creators.

Keywords: *Video Highlight Generation, Deep Learning Architectures, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Long-term Recurrent Convolutional Network (LRCN), YOLOv8, Mobile Legends: Bang Bang*

ACKNOWLEDGEMENT

First, we extend our gratitude to almighty God for blessing us with good health and overall wellness, which were crucial for the successful completion of this study.

We are thankful to Prof. Edward N. Cruz, our research adviser, without whom this project would not have been achievable. His continuous support and active engagement at every level of the process were critical to the study's success. We sincerely appreciate his hard work and cooperation with our research.

We also want to thank our panelists, including Prof. Peragrino B. Amador Jr., Prof. Priscilla Sotelo-Bator and Prof. Dolores L. Montecines. We appreciate their genuine assistance and candid assessment to our study, that enabled us to improve and polish our work.

We are grateful to numerous individuals who contributed more than just academic support to this thesis. We would like to express our heartfelt appreciation to our respondents, particularly IT students and professionals, for their honest and accurate responses to our survey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table **Page**

LIST OF APPENDICES

Chapter 1

THE PROBLEM AND ITS SETTING

Introduction

In the era of massive digital media consumption, the demand for efficient video summarization and highlight generation has become increasingly important. The fast growth of video content hosted on different platforms had given rise to the development of effective methods for summarizing and extracting significant moments, including Esports, sports, entertainment, as well as instructional and professional programming. The system aimed to automatically detect and extract the most interesting or noteworthy moments from videos, providing consumers with a curated summary of the most important events.

This study developed an automated highlight generator tailored for Mobile Legends: Bang Bang, a widely played Esports title. It utilized a hybrid deep learning framework that integrated a Convolutional Neural Network (CNN) for image classification and a Long Short-Term Memory Network (LSTM) for sequence prediction in gameplay (Pardeshi, 2023). While the CNN identified team fights, the LSTM captured temporal dependencies associated with these key events. Using a scoring system based on the excitement level of the detected banners, the system determined which moments should be included in the final highlight video. To ensure accuracy and efficiency, the system imposed an 80% confidence threshold for banner detection and filtered out clips shorter than three seconds. Google Colab was where the models were developed and trained in, offering scalable resources and collaborative capabilities essential for deep learning tasks.

Background of the Study

The demand for efficient video summarization was underscored by the vast reach of online video consumption, with a reported 92.3% of internet users engaging with video content in Q2 2023 (Shepherd, 2024). While gaming videos were among the least popular during this period, sports clips and highlight reels gained considerable attention, aligning with the objective of this research—to extract key moments from Esports videos (Shepherd, 2024).

Esports had rapidly become a mainstream entertainment phenomenon. According to review by Chung et al. (2019), Esports largely affected modern culture, particularly younger audiences. Well-known competitive gaming titles include Dota 2, Valorant, League of Legends, and Mobile Legends have developed professional leagues and major tournaments, drawing millions of viewers. TBy 2024, it was estimated that the worldwide Esports viewership would exceed 577 million, which indicated a 7.7% compound annual growth rate (CAGR) (Alder, 2024). Esports highlight reels have become crucial for fan engagement, providing viewers with the most exciting moments in a condensed format.

Traditionally, creating these highlight reels had required extensive manual effort, with editors sifting through hours of footage to find key moments (Pokharel, 2021). This way of sharing key moments was not only time-consuming but also subjective, as different editors might have varying opinions on what constituted a "highlight" (*Video Highlight Reels: 5 Best Practices for Sharing Insight*, n.d.). Recent developments in AI and ML have shown strong potential in automating processes such as highlight generation. By utilizing these technologies, the highlight generation process could be significantly accelerated and made more consistent.

The system was designed for Mobile Legends: Bang Bang and employed a modified Long-term Recurrent Convolutional Network (LRCN), integrating CNN for detecting team fight visuals with LSTM to model the time-based progression of these events. Once a team fight was recognized by the LRCN, the YOLOv8 analyzed the game to identify banners such as "Triple Kill," "Legendary," and "Unstoppable," integrating these high-impact events into the final highlight reel.

LSTM, a variant of Recurrent Neural Networks (RNNs), is specifically effective at handling long-term dependencies in sequential data. Unlike traditional RNNs, LSTMs introduce memory cells that enable them to hold information over extended periods, making them especially useful for video analysis (GeeksforGeeks, 2025e, 2025f; “Intelligent Prediction of Separated Flow Dynamics Using Machine Learning,” 2024; Khan et al., 2021; Saxena, 2025). In this research, the LRCN identified when a team fight occurred, triggering YOLOv8 to analyze game banners for specific in-game highlights. By recognizing long-term temporal patterns, the LSTM component ensured that generated highlights-maintained continuity and accurately reflected event sequences (Fernandes et al., 2020).

CNNs are deep learning models suited for analyzing grid-structured inputs like video frames, excelling at identifying spatial patterns and structures, using layers of convolution to progressively recognize more complex visual elements. In this research, CNN was used to analyze video segments identified by the LSTM as team fights.

Specifically, YOLOv8 recognized in-game banners and visual indicators like "Triple Kill," "Legendary," and "Unstoppable," ensuring that the system accurately pinpointed high-impact moments. The system used a priority-based scoring mechanism to determine which moments should be included in the final highlight reel, while automatically disregarding non-highlight moments such as resource farming or AFK (Away From Keyboard) periods.

This research adopted the LRCN framework, where CNNs handled visual feature extraction and LSTMs modeled sequential dependencies in gameplay data, making it applicable in video analysis tasks (GeeksforGeeks, 2025f; Uddin et al., 2024). By following this approach, the system effectively extracted both spatial characteristics and time-based patterns, enabling it to detect and highlight significant in-game actions. This combination ensured that the highlight reel maintained both visual quality and narrative coherence, enhancing the overall viewing experience.

For banner recognition, this study employed YOLOv8, an object detection algorithm that balances precision and speed, making it apt for post-processing systems like this highlight generator. Although YOLOv12 was the latest version, there was a minimal difference in performance between YOLOv8 and YOLOv12 for this application, particularly in a non-real-time context. Therefore, YOLOv8’s established reliability and ease of implementation made it the preferred choice for this study.

The system employed the following methods for highlight detection:

- **Player Clustering**: Using LSTM, the system identified areas where at least three players gathered, signaling potential team fights.

- **Objective Events**: The system captured significant in-game events like slaying the Turtle or Lord, which were pivotal to the game’s progress.
- **Banners and Kill Streaks**: The system detected banners visually, such as "Triple Kill" or "Legendary," and prioritized moments of high impact.
- **Final Tower Destruction**: The conclusion of the game, marked by the final tower's destruction, was included in the highlight reel.
- **Exclusion of Non-Highlights**: The system filtered scenes where players were farming or inactive (AFK), as these scenes were not considered exciting or crucial to the outcome of the match.

The research aimed to construct an automated Esports highlight system by integrating CNNs for image recognition and LSTMs for analyzing time-based gameplay sequences. The research streamlined content creation in the Esports industry, this approach enhanced efficiency and consistency relative to conventional manual editing workflows. Beyond Esports, insights from this study may inform improvements in general video editing processes beyond Esports, transforming how video content was curated across various industries. Specifically, this research sought to create an efficient, scalable, and reliable method for producing high-quality, engaging highlight reels for Mobile Legends: Bang Bang.

Objectives of the Study

This project aimed to design and develop an automated highlight generation system tailored for Mobile Legends: Bang Bang using the LRCN framework, where CNNs extracted spatial details and LSTMs handled the chronological modeling of gameplay. The goal was to identify and summarize critical game moments efficiently, streamlining the creation of engaging and high-quality highlight reels. By offering a scalable and consistent alternative to traditional manual editing methods, the system enhanced the viewing experience for Esports audiences.

Specifically, the study’s objectives are to:

1. Develop a video highlight generation system based on the LRCN architecture, integrating CNN for visual recognition and LSTM for identifying and classifying potential team fights, ensuring that the highlights maintained contextual relevance and narrative coherence.

2. Implement YOLOv8 for object detection, focusing on in-game banners such as "Triple Kill," "Legendary," and "Unstoppable." These banners were assigned corresponding scores to prioritize high-impact events in *Mobile Legends: Bang Bang* gameplay footage.

3. Design an event-based system to automatically identify significant gameplay moments, including player clustering, objective kills (such as slaying the Turtle or Lord), and the destruction of the final tower.

4. Ensure that the highlight generation system produced high-quality, engaging video summaries that preserved narrative flow, providing a compelling storytelling experience for Esports audiences.

5. Assess the system’s quality in accordance with relevant ISO 25010 criteria for software evaluation, such as Functional Suitability, Performance Efficiency, Compatibility, Usability, Reliability, Security, Maintainability, and Portability.

Scope and Limitations of the Study

The system was specifically developed for Mobile Legends: Bang Bang (MLBB), a popular multiplayer online battle arena (MOBA) game. Due to this targeted design, the outcomes and insights derived from the study may not be directly applicable to other games or genres. The system integrated a Long-term Recurrent Convolutional Network (LRCN), which merged a Convolutional Neural Network (CNN) and Long Short-Term Memory Network (LSTM), to automatically identify and generate highlights of critical game events such as kills, team fights, and game-changing moments. The LSTM was responsible for identifying when a team fight occurred, triggering the CNN to analyze the corresponding game banners and highlight relevant events. To further enhance visual recognition, the system utilized YOLOv8 for object detection, specifically recognizing in-game banners like "Triple Kill" and "Legendary." The decision to use YOLOv8, rather than the more recent YOLOv12, was due to the minimal performance difference between the two versions in non-real-time applications, making YOLOv8 a more stable and well-documented choice for this implementation. This combination of temporal understanding and visual recognition helped the system prioritize high-impact moments and sequence them logically, maintaining a coherent narrative flow. Non-essential moments, such as farming or being AFK, were automatically excluded to enhance the quality of the final highlight reel.

The system automatically produced a final highlight video reel by concatenating the selected video clips, ensuring a seamless and engaging viewing experience. This approach guaranteed that the final product met quality standards and captured the most engaging gameplay moments.

The system was tailored to fit *Mobile Legends: Bang Bang*, making the findings not applicable in other games or genres. Additionally, it was intended for post-match highlight generation rather than real-time processing, meaning live-streaming scenarios were not supported, and real-time variations were not fully captured. The system also lacked direct integration with social media platforms, requiring manual review and uploading of generated videos by editors, which may have introduced delays in content distribution.

Significance of the Study

Esports Players and Team

This study provided a valuable tool for performance analysis, strategic planning, and content creation by developing an automated Video Highlight Generator. By producing professionally crafted highlight reels, players were able to efficiently review critical moments, pinpoint weaknesses and enhance their strategic approach for future competitive events. Additionally, these highlights served as effective promotional content, assisting players and teams in self-marketing, attracting potential sponsors, and building their online presence.

Esports Organizations

The study validated that integrating CNN and LSTM models yields effective results in gameplay analysis and highlight generation in enhancing audience engagement. By delivering timely, high-quality highlights, organizations were able to attract and retain viewers, increasing both viewership and revenue opportunities. Automated highlight production also enhanced marketing efforts and social media presence, allowing organizations to share engaging content quickly and efficiently. The operational efficiency gained enabled resources to be redirected to other crucial areas, promoting the overall growth of the Esports ecosystem.

Esports Fans and Viewers

This study enhanced the viewing experience by providing high-quality, exciting highlights that captured the essence of the game. It allowed casual fans to stay updated on key events and moments without needing to watch entire matches, broadening the appeal of Esports. The system ensured that even shorter-form content remained engaging and representative of the game's most thrilling and significant moments.

Content Creators and Streamers

This research highlighted the significant benefits of automation in video production with CNNs and LSTMs. The tool streamlined the production of highlight videos, which reduced the technical and time-consuming barriers of manual editing. This allowed creators to focus more on engaging with their audience and produce creative content. The consistent quality of the highlights ensured a better viewer experience, which was crucial for building and maintaining a loyal audience.

Future Researchers

The findings contribute to future research directions in AI and ML, particularly in media automation and hybrid modeling techniques such as LRCN and advanced object detection with YOLOv8. The findings could inform subsequent developments and innovations in the field, providing a model that could be adapted and expanded for other areas of digital content creation.

Chapter 2

CONCEPTUAL FRAMEWORK

This section lays out the study’s conceptual foundation and defines its core terminology. It also brings pertinent academic work and investigation⍰. In cooperation, these components underscore why the researchers believe this study is ⍰important and clarify what it aims to achieve.

Review of Related Literature and Studies

An analysis of the fundamental notions linked to this research. Specifically, topics such as the Video Highlight Generator, Mobile Legends: Bang Bang, CNNs, LSTMs, Ultralytics YOLOv8, Google Colab, and the ISO 25010 standard.

Video Highlight Generator

Automatic video summarization, as explained by Sridevi and Kharde (2020), refers to the production of a short, informative summary of a long video. This technique is especially beneficial for organizing and categorizing videos in large-scale databases. The authors also emphasized the complexity of the task, citing its high difficulty level. In a related observation, Khan and Shao (2022) noted that both amateur creators and professional organizations devote significant time and effort—often amounting to thousands of labor hours—to manually select and edit the most engaging segments from full-length sports broadcasts in order to produce highlight reels. Meanwhile, Kang and Lee (2020) stressed that curating highlight clips—collections of a match’s most crucial moments—is an essential practice in the Esports industry, largely due to the sheer number of matches held daily thanks to the digital and on-demand nature of Esports. In response to this need, Kang and Lee proposed a highlight generation model that detects fluctuations in real-time win-loss probability during a match. Their system, which achieved an accuracy of 89.9%, mirrors the quality of officially produced highlight reels. However, they also pointed out that the effectiveness of the output still depends significantly on the editor's judgment and decision-making when finalizing the clips.

Short-form video is a content style generally characterized by its brief runtime — usually under 60 seconds—and vertical format. However, this standard remains debatable (Oladipo, 2023). Das and Mishra (2024) define it as usually ranging from 15 to 60 seconds in duration. According to Oladipo, despite the variable definitions, most short video platforms support content durations ranging from 30 seconds to about 3 minutes. Utilizing these bite-sized videos can offer a competitive advantage and boost audience engagement (Bretous, 2024). Research conducted on Sina Weibo—the largest microblogging and micro-video site in China—revealed that users tend to favor short videos that run for a few minutes, indicating a clear preference for brief, easily consumable content (Ding et al., 2018). The researchers considered short-form videos and micro-videos specifically their duration in generating highlights.

Mobile Legends: Bang Bang

MLBB features two opposing teams, each consisting of five players, competing to destroy the enemy base while protecting their own. Key mechanics such as team fights, farming for gold and experience, and objectives like securing the Lord contribute to the game’s dynamic nature. As noted by Dao (2024), the game ranks among the most widely played mobile titles globally. A Business World report from November 2020 stated that MLBB had surpassed one billion downloads and had amassed over 100 million active players. (Rossel, 2021). As one of the few mobile games which have managed to cross the 1 billion marks showcasing its immense popularity, Mobile Legends is exclusively focused on by the system designed by the researchers (Ahmed, 2024; Kobis & Tomatala, 2020).

MLBB matches typically span from 10 to 50 minutes and involve players navigating a map split into three lanes. The main objective is to maintain control of the map while simultaneously protecting their own base and targeting the enemy's. The game's core objectives are maintaining map control, defeating opponents, and progressively destroy enemy towers. The destruction of the final tower, or base turret, is a pivotal event in MLBB matches. As the last line of defense for a team's base, its fall often signals the imminent conclusion of the game, marking the climax of the intense combat that has unfolded across the map. This event represents a significant strategic achievement for the attacking team and a critical moment of vulnerability for the defenders, frequently leading to the match's resolution (Academy, 2024; Mawalia, 2020). An article by Singh (2022) explained that throughout the game, various in-game events, such as kills, assists, and objective captures, contribute in determining the outcome of the match. Once a team successfully destroys the opposing team's base, the final banner, either “Victory” or “Defeat”, signals the end of the game. This banner not only marked the conclusion but also displayed the match's statistics, including key performance metrics such as kills, deaths, assists, and gold earned. Additionally, the game identifies the Most Valuable Player (MVP) from each team, based on individual contributions throughout the match. In the researchers' system, this final banner acted as a trigger point, signaling the termination of highlight detection and focusing on post-game analysis.

Convolutional Neural Network (CNN)

Dawood (2023) described Convolutional Neural Networks (CNNs) as a type of deep learning inspired by the human visual system. CNNs consist of layers, such as convolutional, pooling, and fully connected layers, that work in tandem to analyze visual data. Paper by Feddewar and Deshmukh (2022) stated that the classical CNN is used for image processing. Feddewar and Deshmukh (2022) also stated that after seeing the performance results of CNN on images, CNN was used for video processing.

Karpathy et al. (2014) presented a comprehensive study assessing CNN performance for large-scale video classification. Their dataset comprised one million YouTube videos across 487 categories. To speed up training without compromising accuracy, they proposed a multiresolution, foveated model architecture that utilizes two spatial resolution streams. This hybrid architecture proved to be effective. For future developments, Karpathy and colleagues suggested expanding the dataset to encompass more general features, developing models that factor in camera movements, and exploring the potential of recurrent neural networks for refining global predictions derived from clip-level data.

In studying Esports highlight generation, using classification models to detect elements like banners are essential for identifying key gameplay moments. Robb et al. (2017) reported that 42% of participants considered visual cues, including character-hit flashes and health bar changes, key indicators of significant in-game events. These fluctuations highlight critical moments like kills, assists, or high-stakes engagements, which are ideal for inclusion in highlights due to their exciting and climactic nature, crucial for capturing and holding viewer attention. Zhou and Forbes (Fadelli, 2022) analyzed the use of visual effects (VFX) in video games and identified three major functions, one of which is to signal critical gameplay moments. These moments include significant in-game actions like an enemy attack, a timer running out, or transitions to a new game phase (Fadelli, 2022).

In Mobile Legends: Bang Bang, one of its most prominent events is a team fight, which is visually marked by its features. The researchers will utilize CNNs to detect team fights, as they are clear indicators of highlight-worthy moments.

To build on the foundation laid by prior research, this study proposed a system that:

- Suggests potential highlight clips for editors instead of generating final highlight videos;
- Integrates Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) models, can be used to aggregate segmented video insights and produce an overarching prediction based on the entire gameplay footage;
- Leverages CNN-based video recognition to identify key gameplay events such as team fights.

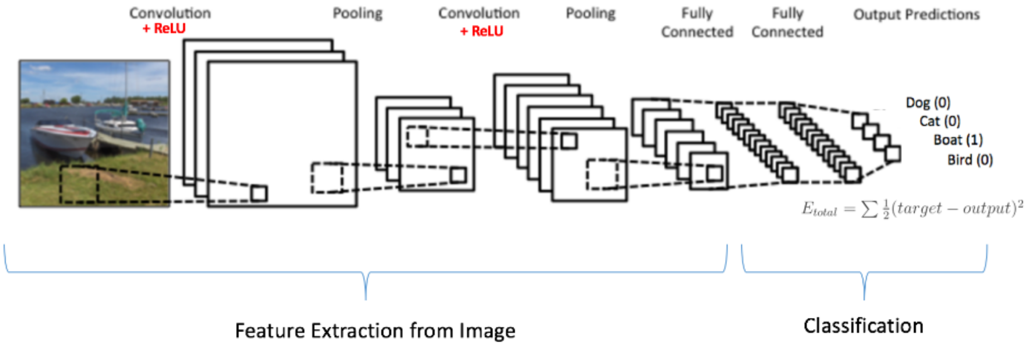
Team fights are identified by recognizing patterns in which three or more players cluster together—a common indicator of significant in-game engagements. By applying convolutional operations across multiple frames simultaneously, the CNN is trained to detect spatial features, while deeper layers build upon these to recognize higher-level representations.

To capture the temporal dynamics of these sequences, the study incorporates LSTM networks. The TimeDistributed layer in Keras (Anwaar, 2020) enabled the model to apply CNN operations across time, preserving the temporal structure of the video data while learning spatial features frame-by-frame.

CNNs, with their stacked layers—convolutional, pooling, and fully connected—allow for effective spatial analysis of in-game visuals. These spatial features are essential for detecting kill banners and team fights. Meanwhile, LSTMs enhanced the system's capacity to understand how these events evolve over time, making both CNNs and LSTMs vital components for automated highlight generation in Esports. This study will further elaborate on the internal structure of CNNs.

Figure 1

CNN Architecture

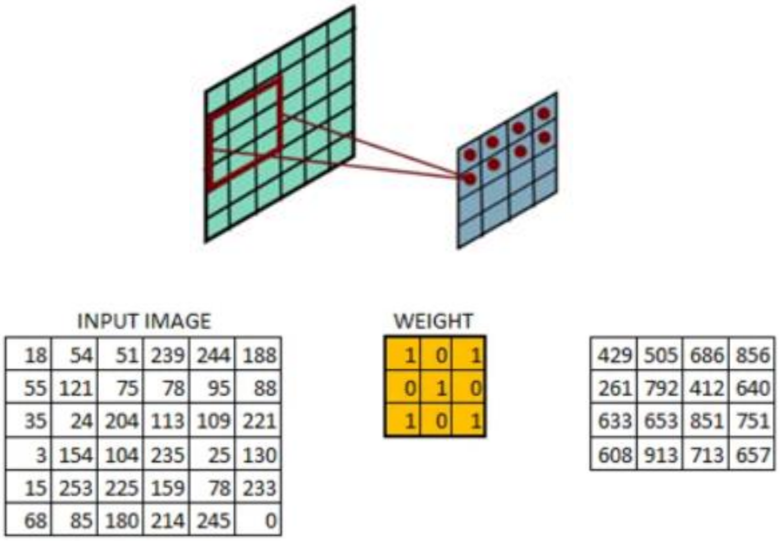


Note. Convolutional Neural Network (CNN) architecture showed the layers where feature extraction from image and classification happen. Adapted from *A simple CNN model beginner guide* by S. Sanagapati, 2020, *Kaggle*. In public domain.

The figure above is the common architecture of CNN; some known modified CNN structures are LeNet and VGG16. According to Freenergi (2019), CNNs are especially useful in processing images because they are inspired by the human's visual cortex. In object classification tasks, CNNs extract a feature matrix and use Softmax probability functions to predict whether an image fits into a specific category. In the system, CNN is utilized to detect team fights. The CNN model will use the input data's video frames.

Figure 2

Convolutional Layer



INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

WEIGHT

1	0	1
0	1	0
1	0	1

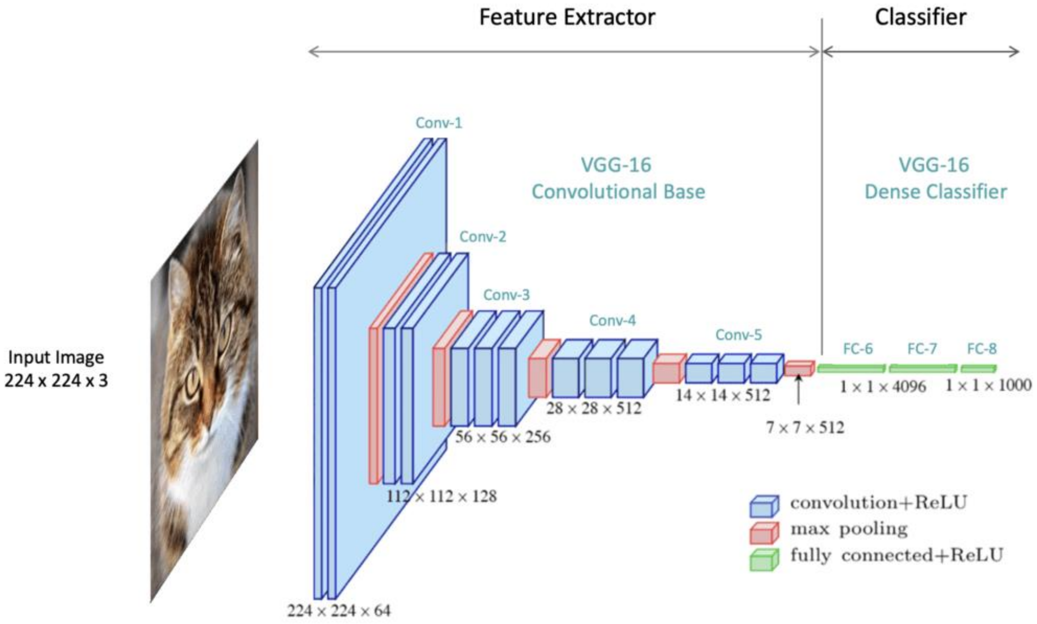
429	505	686	856
261	792	412	640
633	653	851	751
608	913	713	657

Note. This figure illustrated the convolutional operation using a kernel (or weight matrix) over an input image resulting in a convolved feature map matrix, the specific details extracted from the image. The top part showed the concept diagrammatically, while the bottom part presented a numerical example of convolution. Adapted from *Convolutional neural network for object recognition and detection*, by A. Freenergi, 2021, *Medium*, and *A simple CNN model beginner guide* by P. Sanagapati, 2020, *Kaggle*. In the public domain.

Mishra (2020) emphasized that the convolutional layer plays a central role in CNNs, shouldering much of the computational workload. This layer produces a feature or activation map by using a kernel—a set of learnable weights—to process the input matrix. The kernel, labeled as weight in the bottom picture of the figure, is more in-depth but less in space.

Figure 3

VGG-16 CNN architecture



Note. The figure illustrated the VGG-16 convolutional neural network architecture, separating the feature extractor and classifier components. It highlighted how input image dimensions (224 × 224 × 3) are transformed through successive convolutional, pooling, and fully connected layers. The depth (number of channels) increases while spatial dimensions are reduced as the image passes through the network. Adapted from *Convolutional Neural Network: A complete guide*, by B. Kromydas, 2023, *LearnOpenCV*. In the public domain.

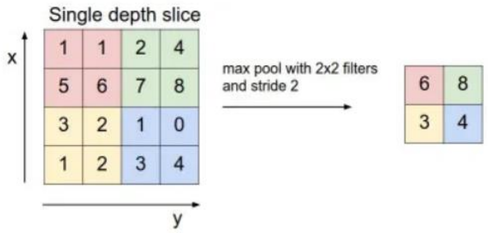
As the data progresses through the network, its spatial dimensions decrease while its depth increases. This depth corresponds to the number of data channels. For example, the VGG-16 CNN model is built to take colored images with an input size of 224x224 pixels and three-color channels (representing RGB) (Kromydas, 2023).

The depth can fill up to three channels, yet its height and width is less than the matrix it is used for. Kernels are only applied where the filter fully fits on top of the matrix. Weights are the training phase which involves the network automatically determining the precise values for its kernels in underlying patterns to a reliable prediction (Beuzen, n.d.).

Sanagapati (2020) further explained that the kernel functions as a filter that extracts specific details from an image by sliding over the input. The kernel slides from the image's top left pixel until it reaches the bottom pixel. The amount it shifts at each step is called the stride. When utilizing many convolutional layers, the feature maps become more complex as the network gets deeper, with the first layer extracting more generic information as discussed by Sanagapati.

Figure 4

Pooling Layer



Note. This figure demonstrated the max pooling operation with 2×2 filters and a stride of 2. Pooling layers reduce the spatial dimensions of an input while preserving depth, as each depth slice is processed independently. Max pooling selects the maximum value from each region of the input matrix. Adapted from *Convolutional neural network for object recognition and detection*, by A. Freenergi, 2021, *Medium*. In the public domain.

Pooling operations help minimize the image's spatial dimensions while retaining the depth. Among various types, max pooling is the most commonly used. (Sanagapati, 2020). According to Mishra (2020), this is done by *"replacing the output of the network at certain locations by deriving a summary statistic of the nearby outputs"*. Each slice of the representation is treated independently during the pooling operation.

The concluding stage of CNN thru the connected layer, as the name suggests, the neurons from the preceding and succeeding layers are fully connected with each other. The job is performed by neurons in an associated layer, that filters in convolutional network.

Unlike dense layers, where every neuron connects to all previous outputs, convolutional layers link each neuron only to a localized area of the preceding layer. In pooling, each neuron or node connects only to a certain location of the input (Deepai, 2024; GeeksforGeeks, 2024a). Before reaching the dense layer, the network reshapes the multidimensional data into a single-dimension format suitable for final classification or prediction tasks (Freenergi, 2019). This is computed using the standard method of matrix multiplication with bias effect. Finally, the prediction or assigning of classification of the image will be achieved using activation function like Softmax.

Long Short-Term Memory Network (LSTM)

Long Short-Term Memory (LSTM) is a widely adopted variant of the Recurrent Neural Network (RNN), primarily utilized for handling sequential or temporal data (Sharma et al., 2021). According to Sharma et al., LSTM has emerged as a leading algorithm in video processing studies due to its capability to model long-range dependencies effectively. Taking advantage of the temporal nature of videos, actual video classification task result can be improved using more advanced neural network architectures such as LSTM (iMerit, 2022). Cited in Sharma et al. (2021), Ibrahim et al. (2016) introduced a two-stage deep temporal framework grounded on LSTM to recognize group activities. Their approach involved one LSTM network to capture individual-level action dynamics over time and a second LSTM to synthesize these person-level cues into an overall activity understanding. Evaluation against existing methods on datasets like the Collective Activity Dataset and a volleyball dataset showed that their model surpassed previous baselines in performance.

A study from Benoughidene and Titouna (2021), showcased the effectiveness of LSTM for video processing, who also proposed a novel approach combining CNN and LSTM for automatic sports video classification. Their model utilized the Inception V3 network, which extracts low, middle, and high-level spatial features from video frames. After extracting the spatial details, an LSTM model works to check their time-based behaviors more accurately in video classification. Benoughidene and Titouna evaluated their model using the UCF Sports Dataset, achieving superior results over traditional baseline models. Their findings highlighted the effectiveness of integrating CNNs for spatial feature learning with LSTMs for temporal analysis, which is critical for tackling complex video classification tasks.

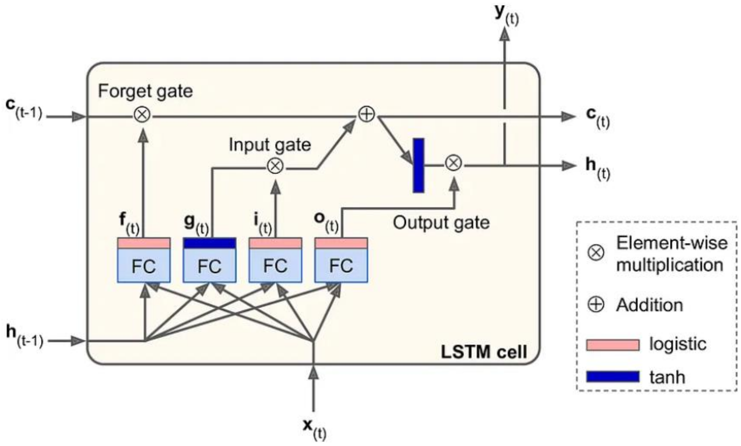
Similarly, in this research, LSTM will be employed to capture temporal information in Mobile Legends: Bang Bang, ensuring better identification of relevant actions within sequences involving three or more players. The researchers try to fill the gaps of the related studies by:

- Employing an LSTM network to capture temporal information in MLBB which is the focus of the system, ensuring a better identification of relevant actions within sequences involving three or more players.

LSTM models are particularly tailored to manage the temporal characteristics of sequential data. Their design includes memory cells and gating mechanisms that allow them to retain or discard information from previous time steps, enabling them to handle long-term dependencies in video sequences (Anishnama, 2023). Below is a breakdown of the LSTM architecture.

Figure 5

LSTM Architecture



Note. The diagram illustrated the structure of a Long Short-Term Memory (LSTM) cell, including its three key gates: the input gate, forget gate, and output gate. These gates regulate the flow of information across time steps in a recurrent neural network. Element-wise operations, logistic functions, and tanh activations manage memory and state updates. Adapted from *Understanding LSTM: architecture, pros and cons, and implementation*, by Anishnama, 2023, *Medium*. In the public domain.

Anishnama suggested that LSTM's architecture can be seen as a sequence of recurrent "blocks" or "cells," each of which has a group of connected nodes. An LSTM cell composes of Input gate, Forget gate, and Output Gate:

- The input gate filters incoming data to decide what to store in memory.
- The forget gate determines which pieces of previously stored data should be discarded.
- The output gate manages the information passed to the next time step and the final output.

In the system, LSTM is utilized to process the sequence of time frames in the input video data, enabling the ability to assess if the sequence of time frames is classified as a highlight. CNN will initially detect a specific time frame through the intensity of game banner, suggesting for team fights and possible game highlights. The time frame will be passed through the LSTM, checking the previous and the following video frames of the initially detected highlights. Furthermore, CNN and LSTM can be modified to accommodate the system project’s objectives with optimization in mind.

The researchers also recognized the element, notifications specifically kill notifications, and the final “Victory” or “Defeat” banners utilizing YOLOv8. Killing notifications and the final banner are considered as signals of important in-game events and the end of the game (Robb et al., 2017). Considering killing notifications and the final banner or game banners, it is possible to determine significant events in the initial highlight segments made possible through CNN and LSTM.

Long-term Recurrent Convolutional Network

The Long-term Recurrent Convolutional Network (LRCN) is a composite deep learning architecture that merges CNNs for spatial feature extraction and LSTMs for modeling temporal sequences. This combination is especially suited for analyzing spatiotemporal data, making it applicable in areas like video classification, activity detection, and automated video captioning (Uddin et al., 2024).

As highlighted by Dofitas et al. (2024), the LRCN architecture was designed to overcome a key limitation of traditional CNNs: while CNNs can extract spatial information from individual frames, they cannot account for the temporal dependencies that exist in video sequences. By integrating LSTM layers after the CNN component, the LRCN is able to model the progression of events over time. This combination allows the system to interpret not only what is happening in a single frame but also how the scene evolves across multiple frames—an essential aspect for applications in video analysis (Khalil et al., 2025).

In video-based applications like action recognition and highlight detection, the LRCN's dual-layer design enabled frame-level feature extraction using CNNs, followed by sequence learning through LSTMs. This approach has been proven effective in tasks that require understanding of motion, progression, and context over time (Uddin et al., 2024).

In support of this, Pyun et al. (2023) explored how audiences engage with Esports highlights, particularly in the context of the League of Legends Champions Korea (LCK). Their study identified several key factors that drive highlight viewership, including the timing of major game events (e.g., team fights and objectives), the popularity of players or teams involved, and the narrative significance of a given moment in the match. While their research focused on viewer behavior rather than machine learning models, it provided valuable insight into which in-game moments are considered "highlight-worthy." These findings emphasize the importance of aligning machine-generated highlights with actual viewer interests—an area where LRCNs can play a key role by modeling not only the visual content but also its temporal and contextual importance.

Lohokare et al. (2020) further demonstrated the relevance of LSTM-based models in Esports by developing deep learning agents for a simplified version of League of Legends. Their study had implemented two agents: one trained using Reinforcement Learning and another using a Supervised LSTM Network. While their focus was on creating intelligent agents rather than highlight generation, their findings support the idea that LSTMs are effective in modeling the complex, sequential patterns found in

Multiplayer Online Battle Arena (MOBA) games. Given the structural and gameplay similarities between League of Legends and Mobile Legends: Bang Bang (MLBB), their work reinforced the potential of LRCN models in automating gameplay understanding tasks such as highlight detection.

One of the key strengths of LRCNs in highlight generation lies in their ability to distinguish between high-impact and low-impact moments by evaluating not just visuals but also the sequence in which events unfold. For example, CNNs might incorrectly highlighted farming scenes simply because of flashy animations or bright colors. However, with the temporal awareness provided by LSTMs, the model can recognize that these moments lack significant game progression or impact, thus avoiding false positives.

Despite their advantages, LRCNs are not without limitations. As discussed by Waqas and Humphries (2024), recurrent models such as LSTMs required significant computational resources and may struggle with modeling very long sequences due to issues like vanishing gradients. However, in the context of Esports highlight generation—where the clips are often short—LRCNs remain an effective and practical choice.

Activation Functions

Activation functions are essential components in neural networks that apply non-linear transformations to inputs, enabling the network to learn intricate data relationships (GeeksforGeeks, 2025). Without them, even multilayered networks would behave as linear models with limited representational power.

Frequently used activation functions include:

- Sigmoid, which compresses inputs into a range between 0 and 1 and is frequently applied in binary classification tasks;
- Tanh, which scales values between -1 and 1 and is centered at zero;
- ReLU (Rectified Linear Unit), which outputs the input if it is positive and zero otherwise, making it highly efficient for hidden layers;
- Softmax, which is commonly used in the final layer of classification networks to normalize output probabilities across multiple classes.

Choosing the right activation function greatly influences a model's training efficiency and overall performance.

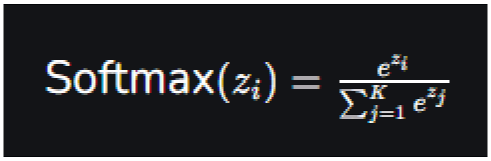
Softmax

The Softmax activation function is specially designed for classification problems involving multiple classes. It transforms the raw output values, or logits, from the final layer into a probability distribution, ensuring that the sum of all output probabilities equals one. This probabilistic interpretation is essential when the model needs to predict the most likely class among several possibilities (GeeksforGeeks, 2024).

Mathematically, the Softmax function is defined as:

Figure 6

Softmax Activation Function



$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Note. The diagram illustrates the Softmax activation function, which transforms a vector of raw scores (logits) into probabilities that sum to 1. Adapted from *Softmax activation functions in Neural Networks*, by GeeksforGeeks, 2025. In the public domain.

As noted by GeeksforGeeks (2024), Softmax not only aids in classification tasks but also works optimally with the cross-entropy loss function, providing a robust gradient signal during backpropagation. This synergy enhances the training process's stability and effectiveness.

However, Softmax has limitations. One notable concern is its sensitivity to large input values, which can cause numerical instability. To improve numerical stability during computation, it is standard to subtract the maximum logit value before applying the Softmax function—a method referred to as the log-sum-exp trick.

Softmax has seen wide application in models such as CNNs and LSTMs in areas including image classification, natural language processing, and video analysis, where it helps in interpreting which event or category a frame or sequence most likely belongs to (Alzubaidi et al., 2021).

Adam Optimizer

The Adam (Adaptive Moment Estimation) optimizer is a widely adopted optimization algorithm in deep learning, particularly valued for its adaptive learning capabilities and efficient performance on large and complex datasets. It computes individual adaptive learning rates for different parameters by maintaining exponentially decaying averages of past gradients (first moment) and squared gradients (second moment), which helps in stabilizing and accelerating the convergence process (GeeksforGeeksg, 2024). Adam is designed to combine the benefits of both AdaGrad, which adapts learning rates for each parameter, and RMSProp, which adjusts rates based on recent gradients. Its bias correction mechanism ensures reliable gradient updates in the early training phases. Due to its low memory requirements and ease of implementation, Adam has become a default optimizer choice for training models that utilize activation functions like ReLU and Softmax across tasks such as image recognition, natural language processing, and video classification.

Ultralytics YOLOv8

Ultralytics developed YOLOv8, one of the evolutions in the YOLO (You Only Look Once) object detection series. It delivers state-of-the-art accuracy and speed, suitable for various computer vision applications such as object detection, classification, segmentation, tracking, and pose estimation. YOLOv8, a CNN-based model, can classify visual content in images by assigning them to predefined categories (Hindarto, 2023; Ultralytics, 2024). A study shows that adding a specified detection head can improve the performance of YOLOv8, finding small objects as small as 4*4 pixels (Huang et al., 2024).

In this study, the researchers explored the use of YOLOv8 for object classification, specifically for detecting in-game banners in Mobile Legends: Bang Bang. These banners signaled significant gameplay events and are crucial for identifying key moments. By applying YOLOv8 to video segments, the system detected and classified game banners and suggested corresponding clips for editors to include in the final highlight reel.

Model Evaluation

Assessing the performance of a deep learning model is vital to ensure its ability to generalize to unseen data and fulfill task-specific objectives. Performance metrics help determine whether the model's predictions align closely with expected outcomes.

Accuracy

Accuracy, one of the most basic evaluation metrics, represents the proportion of correctly predicted instances out of the total predictions made.

Figure 7

Accuracy Formula

$$Accuracy = \frac{True\ Positive\ +\ True\ Negative}{True\ Positive\ +\ True\ Negative\ +\ False\ Positive\ +\ False\ Negative}$$

Note. The formula presented the formula for computing Accuracy in classification problems. Accuracy is defined as the ratio of correctly predicted observations (True Positives and True Negatives) to the total number of predictions. It is a fundamental evaluation metric in machine learning used to assess the overall correctness of a model's predictions. Adapted from *Accuracy, Precision and Recall*, by PadhaiTime. In the public domain.

Although accuracy is straightforward, it may not reflect true performance in imbalanced datasets. A model can achieve high accuracy by favoring the dominant class, even while underperforming on minority classes.

Precision, Recall, and F1 Score

In scenarios with class imbalance, using additional metrics such as Precision, Recall, and the F1 Score provides a more comprehensive picture of a model's effectiveness.

- Precision is the proportion of correctly predicted positive instances out of all instances predicted as positive:

Figure 8

Precision Formula

$$Precision = \frac{True\ Positive}{True\ Positive\ + False\ Positive}$$

Note. The formula illustrated the formula for Precision in classification problems, which is the ratio of True Positives to the sum of True Positives and False Positives. Precision evaluated the accuracy of positive predictions made by the model. Adapted from *Accuracy, Precision and Recall*, by PadhaiTime. In the public domain.

- Recall, also termed Sensitivity or the True Positive Rate, measures how many actual positive instances were accurately identified by the model:

Figure 9

Recall Formula

$$Recall = \frac{True\ Positive}{True\ Positive\ +\ False\ Negative}$$

Note. The formula presented the formula for Recall (or Sensitivity), which is the ratio of True Positives to the sum of True Positives and False Negatives. Recall measures the model's ability to identify all positive instances. Adapted from *Accuracy, Precision and Recall*, by PadhaiTime. In the public domain.

- The F1 Score calculates the harmonic mean between Precision and Recall, offering a balanced measure especially useful when dealing with imbalanced data:

Figure 10

F1 Score Formula

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Note. The formula illustrates the F1 Score, which is the harmonic mean of Precision and Recall. It balances the trade-off between the two metrics, offering a single measure that captures both the model's ability to avoid false positives and its ability to detect positive instances. Adapted from *F1 score*, by Encord. In the public domain.

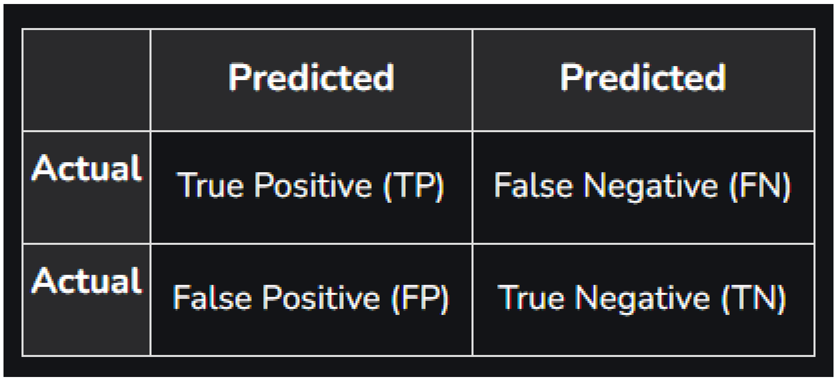
This metric is particularly advantageous in cases where both false positives and false negatives carry significant implications, such as in Esports highlight detection, where missing crucial moments can be more detrimental than including less relevant ones.

Confusion Matrix

A Confusion Matrix provides a structured visualization of how a classification model performs by comparing predicted and actual labels in a tabular format. For binary classification, the confusion matrix looks like this:

Figure 11

2x2 Confusion Matrix



	Predicted	Predicted
Actual	True Positive (TP)	False Negative (FN)
Actual	False Positive (FP)	True Negative (TN)

Note. A 2×2 Confusion Matrix was used to visualize classification performance, showing the counts of True Positives, False Positives, True Negatives, and False Negatives, thereby providing a deeper understanding of the model's predictive behavior. Adapted from *Understanding the Confusion Matrix in Machine Learning*, by GeeksforGeeks, 2025. In the public domain.

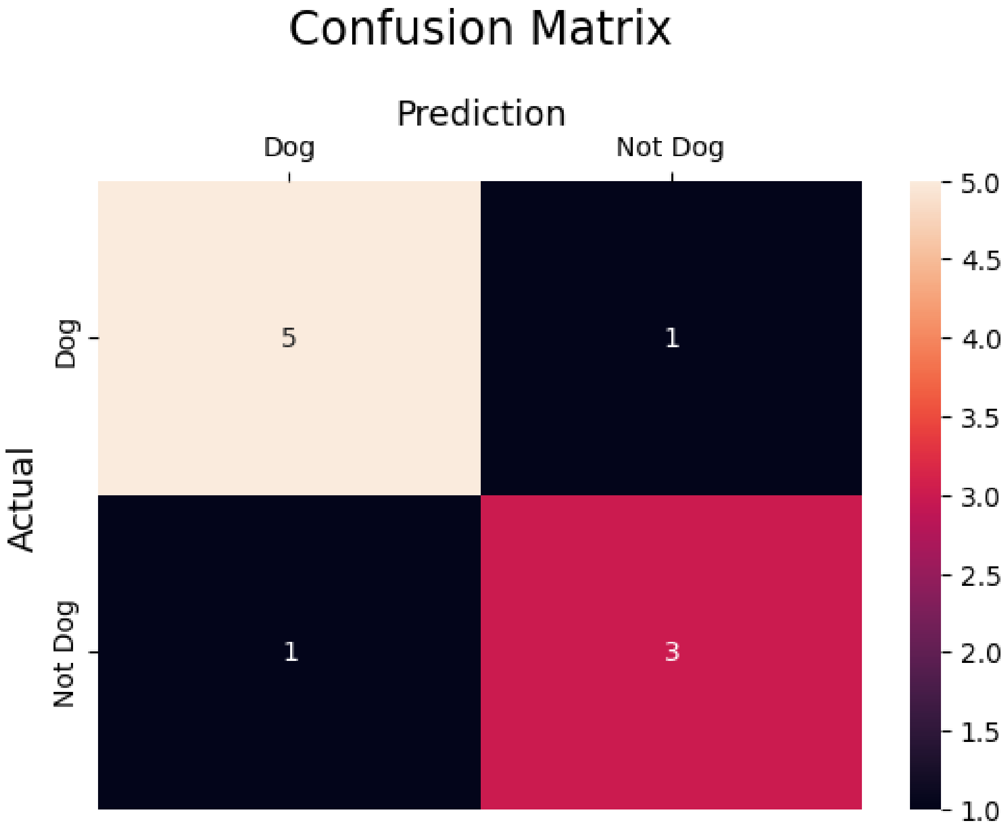
Each element of the matrix is interpreted as follows:

- True Positives (TP): Correctly predicted positive instances.
- True Negatives (TN): Correctly predicted negative instances.
- False Positives (FP): Negative instances incorrectly predicted as positive (Type I error).
- False Negatives (FN): Positive instances incorrectly predicted as negative (Type II error).

An example of confusion matrix:

Figure 12

Example of Confusion Matrix



Note. The diagram is a confusion matrix visualizing a binary classification model's performance in identifying "Dog" and "Not Dog" classes. It showed 5 true positives (Dog correctly predicted as Dog), 3 true negatives (Not Dog correctly predicted), 1 false positive (Not Dog misclassified as Dog), and 1 false negative (Dog misclassified as Not Dog). The matrix helps in understanding the distribution of prediction errors and correct classifications. Adapted from *Understanding the Confusion Matrix in Machine Learning*, by GeeksforGeeks, 2025. In the public domain.

The confusion matrix is also applicable to multi-class classification tasks, providing deeper insights into model performance, particularly in datasets with class imbalance (Bharathi, 2025).

Loss Function

Process of refining machine learning models relies heavily on objective functions, which precisely gauge the variance between a model's projected outputs and the data points. A well-chosen loss function not only accelerates training convergence Also enhances the model's capability to perform well on unseen data. Entropy, a concept from information theory, captures uncertainty in probability distributions. Building on this, cross-entropy measures the divergence between predicted probabilities and actual labels, commonly used in classification problems.

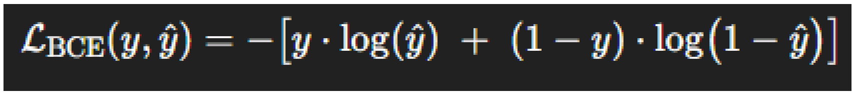
Cross-Entropy Loss comes in two main flavors:

1. Binary Cross-Entropy (Log Loss)

   Used for binary classification (e.g., logistic regression or a two-class neural network with a sigmoid output). It penalizes the deviation of the likelihood estimation against the actual two – category classification:

   Figure 13

   Binary Cross-Entropy Formula



$$\mathcal{L}_{\text{BCE}}(y,\hat{y}) = -\left[y \cdot \log(\hat{y}) \; + \; (1-y) \cdot \log\left(1-\hat{y}\right)\right]$$

Note. The equation represented the Binary Cross-Entropy (BCE) loss, commonly used in binary classification tasks. Here, $y \in \{0,1\}$ denoted the true class label, and $\hat{y} \in (0,1)$ is the predicted probability for the positive class. The BCE loss penalized the divergence between predicted probabilities and actual binary outcomes. Adapted from *Loss Function in Machine Learning*, by DataCamp, 2024. In the public domain.

2. Categorical Cross-Entropy

   Used for multi-class classification with one-hot labels and a Softmax output. It generalizes binary cross-entropy to C classes:

Figure 14

Categorical *Cross-Entropy Loss Formula*

$$CE = -\sum_{i}^{C} t_i log(f(s)_i)$$

Note. The formula represented the multi-class Cross-Entropy (CE) loss, used in classification problems involving more than two classes. C is the total number of classes, $y_i \in \{0,1\}$ is the one-hot encoded true label for class i, and $f(s)_i$ is the predicted probability for class i, typically computed using the softmax function. This loss measured the dissimilarity between the true distribution and the predicted distribution. Adapted from *Cross Entropy Loss*, by Gombru, 2018. In the public domain.

Google Colaboratory

Google Colaboratory, more commonly known as Colab, is a cloud-based platform built upon Jupyter Notebooks that simplifies the teaching and dissemination of machine learning research. One of its most notable features is that it requires no installation, allowing users immediate access to high-performance computing resources such as GPUs (Graphics Processing Units) and TPUs (Tensor Processing Units) at no cost. Colab is widely recognized for significantly improving the speed and efficiency of training deep learning models, and it also supports a broad spectrum of GPU-intensive scientific computations. In studies that assessed Colab’s performance, it was observed that training convolutional neural networks (CNNs) on Colab's accelerated runtime yielded training speeds that were nearly twice as fast as utilizing all physical cores of a traditional Linux server (Carneiro et al., 2018; Google Colab, n.d.). Given this result, the researchers used Colab to train and develop a CNN and a LSTM network, integrated through the LRCN architecture.

Roboflow

Roboflow has become one of the leading platforms for accessing open-source datasets and tools tailored to computer vision. It serves as a collaborative hub where practitioners contribute to and utilize a vast archive of annotated image datasets. As of recent counts, over 60 million public images and around 90,000 datasets are actively being labeled and improved upon through the platform’s web-based tool, Roboflow Universe (Ciaglia et al., 2022; Roboflow, 2025). Derived from this effort is Roboflow 100 (RF100), an open-source and community-driven benchmark designed specifically for evaluating the performance of object detection models (Francesco, 2024; *YOLOv8 Object Detection Model: What Is, How to Use*, 2025). The researchers assembled and labeled the Banner Dataset on the external service Roboflow API to be used by YOLOv8 in Banner Detection.

ISO 25010

The ISO/IEC 25010 standard, often abbreviated as ISO 25010, outlines a comprehensive model for assessing the quality of software products, such as the system developed in this study. This international standard serves as a structured framework to evaluate key software quality characteristics, including but not limited to functional performance, usability, maintainability, and security (QMII, 2023; Robbins, 2023). According to QMII, the model supports organizations in identifying strengths and improvement areas across eight major quality dimensions, which are further broken down into detailed sub-characteristics.

1. Functional Suitability

Refers to the software's ability to meet both stated and implied needs.

a. Completeness - Coverage of all required functions and objectives.

b. Correctness - Accuracy of results with the expected level of precision.

c. Suitability or Appropriateness - Alignment of functions with user goals.

2. Performance Efficiency

Measures how well the system performs relative to the resources it consumes.

a. Time Behavior - Responsiveness and processing speed.

b. Resource Utilization - Efficiency in using hardware/software resources.

c. Capacity - System's ability to handle specified workload limits.

3. Compatibility

Assesses how well the system integrates or coexists with other systems.

a. Co-existence - Ability to function without causing disruption in a shared environment.

4. Usability

Determines how easily users can operate and learn the system.

a. Appropriateness Recognizability - Ease of identifying system relevance to user needs.

b. Learnability - Ease with which users become proficient in system use.

c. Operability - Simplicity of controlling the system.

   d. User Error Protection - Safeguards against user mistakes.
   e. User Interface Aesthetics - Visual appeal and user satisfaction.
   f. Accessibility - Inclusiveness for users with varying capabilities.

5. Reliability

   Evaluates consistency in performance over time and under specific conditions.

   a. Maturity - Dependability during normal operations.
   b. Fault Tolerance - Continuity in functionality despite component failures.

6. Security

   Focuses on safeguarding data and preventing unauthorized access or modifications.

   a. Integrity - Protection against data tampering or unauthorized manipulation.

7. Maintainability

   Represents the ease with which the software can be updated or modified.

   a. Modifiability - Flexibility to implement changes without introducing new errors.
   b. Modularity - Independence of components to limit the impact of changes.

8. Portability

   Refers to how easily the system can be moved across different environments.

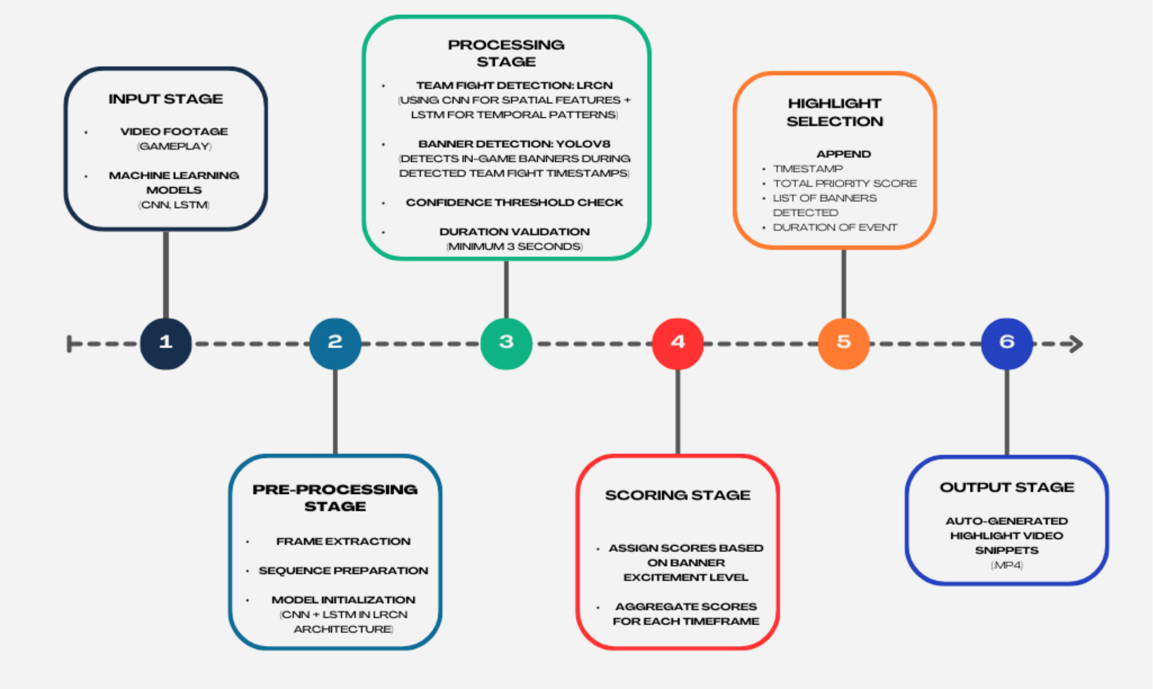
   a. Installability - Ease of installing or uninstalling the software on various platforms.

To validate the applicability of ISO 25010, Canlas et al. (2021) conducted a study involving 10 domain experts from both the IT industry and academia. Their research confirmed the standard’s relevance, reporting content validity indices ranging from 90.00% to 100.00%, which aligns with the acceptable thresholds based on Lynn’s criteria. The overall scale validity ranged from 96.67% to 100.00%, confirming the reliability of ISO 25010 as a quality assessment framework. In this study, the same ISO 25010 characteristics will be employed to evaluate the system’s performance across all eight dimensions: functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability.

Conceptual Model of the Study

Figure 15

Conceptual Model


INPUT STAGE
VIDEO FOOTAGE (GAMEPLAY)
MACHINE LEARNING MODELS (CNN, LSTM)
PROCESSING STAGE
TEAM FIGHT DETECTION: LRCN (USING CNN FOR SPATIAL FEATURES + LSTM FOR TEMPORAL PATTERNS)
BANNER DETECTION: YOLOV8 (DETECTS IN-GAME BANNERS DURING DETECTED TEAM FIGHT TIMESTAMPS)
CONFIDENCE THRESHOLD CHECK
DURATION VALIDATION (MINIMUM 3 SECONDS)
HIGHLIGHT SELECTION
APPEND
TIMESTAMP
TOTAL PRIORITY SCORE
LIST OF BANNERS DETECTED
DURATION OF EVENT
1
2
3
4
5
6
PRE-PROCESSING STAGE
FRAME EXTRACTION
SEQUENCE PREPARATION
MODEL INITIALIZATION (CNN + LSTM IN LRCN ARCHITECTURE)
SCORING STAGE
ASSIGN SCORES BASED ON BANNER EXCITEMENT LEVEL
AGGREGATE SCORES FOR EACH TIMEFRAME
OUTPUT STAGE
AUTO-GENERATED HIGHLIGHT VIDEO SNIPPETS (MP4)

The conceptual model for the automated Video Highlight Generator for Mobile Legends: Bang Bang illustrated the flow from input to final output through various processing stages. The system began with the Input Stage, which includes raw gameplay footage and machine learning models (Convolutional Neural Network [CNN] and Long Short-Term Memory [LSTM] networks), integrated through the Long-term Recurrent Convolutional Network (LRCN) architecture.

Input. This phase included raw gameplay footage and machine learning models (CNN and LSTM), integrated through the LRCN architecture.

Preprocessing. In this phase, the system prepared the video by segmenting frames and sequences. This phase sets up the foundation for accurate event detection.

Processing. This phase involved the collaborative work of the CNN and LSTM models. The LRCN architecture detected team fights by analyzing both spatial features (using CNN) and temporal patterns (using LSTM). Once a team fight is detected, the YOLOv8 model identified in-game banners that appeared within those timestamps. The system applied an 80% confidence threshold to ensure the accuracy of detected banners and validates that the duration of detected events is at least three seconds.

Scoring. In this phase, the system assigned excitement-based scores to each detected banner and aggregates these scores for every detected timeframe. The total priority score determines the likelihood of including a specific timeframe in the final highlights.

Highlight Detection. This phase compiled key information for each event: timestamps, total priority scores, detected banners, and event duration. This data helped determine which moments should be part of the final highlight reel.

Output. This phase generated the highlight video snippets in .mp4 format. This automated output captured the most exciting and impactful gameplay moments requiring additional user review. The review is for the feature that keeps or deletes the moments which are sorted by priority or chronological order, enabling user control and freedom while still streamlining the highlight creation process.

Definition of Terms

An Application Programming Interface (API) establishes the guidelines for how distinct software programs communicate.

Banners refer to messages that are displayed as part of MLBB’s announcer system. The announcer in the game announces all events that are happening throughout the game. These are the events that YOLOv8 is detecting in the initial highlights.

Destroyed refers to each of the two teams having ten turrets that must be destroyed to win the game. This term means one of the turrets has been destroyed.

Wiped Out refers to all five of the heroes who are your enemies being dead.

Has Slain refers to one of the heroes who is your teammate or enemy getting killed.

Killing Spree refers to a hero killing three opponent heroes in a row without dying even once (Kill Streak).

Mega Kill refers to a hero or opponent hero getting a Kill Streak four times.

Unstoppable refers to a hero or opponent hero getting a Kill Streak five times.

Monster Kill refers to a hero or opponent hero getting a Kill Streak six times.

Godlike refers to a hero or opponent hero getting a Kill Streak seven times.

Legendary refers to a hero or opponent hero getting a Kill Streak eight times.

Shut Down refers to a hero or opponent hero who got a Kill Steak getting killed.

First Blood refers to the first kill in a match, often signifying a momentum shift.

Double Kill refers to a hero or opponent hero getting two kills at once or in a short time.

Triple Kill refers to a hero or opponent hero getting three kills at once or in a short time.

Maniac refers to a hero or opponent hero getting four kills at once or in a short time.

Savage refers to a hero killing all opponent heroes at once or in a short time.

Classes refer to events that happen throughout the game. They are also classified as farming, team fight, game ending, lord, and turtle. This is the list of categories that the LRCN architecture is detecting in a raw gameplay video.

Farming involves collecting in-game resources by defeating minions, which is essential for character growth.

Team fight is a coordinated clash between both teams, often determining control of key objectives.

Game ending refers to another concept often seen in the sphere of online interactive entertainment, originating from the idea of a video game ending when one player or, in MLBB, one team successfully defeats another.

Lord refers to the Elemental Lord, who is a major objective in the game. The objective first appears within eight minutes, spawning randomly on either of the two pits located in the river area of the map. The Lord's appearance and timer are displayed on the minimap and are critical strategic elements in gameplay.

Turtle refers to another important objective in the game. Also known as Dragon Turtle or Cryoturtle depending on the style of the map, this jungle creep spawns within the first two minutes of the game and subsequently respawns every two minutes. It stops respawning after eight minutes when its successor, the Lord, first spawns.

Classification refers to sorting videos into predefined categories based on visual content or detected actions, similar to image classification but over temporal sequences.

Classification models are supervised learning systems trained to assign input data into specific categories.

Final highlights refer to moments when banners detected by YOLOv8 in the initial highlights. The banners are assigned priority scores and are sorted by them or chronological order, whichever is by the user's decision.

Graphics Processing Unit (GPU) provides parallel processing power, significantly speeding up machine learning computations.

Epoch represents one full cycle of training over the entire dataset.

Initial highlights refer to team fights detected by the LRCN architecture in the raw gameplay video.

Priority Score refers to a numerical value assigned to in-game banners based on their significance, used to determine the relevance of a scene for highlight generation.

Raw gameplay footage refers to complete, unedited recording of one full MLBB match captured through the game's spectator mode. This footage includes all in-game visuals such as heroes' movements, team fights, in-game banners, skill effects, and map events, as well as audio elements like background music, sound effects, and live commentary.

Tensor Processing Unit (TPU) a piece of specialized hardware from Google, efficiently manages intensive machine learning workloads.

Video Highlight Generation refers to the process of automatically extracting and summarizing important moments from long-form video content, typically focused on enhancing viewer engagement.

Chapter 3

METHODOLOGY

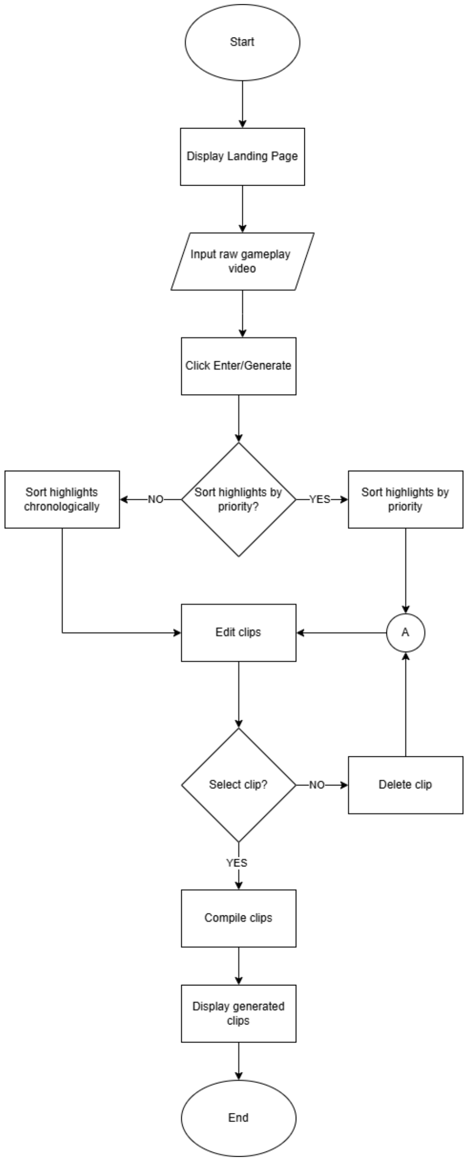
In this division, we covered the layout of the system, how it came to be, and the operational methods to be carried out by the researchers, as well as the proposed flow of the steps to be completed to accumulate information essential for the study.

Project Design

The central goal of this work focuses on formulating highlight generator assisting in organizing and quickly retrieving esports video content, allowing creators, who are one of the stakeholders, to focus less on the labor-intensive editing process (Sridevi & Kharde, 2020). Manual editing can lead to delays, reducing the immediacy of content delivery, which is essential in the dynamic world of Esports (Pyun et al., 2023). This process can be automated using a highlight generator powered by CNN and LSTM networks, thereby streamlining content creation and enhancing viewer engagement.

Figure 16

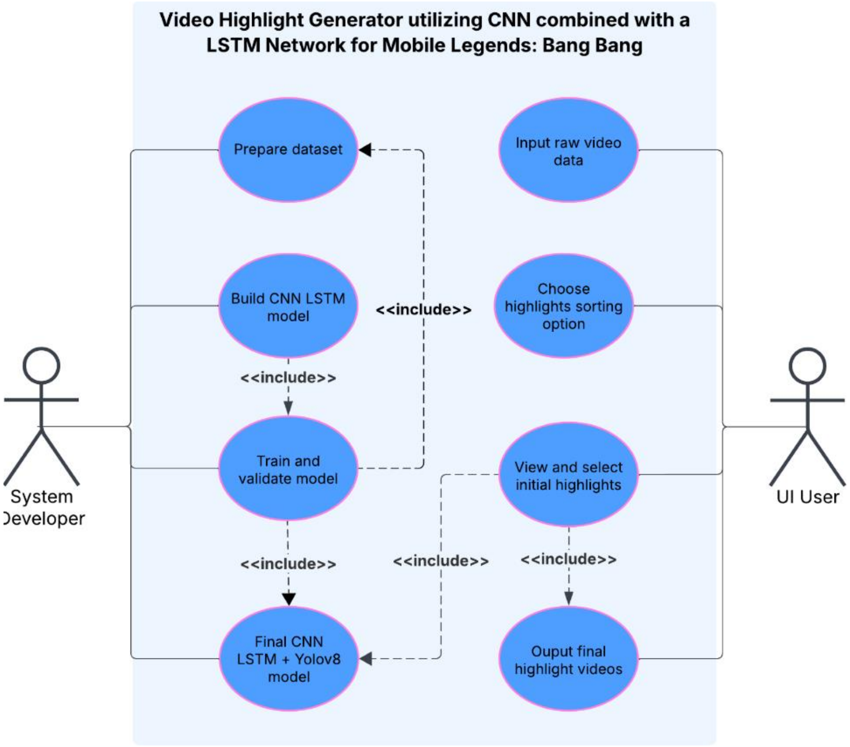
Flowchart of the Proposed System



The flowchart illustrates the system's internal processes. The first process was displaying the landing page in which the user inputted the raw gameplay video. The second process was clicking enter or generate in which the user clicked the button to enter or generate highlights from the input. The next process was sorting highlights chronologically or by priority whichever was by the user's decision. The process of editing clips was where the user reviewed the highlight clips and decided whether to keep them in the final output or delete them. The highlight clips the user decided to keep were then compiled into the final output and displayed to the user.

Figure 17

Use Case Diagram of the System


Video Highlight Generator utilizing CNN combined with a LSTM Network for Mobile Legends: Bang Bang
Prepare dataset
Input raw video data
Build CNN LSTM model
<<include>>
Choose highlights sorting option
<<include>>
System Developer
Train and validate model
View and select initial highlights
UI User
<<include>>
<<include>>
<<include>>
Final CNN LSTM + Yolov8 model
Ouput final highlight videos

The diagram displays the system workflow and how the processes interact between the two user roles — represented by the blue circular nodes and the arrows. These two actors are:

a. **System Developer** – handles system technical tasks like dataset setup, model creation, training, and validation.
b. **UI User** – uses the interface to input data and retrieve system output.

Below are the available methods for System Developers:

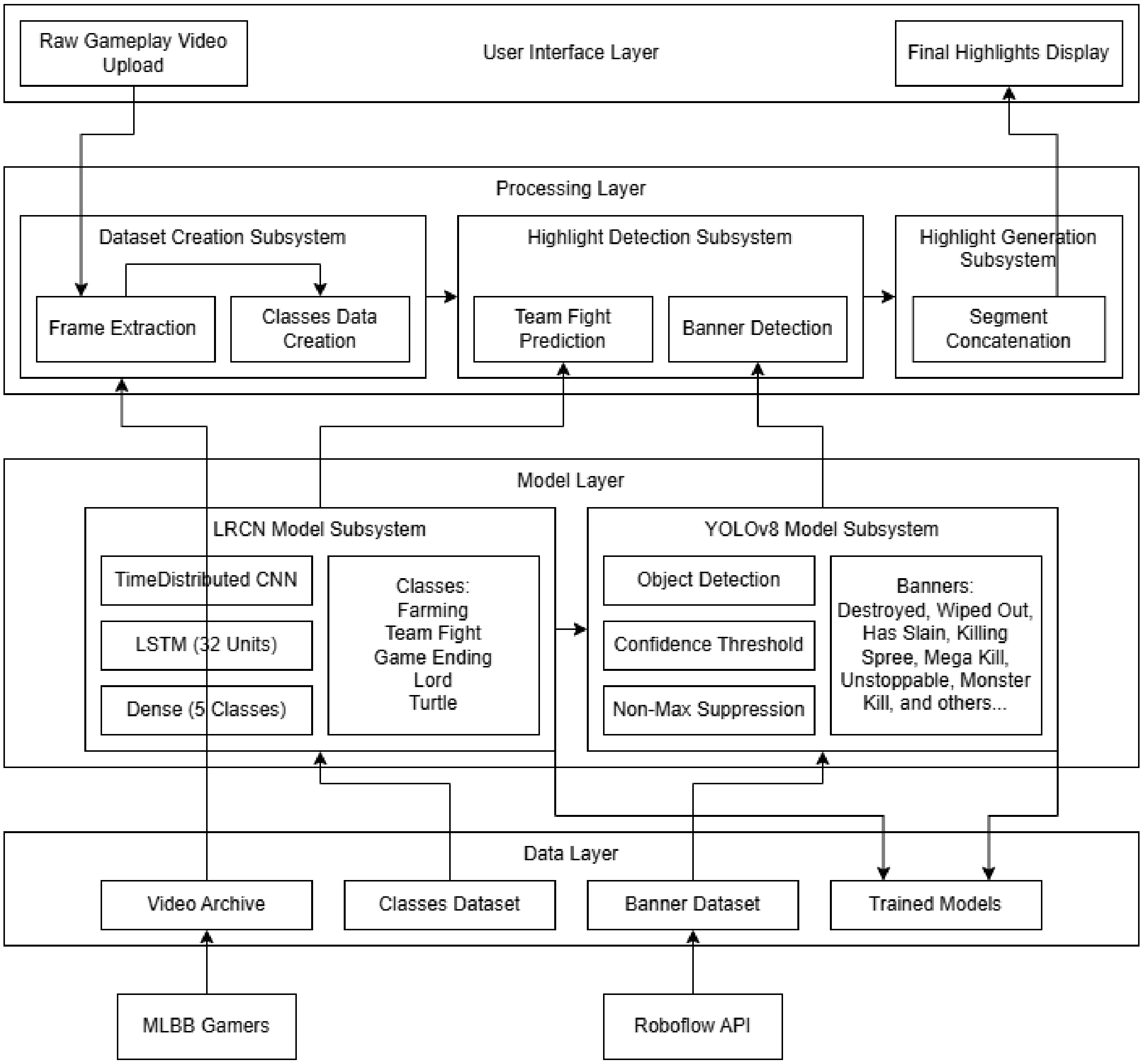
1. Prepare Dataset – involves collecting and pre-processing data for training and validation.
2. Build CNN LSTM model – creation or coding of the model through Python.
3. Train and Validate model – testing and optimizing the model. The dataset and the model structure are vital for this process.
4. Final CNN LSTM + YOLOv8 model – built after undergoing training and validation stages.

The UI user can access the following features of the interface:

1. Input raw video – process whereby the system imports the raw MLBB video.
2. Choose highlights sorting option – the user can choose between ‘sort highlights chronologically’ and ‘sort highlights by priority’.
3. View and select final highlights – let users edit highlight sequences generated by the CNN LSTM + YOLOv8 model from raw video input.
4. Output final highlights video – concatenation of the edited final highlight sequences.

Figure 18

System Architecture



The figure above is the system architecture diagram for the Video Highlight Generator. The diagram showed how the Video Highlight Generator is structured and was organized into four main layers, showing the key components and their subsystems and relationships that formed a comprehensive view of the system.

The User Interface Layer forms the first layer in the architecture. In it are the components Raw Gameplay Video Upload and Final Highlights Display. As their names suggest, the components provide processes for uploading a raw gameplay video and displaying its final highlights. This layer represents the user interaction points.

The Processing Layer follows, comprising subsystems for dataset creation, highlight detection, and generation

In Dataset Creation, the component Frame Extraction extracts the frames from the raw gameplay video uploaded in the first layer. In the same subsystem is the Classes Data Creation. This component does not create classes data from the raw gameplay video but from the videos prepared by the researchers. These videos are clips of other gameplay videos classified into farming, team fight, game ending, lord, and turtle.

The next subsystem is Highlight Detection and in it are the components Team Fight Prediction and Banner Detection. Team Fight Prediction predicts team fights from the raw gameplay video and Banner Detection detects banners from those team fights. This subsystem detects the initial highlights.

The final subsystem is Highlight Generation in which the initial highlights (as clips or segments) undergo user review and filter which (segments) are kept, and which are deleted. After filtration, the segments, which the user does decide to keep, are concatenated into what is known as the final highlights. The final highlights are then displayed in the User Interface Layer.

Next is the Model Layer which includes the LRCN and YOLOv8 models as subsystems.

The LRCN Model consists of the TimeDistributed, LSTM, and Dense Output layer. The TimeDistributed layer applies CNN functions while keeping the time sequence of raw gameplay data intact. Using the temporal structure, the LSTM layer with 32 units captured the evolution of the learned features frame by frame or events over time. The Dense layer uses Softmax activation to classify extracted features into five categories. The model used the classes data created in Classes Data Creation to predict team fights in Team Fight Prediction, both in the Processing Layer.

The YOLOv8 model serves as the object detection tool used to identify banners in predicted team fight segments. The confidence threshold ensured the accuracy of the detected banners and validates that the team fights last at least three seconds. The non-max suppression also ensures accuracy by making sure the banners are detected only once (GeeksforGeeks, 2025a).

Lastly, the Data Layer consists of Video Archive, Classes Dataset, Banner Dataset, and Trained Models components. The Video Archive stored the raw gameplay videos uploaded by the external system MLBB Gamers. The Classes Dataset is where the created classes data is stored. Meanwhile, the banner dataset was sourced via the external Roboflow API. That is where the researchers prepared the banner data to be used by the YOLOv8 Model which is then used in Banner Detection in the Processing Layer. The trained LRCN and YOLOv8 models are stored in Trained Models.

Project Development

Program Coding

The classification and object detection models were created using the TensorFlow and Ultralytics libraries. A CNN-LSTM hybrid was used to classify gameplay events, while YOLOv8 handled key in-game banner detection. Both models were trained in Google Colab and later integrated into a Python-based frontend application.

A. Gameplay Classification Model

The classification model combined CNN and LSTM to capture spatial and temporal features from gameplay frames. It was built through these steps:

1. Legends: Bang Bang gameplay videos were sourced through a formal request made to the Gaming Enthusiasts Association Ring (TUP GEAR), an accredited Esports club and student organization at the Technological University of the Philippines (TUP).
2. The collected gameplay videos were manually trimmed and segmented into five categories: team fights, farming, killing of the Turtle, killing of the Lord, and game ending.
3. Each video segment was then converted into a sequence of frames to serve as input for the model.
4. Establish data consistency and better model outcomes, frames were taken out at fixed points, resized uniformly, and then normalized.
5. These frame sequences were then grouped and labeled according to their corresponding classes.
6. The model used an LRCN architecture that combines CNN for spatial feature extraction and LSTM for temporal dependency modeling across frame sequences.
7. A softmax output layer was utilized to enable multi-class classification.

8. The dataset was split into training and validation sets to facilitate supervised learning and performance testing.
9. The model was optimized with Adam and categorical cross-entropy, suitable for multi-class classification tasks.
10. Performance metrics like accuracy and loss were tracked during training to assess convergence and generalization.
11. Key hyperparameters such as learning rate and batch size were tuned using validation results to boost accuracy and prevent overfitting.
12. The final trained model was saved in .h5 format.
13. It was then prepared for deployment within a Python-based frontend application.

B. Banner Detection Model

YOLOv8n, a lightweight version of the YOLOv8 detection model, was used to detect in-game banners efficiently in near-real-time scenarios. The development process of the banner detection model followed these steps:

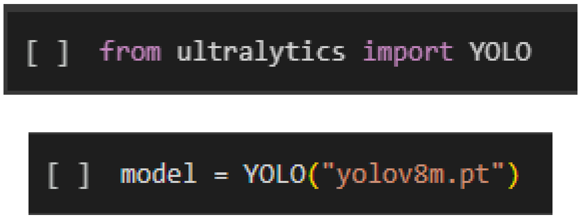
1. Frames were extracted from video segments previously classified as team fights by the gameplay classification model. These segments are the most likely to contain relevant in-game banners.
2. The extracted frames were imported into the Roboflow platform for manual annotation. Using Roboflow’s annotation tools, bounding boxes were drawn around the following banner categories:
    a. “Destroyed”
    b. “Wiped Out”

    c. “Has Slain”
    d. “Killing Spree”
    e. “Mega Kill”
    f. “Unstoppable”
    g. “Monster Kill”
    h. “Godlike”
    i. “Legendary”
    j. “Shut Down”
    k. “First Blood”
    l. “Double Kill”
    m. “Triple Kill”
    n. “Maniac.”
    o. “Savage”

3. Roboflow handled dataset splitting and exported the data to Google Colab via API or ZIP format.
4. The YOLOv8m model (yolov8m.pt) was initialized using the Ultralytics YOLOv8 library

Figure 19

YOLOv8 Initialization



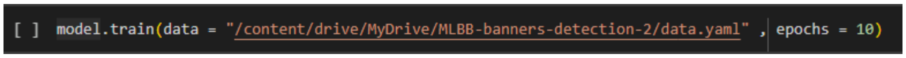
```
[ ]  from ultralytics import YOLO
```

```
[ ]  model = YOLO("yolov8m.pt")
```

Training was performed using the following command:

Figure 20

YOLOv8 Training Function



```
[ ]  model.train(data = "/content/drive/MyDrive/MLBB-banners-detection-2/data.yaml" , epochs = 10)
```

Training was set for 10 epochs with a batch size of 16 and 640 image resolution. The data.yaml defined 15 banner categories.

5. Training occurred in Google Colab on CPU, with metrics like mAP and loss monitored. Data augmentations like flipping and scaling were used for better generalization.
6. The final model consisted of 295 layers with approximately 25.8 million parameters and supported 15 classes corresponding to the in-game banners. Pretrained weights were partially transferred, and key layers were fine-tuned based on the custom dataset.

C. Python Frontend Development

A Python-based GUI was built to let users upload gameplay videos and interact with the trained models, initiate classification and detection processes, preview extracted highlights, and export final output clips. The frontend was built using several Python libraries including *customtkinter*, *tkinterdnd2*, *pillow*, *tkVideo*, *opencv-python*, *moviepy*, and others. Below is a step-by-step outline of the frontend development and backend integration process.

1. A new Python project was initialized as a CustomTkinter application, setting up the base window using CTk (CustomTkinter's main window class).

2. The graphical user interface was designed with a modern layout that included the following components:
    a. Drag-and-drop area for uploading gameplay videos.
    b. Buttons for triggering extraction and preview processes.
    c. Embedded video preview window for extracted clips.
    d. Message dialogs for alerts, confirmations, and reminders.
3. The application window was styled using Pillow to set custom window icons, and CustomTkinter styling options (e.g. rounded corners) were applied for a polished appearance.
4. The tkinterdnd2 library was integrated to allow users to drag gameplay video files directly into the application window. This enhanced usability by reducing the need for manual file path selection.
5. The os module was used to extract and manage file paths and filenames. This was essential for dynamically displaying the uploaded video’s name and handling storage directories.
6. tkVideo was employed to allow users to preview extracted highlight clips within the GUI. The module enabled silent playback of .mp4 clips embedded directly into the frontend window.
7. The CTkMessagebox module was used to implement error alerts, success notifications, and user confirmations, improving interaction and reducing the risk of user-side errors.

D. Backend Integration

1. The trained models were loaded into the application environment at runtime:
    a. The CNN-LSTM classification model was loaded in .h5 format using TensorFlow/Keras.
    b. The YOLOv8 object detection model was loaded in .pt format using the Ultralytics YOLOv8 API.
2. Uploaded videos were processed using OpenCV:
    a. Frames were extracted at fixed intervals.
    b. Frames were resized and normalized to match the classification model's input requirements.
3. Frame sequences were fed into the CNN-LSTM model to classify video segments into four categories: team fights, farming, Turtle kills, and Lord kills. Time segments labeled as team fights were selected for further banner detection.
4. The selected video frames were passed through the YOLOv8 model to detect specific in-game banners. Bounding boxes and class labels were returned for each detected banner.
5. Using MoviePy, the timestamps corresponding to detected events were used to extract video snippets, which were temporarily stored for review within the frontend application. Additionally, it was used to concatenate user-approved clips into a single highlight reel for export.

Operation and Testing Procedure

Operation

To use the application, follow these steps:

1. The user launches the system on their personal computer or laptop.
2. The user uploads a raw gameplay video by clicking the upload button or dragging and dropping the file into the system.
3. The user clicks the generate button to start the highlight generation process.
4. The system asks the user to choose whether the output should be arranged chronologically or by priority.
5. The system detects team fights in the video while displaying a loading screen.
6. The system proceeds to detect banners from the detected team fights, showing another loading screen.
7. The system extracts clips from the identified moments, with another loading screen appearing during this process.
8. The user waits for the processing to finish, which takes several minutes and is faster with a GPU.
9. The system shows the extracted clips one by one, and the user decides to keep or delete each clip.
10. The user can use playback controls such as replay, pause, and adjust speed while reviewing clips.
11. Once all clips are reviewed, the system notifies the user that they can concatenate the selected clips.

12. The user selects where to save the output video and names the file, then clicks the save button.
13. A confirmation message appears to indicate that the highlight video has been successfully saved.
14. After using the application, the user can now simply close the application.

Testing Procedure

To identify and rectify errors, as well as to verify the functionality of the system's features, a series of tests were conducted. Table 1 outlined the various test cases executed, detailing the specific steps undertaken and their corresponding expected outputs.

Table 1

Operation and Testing Procedure for Functionality

Test Cases	Steps Undertaken	Expected Output
Insert clip using "Choose File" button	1. User clicked the "Choose File" button. 2. File explorer is opened showing only accepted file types (MP4, AVI, MOV, MKV, WMV). 3. User selected a file.	The file name will be displayed on the file drop zone.
Insert clip using drag and drop feature	1. User opened file explorer. 2. User dragged a video file to the dropzone.	The file path will be displayed on the drop zone.
Inserting a non-video file type using drag and drop feature	1. User opened file explorer. 2. User dragged a file type that is not accepted by the system (PDF, PNG, JPEG, XLSX, etc.)	This error message will appear on the drop zone: "Invalid file type. Please select a video file (MP4, AVI, MOV, MKV, WMV)."
Inserting more than one clip using the	1. User clicked the "Choose File" button.	The file name of the most recent file the

“Choose File” button.	2. File explorer is opened showing only accepted file types (MP4, AVI, MOV, MKV, WMV). 3. User selected a file. 4. User clicked the “Choose File” button again. 5. User selected another file.	user has selected will be displayed on the drop zone.
Inserting more than one clip using the drag and drop feature.	1. User opened file explorer. 2. User dragged a video file to the dropzone. 3. User opened file explorer again. 4. User dragged another video file to the dropzone.	All the file path of the videos dragged into the drag zone will be displayed. But will not be considered by the system. When the “Generate” button is clicked, an error message “Please select a file” will be displayed.
“Generate” button functionality	1. User clicked the “Generate” button after inserting a valid video clip. 2. A pop-up appeared prompting the user to select whether they want the clips to be sorted chronologically or sorted from most to least exciting.	After selecting a sorting type, the input processing progress bar will appear. First, it will display “Team Fight Detection”. Then, it will display “Banner Detection”. Lastly, it will display “Extracting Clips” while also showing the number of clips being extracted below the progress bar.
User clicks the “Generate” button without selecting a clip	1. User clicked the “Generate” button without selecting a clip	An error message will pop up notifying the user: “Please select a video file.”
User inserts a video that is not MLBB gameplay.	1. Users inserted a clip from their gameplay from different games like Valorant or League of Legends. 2. User clicked the “Generate” button.	A pop-up will appear prompting the user to select whether they want the clips to be sorted chronologically or sorted from most to least exciting. After selecting a sorting

		type, the input processing progress bar will appear. If there are no detected team fights, an error message will pop up notifying the user: “No generated highlight clips.”
“Video Editor” window	1. After video processing is done, a “Video Editor” window popped-up.	All progress bar pop-ups and the homepage window will be closed.
Browsing generated videos using arrow buttons	1. User clicked the right arrow key. 2. User clicked the left arrow key.	Right arrow key allows the user to go back to a previous video clip, while the left arrow key allows the user to see the next video clip.
Keeping video clips	1. User clicked the “Keep Clip” button.	The selected button will have a green foreground color, and the number of clips selected for concatenation on the instruction label will increase.
Deleting video clips	1. User clicked the “Delete Clip” button.	The selected button will have a red foreground color, and the number of clips selected for concatenation on the instruction label will not change.
Clicking the “Keep Clip” button after the “Delete Clip” button is selected.	1. The user clicked “Keep Clip”. 2. The user changed their mind and went back to the clip to select “Delete Clip”	The “Keep Clip” will return to its default setting and the “Delete Clip” will have a red foreground color. The number of clips selected for concatenation will decrease.
Clicking the “Delete Clip” button after	1. The user clicked “Delete Clip”.	The “Delete Clip” will return to its default setting and the “Keep

the "Keep Clip" button is selected.	2. The user changed their mind and went back to the clip to select "Keep Clip"	Clip" will have a green foreground color. The number of clips selected for concatenation on the instruction label will also increase.
User has viewed all clips and has selected keep/delete for all clips	1. User browsed through all the clips. 2. User selected to *keep* some clips to be concatenated. 2. User selected *delete* to the video clips they chose not to keep.	A pop-up message will appear notifying the user: "You've reached the last video. You can now concatenate all selected clips."
Arrow buttons functionality	1. Left and right arrow buttons are disabled by default. 2. User selected keep/delete for a video clip. 3. Left and right arrow buttons are enabled.	The arrow buttons are only enabled after the user makes a choice. Once enabled, the user can view the previous or next video clip by clicking the arrow keys.
Restart button functionality	1. User clicked the "Restart" button in the "Video Editor" window.	The current video being displayed will go back to the start of the clip.
Stop button functionality	1. User clicked the "Stop" button in the "Video Editor" window.	The current video being displayed will be paused.
Speed button functionality	1. User clicked the "1.0x" button in the "Video Editor" window.	The current video will be sped up based on what number the user selects.

Evaluation Procedure

The evaluation tool utilized to measure the system's acceptability was adapted from ISO 25010, titled "*Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – System and Software Quality Models.*"

To assess the acceptability of the developed system, the following steps were carried out:

1. A total of 50 purposively selected respondents were invited, including MLBB players, an MLBB Esports manager, an MLBB video highlights editor, Esports officer, and MLBB fan.
2. A Google Form containing a video demonstration of how the MLBB Video Highlight Generator functions, along with a brief explanation of the algorithm it uses, was shared with the evaluators.
3. Evaluators gave their appraisal of the system by means of analysis document rooted in a 4-point Likert scale, as presented in Table 2.
4. Completed evaluation forms were collected, and the responses were compiled and tabulated to calculate the weighted mean ratings.
5. The weighted mean findings were subsequently understood by applying the descriptive rating scale presented in Table 2.

Table 2

Range of weighted mean and its Qualitative Interpretation

Scale	**Qualitative Interpretation**	**Range**
4	Highly Acceptable	3.26 – 4.00
3	Very Acceptable	2.51 – 3.25
2	Acceptable	1.76 – 2.50
1	Not Acceptable	1.00 – 1.70

Chapter 4

RESULTS AND DISCUSSION

This chapter contains the project description, project structure, project capabilities and constraints, and results of evaluation for the project

Project Description

The proposed system, titled "Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang", is a desktop-based software designed to assist users in generating gameplay highlights from recorded matches of Mobile Legends: Bang Bang (MLBB). The system identified high-intensity moments such as team fights, farmings, and major objectives (e.g., Turtle and Lord), streamlining the creation of highlight reels for content creators and Esports analysts.

The system integrated Long Short-Term Memory (LSTM) networks to detect the temporal progression of in-game events, particularly sequences that indicated team fights or engagements. To support event recognition through visual context, the system employed Convolutional Neural Networks (CNN) enhanced with YOLOv8 (You Only Look Once version 8) for object detection. YOLOv8 processed visual cues such as banners or indicators that signified key moments in the game.

The system offered suggested clips based on detected events, which users can manually review and export. This provided flexibility for selective editing and ensured that important but nuanced plays are not overlooked by a rigid algorithm.

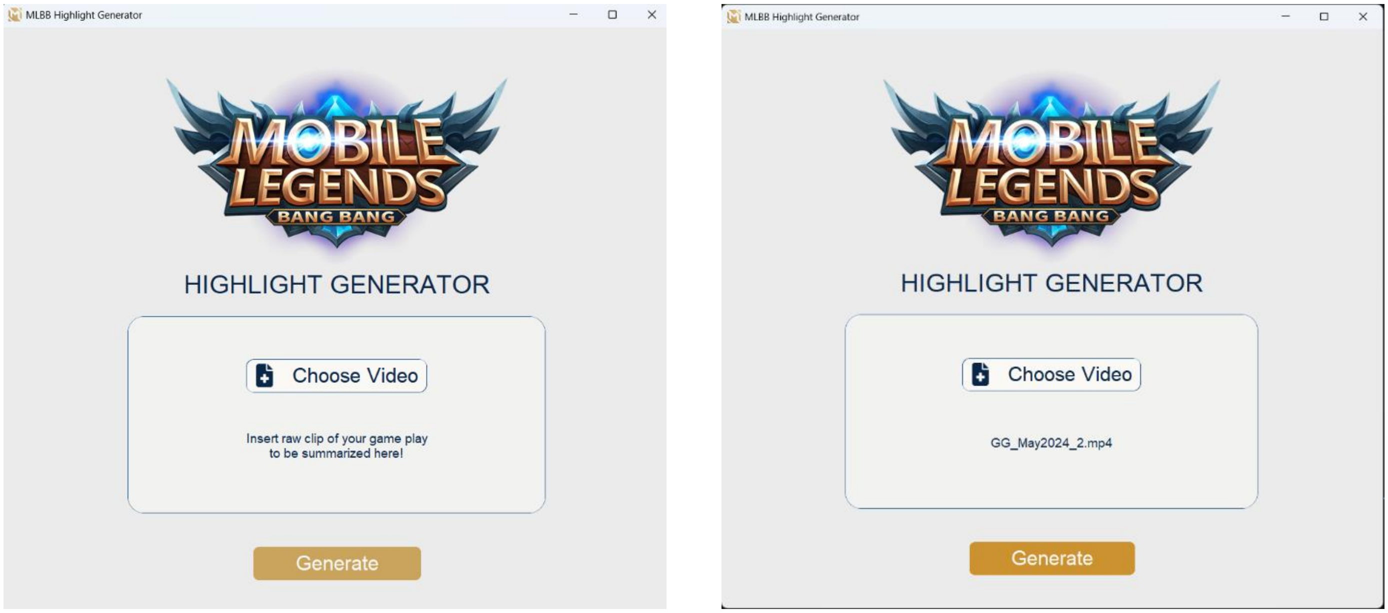
The software featured a user-friendly graphical interface built with Tkinter and CustomTkinter, and included drag-and-drop functionality through tkinterdnd2, enabling easy video input and interaction. All model inference and video processing are executed locally on the user's machine, allowing full offline usability.

Project Structure

The Video Highlight Generator offered a linear, step-by-step interface that guide users through the process of highlight creation. Upon launching the application, users are greeted with a main dashboard (see Figure 21, left panel), where they can upload their Mobile Legends gameplay footage by either clicking the "Choose Video" button or using the drag-and-drop feature. This intuitive interface is developed using Tkinter, CustomTkinter, and tkinterdnd2.

Figure 21

Main Dashboard Before and After Video Upload



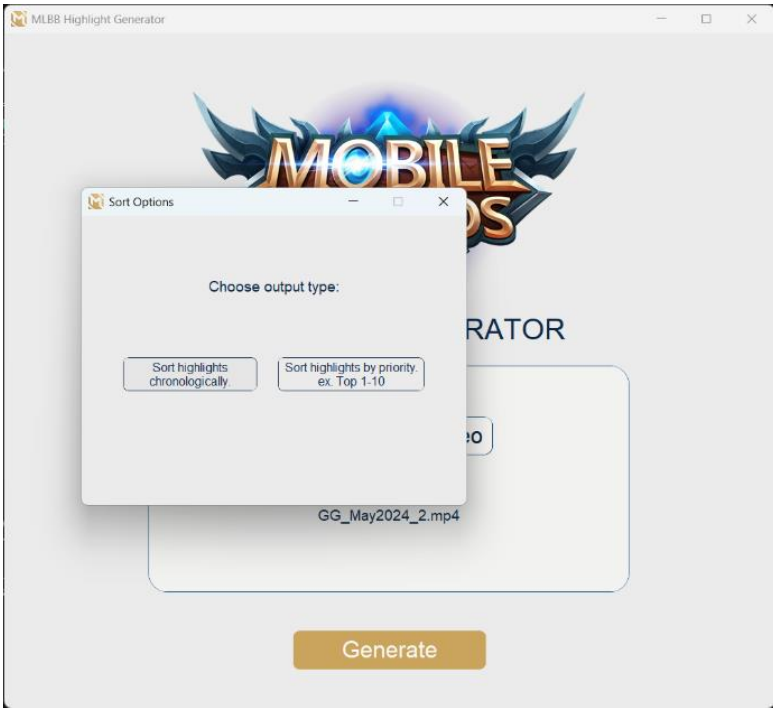
Insert raw clip of your game play
to be summarized here!

GG_May2024_2.mp4

After uploading the video in the dashboard (see Figure 21, right panel), users initiated the highlight generation process by clicking the "Generate" button.

Figure 22

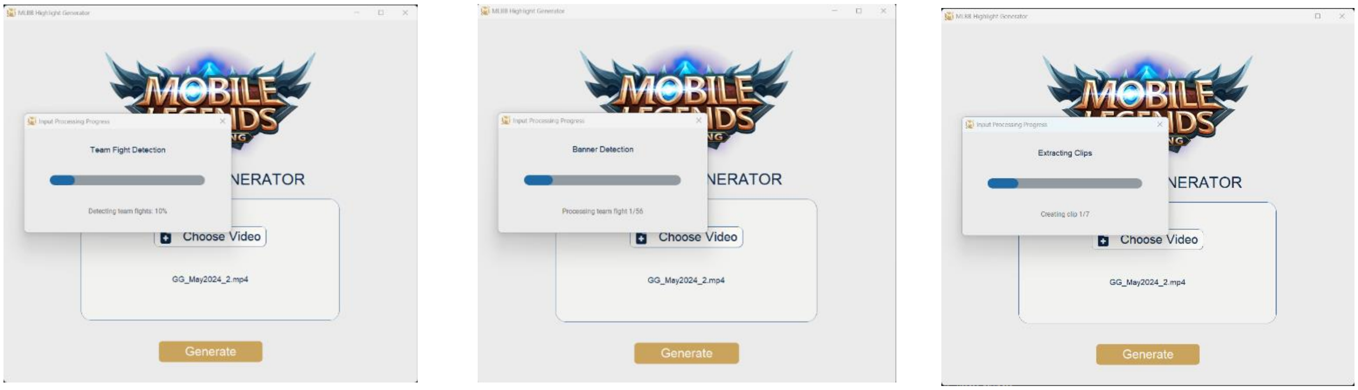
Highlight Arrangement Options



As shown in Figure 22, after clicking the Generate button, the system prompted the user to choose how the highlights should be arranged—either in chronological order or based on event priority.

Figure 23

Loading Screens for each Stage of Highlight Generation Process



Detecting team fights: 10%

GG_May2024_2.mp4

Processing team fight 1/56

GG_May2024_2.mp4

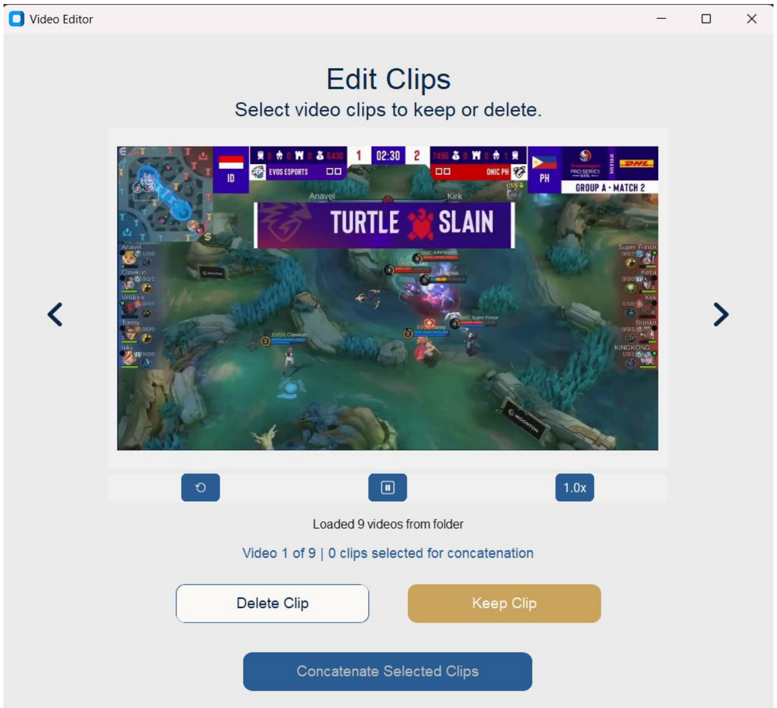
Creating clip 1/7

GG_May2024_2.mp4

The application proceeded through three automated stages, each represented by a dedicated loading screen. The first stage, Team Fight Detection, involves analyzing temporal patterns in the video using the LRCN model (see Figure 23, left panel). The second stage, Banner Detection, utilizes the YOLOv8 model to identify visual banners that signify major objectives (see Figure 23, middle panel). The third and final stage, Clip Extraction, automatically segments the identified highlight moments into separate video clips (see Figure 23, right panel).

Figure 24

Review Screen Showing Sequentially Displayed Highlight Clips with Playback and Management Controls



Edit Clips

Select video clips to keep or delete.

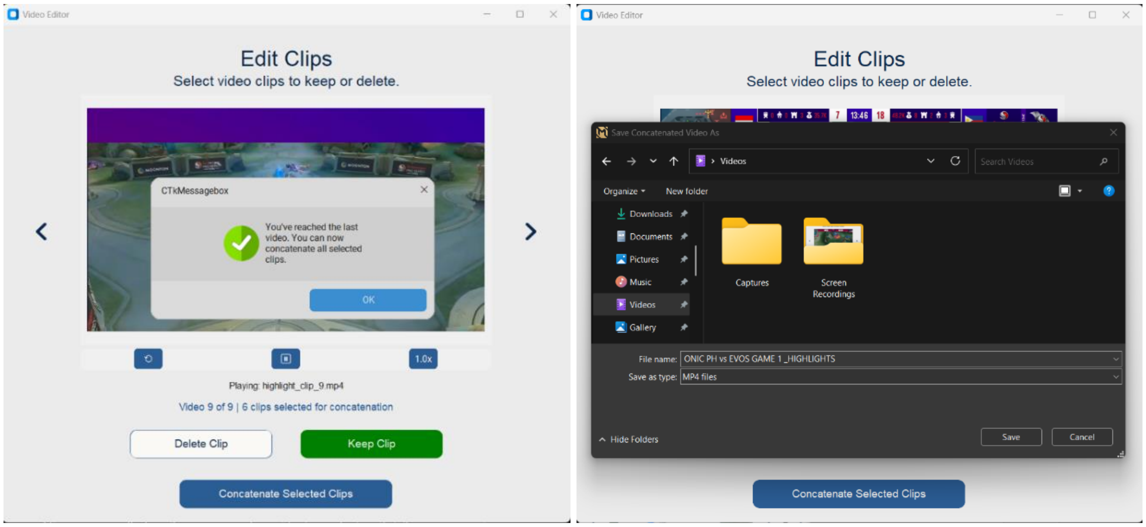
Loaded 9 videos from folder

Video 1 of 9 | 0 clips selected for concatenation

Once processing is completed, the user is directed to a review screen (see Figure 24). In this interface, each generated clip is displayed sequentially, allowing the user to preview, retain, or delete individual segments. Playback controls—such as replay, pause, and speed adjustment—are provided to facilitate convenient and efficient review.

Figure 25

Concatenation Prompt and File-Saving Interface for Exporting Selected Highlights



Edit Clips

Select video clips to keep or delete.

CTkMessagebox

You've reached the last video. You can now concatenate all selected clips.

Playing: highlight_clip_9.mp4

Video 9 of 9 | 6 clips selected for concatenation

Save Concatenated Video As

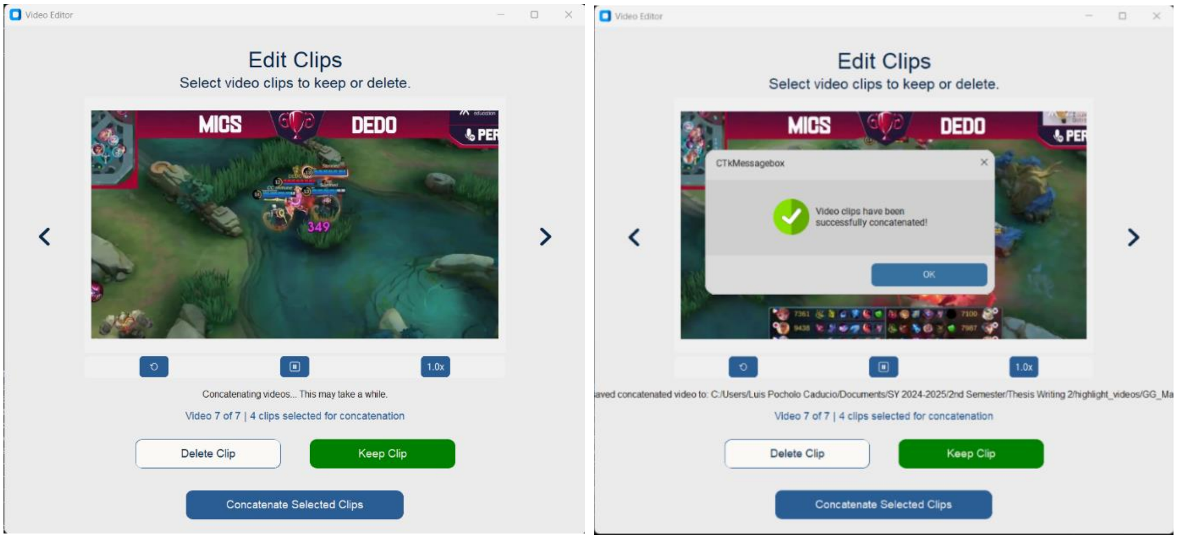
File name: ONIC PH vs EVOS GAME 1 _HIGHLIGHTS

Save as type: MP4 files

After reviewing all clips, the user received a notification indicating that the selected highlights are ready for concatenation (see Figure 25, left panel). The user then selects a save location and specifies a filename for the final output (see Figure 25, right panel).

Figure 26

Final Video Export Process and Confirmation of Successful Save



Edit Clips

Select video clips to keep or delete.

Concatenating videos... This may take a while.

Video 7 of 7 | 4 clips selected for concatenation

CTkMessagebox

Video clips have been successfully concatenated!

aved concatenated video to: C:/Users/Luis Pocholo Caducio/Documents/SY 2024-2025/2nd Semester/Thesis Writing 2/highlight_videos/GG_Ma

Video 7 of 7 | 4 clips selected for concatenation

Once confirmed, the system proceeds to export the concatenated video (see Figure 26), followed by a confirmation message notifying the user of a successful save. All steps are executed locally, ensuring smooth and offline processing. After this, the user may close the application.

Project Capabilities and Limitations

The following are the capabilities of the developed Video Highlight Generator:

1. The Video Highlight Generator could run on a device with a Windows Operating System version 8 and later.
2. The Video Highlight Generator was tested on three (3) systems. Specifically:
    a. Intel® Core™ i5-1155G7 Processor with Intel® Iris® Xe Graphics
    b. Intel® Core™ i7-1255U Processor with Intel® Iris® Xe Graphics
    c. AMD Ryzen™ 5 5600G Processor with Radeon™ Graphics
3. The Video Highlight Generator could run on the Intel® Core™ i5-1155G7 Processor and Intel® Core™ i7-1255U Processor for a minimum time of approximately 32 minutes. On the AMD Ryzen™ 5 5600G Processor, the runtime is 20 minutes less.
4. The system provides a manual review interface for the user to verify and edit highlight clips before exporting, giving more control over final output.
5. The model filters non-highlight scenes such as farming and idle moments, focusing on relevant gameplay events like team fights and major objectives.

These are the Video Highlight Generator's drawbacks as of development:

1. The Video Highlight Generator was only tested on three (3) systems. Testing its performance on different systems and processors is planned for future development.
2. The Video Highlight Generator would not perform as accelerated on a system that has no graphics card.
3. The system may require a significant amount of RAM (8GB minimum recommended) and disk storage when processing long video files, especially during model inference and video segmentation stages.
4. The Video Highlight Generator has not been performed with frames whose size is even lower for an accelerated runtime.
5. The system is not optimized for real-time highlight generation. Video processing is performed after the full match has been recorded.
6. The current version of the Video Highlight Generator is specifically trained and tuned for *Mobile Legends: Bang Bang* gameplay and may not generalize to other games.

Test Results

A. System Testing

The test results on the functionality of the developed system are presented in the tables that follow.

Table 3

Operation and Testing Procedure Results for Functionality

Test Case	Steps Undertaken	Observed Results

Insert clip using "Choose File" button	1. User clicked the "Choose File" button. 2. File explorer is opened showing only accepted file types (MP4, AVI, MOV, MKV, WMV). 3. User selected a file.	The file name will be displayed on the file drop zone.
Insert clip using drag and drop feature	1. User opened file explorer. 2. User dragged a video file to the dropzone.	The file path will be displayed on the drop zone.
Inserting a non-video file type using drag and drop feature	1. User opened file explorer. 2. User dragged a file type that is not accepted by the system (PDF, PNG, JPEG, XLSX, etc.)	This error message will appear on the drop zone: "Invalid file type. Please select a video file (MP4, AVI, MOV, MKV, WMV)."
Inserting more than one clip using the "Choose File" button.	1. User clicked the "Choose File" button. 2. File explorer is opened showing only accepted file types (MP4, AVI, MOV, MKV, WMV). 3. User selected a file. 4. User clicked the "Choose File" button again. 5. User selected another file.	The file name of the most recent file the user has selected will be displayed on the drop zone.
Inserting more than one clip using the drag and drop feature.	1. User opened file explorer. 2. User dragged a video file to the dropzone. 3. User opened file explorer again. 4. User dragged another video file to the dropzone.	All the file path of the videos dragged into the drag zone will be displayed. But will not be considered by the system. When the "Generate" button is clicked, an error message "Please select a file" will be displayed.
"Generate" button functionality	1. User clicked the "Generate" button after inserting a valid video clip. 2. A pop-up appeared prompting the user to	After selecting a sorting type, the input processing progress bar will appear. First, it will display "Team Fight Detection". Then, it will display "Banner

	select whether they want the clips to be sorted chronologically or sorted from most to least exciting.	Detection”. Lastly, it will display “Extracting Clips” while also showing the number of clips being extracted below the progress bar.
User clicks the “Generate” button without selecting a clip	1. User clicked the “Generate” button without selecting a clip	An error message will pop up notifying the user: “Please select a video file.”
User inserts a video that is not MLBB gameplay.	1. Users inserted a clip from their gameplay from different games like Valorant or League of Legends. 2. User clicked the “Generate” button.	A pop-up will appear prompting the user to select whether they want the clips to be sorted chronologically or sorted from most to least exciting. After selecting a sorting type, the input processing progress bar will appear. If there are no detected team fights, an error message will pop up notifying the user: “No generated highlight clips.”
“Video Editor” window	1. After video processing is done, a “Video Editor” window popped-up.	All progress bar pop-ups and the homepage window will be closed.
Browsing generated videos using arrow buttons	1. User clicked the right arrow key. 2. User clicked the left arrow key.	Right arrow key allows the user to go back to a previous video clip, while the left arrow key allows the user to see the next video clip.
Keeping video clips	1. User clicked the “Keep Clip” button.	The selected button will have a green foreground color, and the number of clips selected for concatenation on the instruction label will increase.
Deleting video clips	1. User clicked the “Delete Clip” button.	The selected button will have a red foreground color, and the number of clips selected for concatenation on the instruction label will not change.

Clicking the "Keep Clip" button after the "Delete Clip" button is selected.	1. The user clicked "Keep Clip". 2. The user changed their mind and went back to the clip to select "Delete Clip"	The "Keep Clip" will return to its default setting and the "Delete Clip" will have a red foreground color. The number of clips selected for concatenation will decrease.
Clicking the "Delete Clip" button after the "Keep Clip" button is selected.	1. The user clicked "Delete Clip". 2. The user changed their mind and went back to the clip to select "Keep Clip"	The "Delete Clip" will return to its default setting and the "Keep Clip" will have a green foreground color. The number of clips selected for concatenation on the instruction label will also increase.
User has viewed all clips and has selected keep/delete for all clips	1. User browsed through all the clips. 2. User selected to *keep* some clips to be concatenated. 3. User selected *delete* to the video clips they chose not to keep.	A pop-up message will appear notifying the user: "You've reached the last video. You can now concatenate all selected clips."
Arrow buttons functionality	1. Left and right arrow buttons are disabled by default. 2. User selected keep/delete for a video clip. 3. Left and right arrow buttons are enabled.	The arrow buttons are only enabled after the user makes a choice. Once enabled, the user can view the previous or next video clip by clicking the arrow keys.
Restart button functionality	1. User clicked the "Restart" button in the "Video Editor" window.	The current video being displayed will go back to the start of the clip.
Stop button functionality	1. User clicked the "Stop" button in the "Video Editor" window.	The current video being displayed will be paused.
Speed button functionality	1. User clicked the "1.0x" button in the "Video Editor" window.	The current video will be sped up based on what number the user selects.

Insert clip using “Choose File” button	1. User clicked the “Choose File” button. 2. File explorer is opened showing only accepted file types (MP4, AVI, MOV, MKV, WMV). 3. User selected a file.	The file name will be displayed on the file drop zone.
Insert clip using drag and drop feature	1. User opened file explorer. 2. User dragged a video file to the dropzone.	The file path will be displayed on the drop zone.

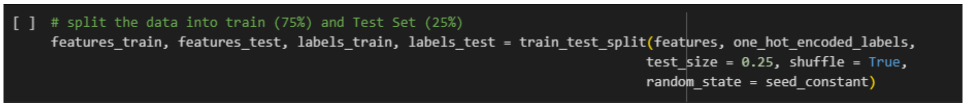
B. Model Testing and Evaluation

The models for gameplay event classification and in-game banner detection were developed using TensorFlow and Ultralytics. For classifying gameplay events, through a Long-term Recurrent Convolutional Network (LRCN) architecture, a combination of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) was employed, while YOLOv8 was utilized for detecting essential in-game banners. The results of the evaluation of the models are presented:

a. Long-term Convolutional Network (LRCN) Model

Figure 27

Dataset Partitioning

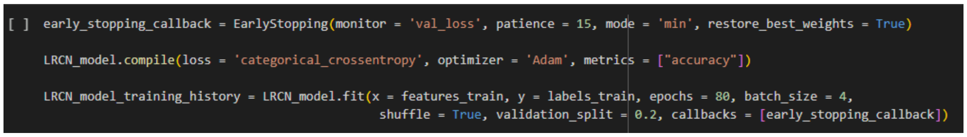


```python
# split the data into train (75%) and Test Set (25%)
features_train, features_test, labels_train, labels_test = train_test_split(features, one_hot_encoded_labels,
                                                                            test_size = 0.25, shuffle = True,
                                                                            random_state = seed_constant)
```

To assess the performance of the LRCN model, the dataset was partitioned into training (75%) and testing (25%) sets using stratified sampling to ensure class balance.

Figure 28

LRCN Model Training



```
[ ]  early_stopping_callback = EarlyStopping(monitor = 'val_loss', patience = 15, mode = 'min', restore_best_weights = True)

     LRCN_model.compile(loss = 'categorical_crossentropy', optimizer = 'Adam', metrics = ["accuracy"])

     LRCN_model_training_history = LRCN_model.fit(x = features_train, y = labels_train, epochs = 80, batch_size = 4,
                                                  shuffle = True, validation_split = 0.2, callbacks = [early_stopping_callback])
```

During training, a validation split of 20% was used from the training data to monitor generalization capability and guide hyperparameter tuning. The model was trained for up to 80 epochs with early stopping enabled to prevent overfitting, halting training if the validation loss did not improve for 15 consecutive epochs. It was compiled using the Adam optimizer and categorical cross-entropy loss, with accuracy as the primary evaluation metric.

To visualize the learning behavior of the model, two plots were generated:

Figure 29

Total Loss VS Total Validation Loss

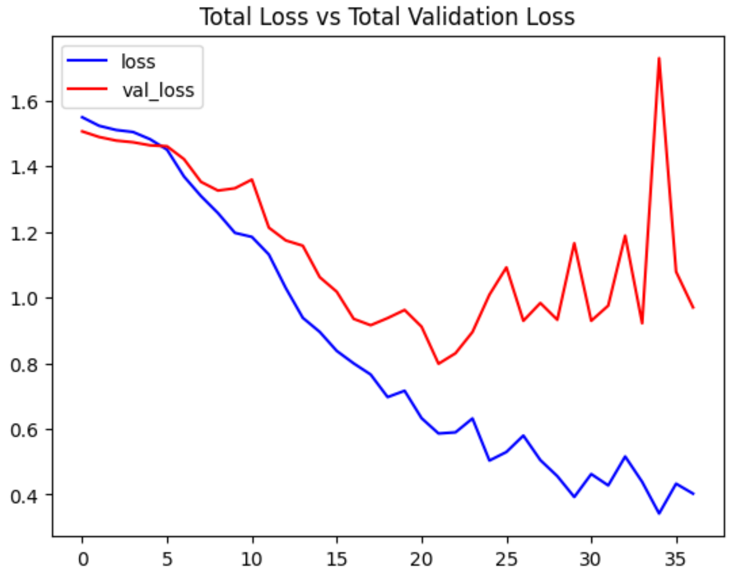


Figure 29 illustrated the training and validation loss trends over time. The training loss (blue) showed a consistent downward trajectory, indicating that the model was effectively learning the features from the training data. The validation loss (red), while initially decreasing, began to fluctuate after epoch 20, suggesting the onset of overfitting. Early stopping was applied to prevent further degradation in validation performance.

Figure 30

Total Accuracy VS Total Validation Accuracy

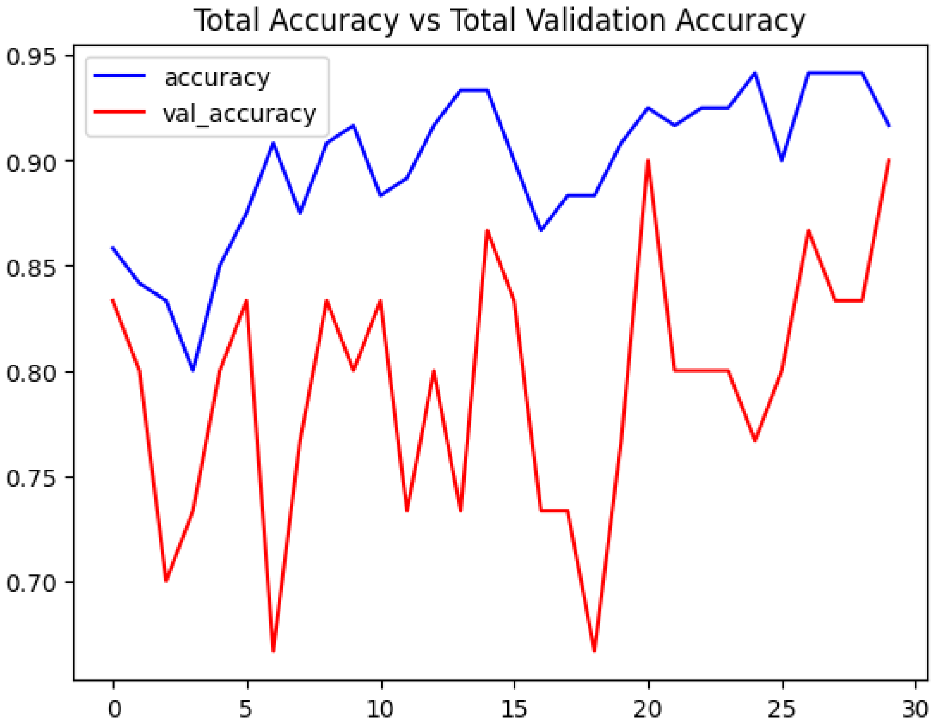
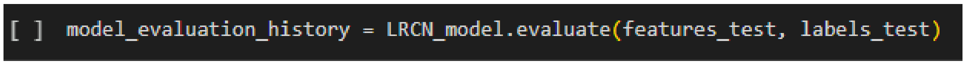


Figure 30 illustrated the progression of training and validation accuracy across epochs. Training accuracy (blue) exhibited consistent improvement, reaching around 0.95, indicating that the model effectively learns patterns in the training data. However, validation accuracy (red) fluctuated significantly and does not show a consistent upward trend. This suggests that while the model performed well on the training set, it struggles to generalize the unseen validation data. This pattern is a hallmark of overfitting and reinforces the use of early stopping. Additional regularization techniques or reconsideration of the validation split could further improve generalization.

After training, the model was evaluated using the reserved test dataset comprising 25% of the total data. The evaluate() function in Keras was used to obtain the final test accuracy and loss:

Figure 31

Evaluate() Function in Keras

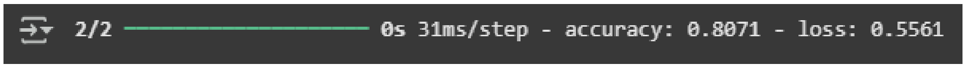


```
[ ]  model_evaluation_history = LRCN_model.evaluate(features_test, labels_test)
```

This yielded the following results:

Figure 32

LRCN Model Accuracy and Loss Results



```
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step - accuracy: 0.8071 - loss: 0.5561
```

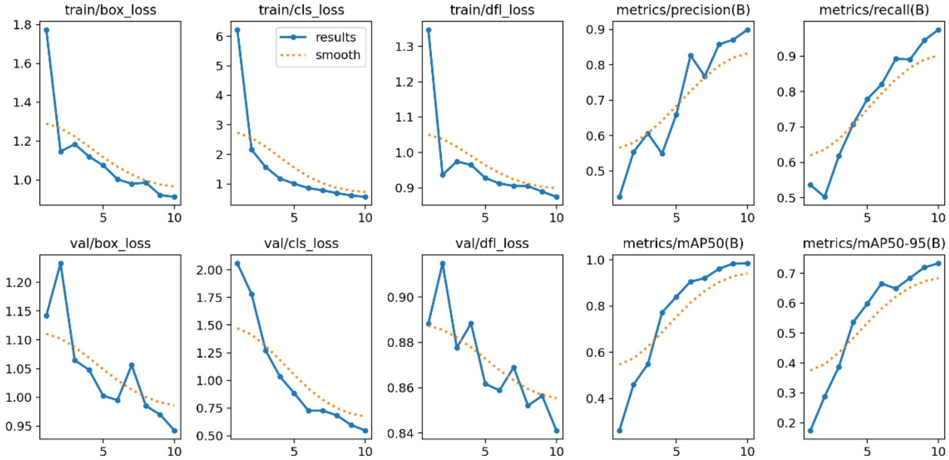
The model achieved an overall accuracy of **80.71%** on the test set, which demonstrates satisfactory performance in classifying unseen gameplay sequences. Although slightly lower than training accuracy, this result reflects a decent balance between fitting the training data and maintaining generalization.

b. You Only Look Once version 8 (YOLOv8) Model

The model was trained for ten (10) epochs using a dataset consisting of 15 predefined banner classes. The results were recorded in a CSV file generated by the YOLOv8 framework, which includes both training and validation losses, performance metrics (precision, recall, mean Average Precision), and learning rates.

Figure 33

YOLOv8 Model Training Results



The model’s performance was assessed using standard object detection metrics: precision, recall, mean Average Precision at Intersection at Union (IoU) threshold 0.5 (mAP@0.5), and mean Average Precision across IoU thresholds from 0.5 to 0.95 (mAP@0.5:0.95). The table below summarized the model's final evaluation scores from Epoch 10.

Table 4

YOLOv8 Banner Detection Metrics at Final Epoch

Test Case	Observed Results
Precision (B)	0.89883
Recall (B)	0.97503
mAP@0.5 (B)	0.98452
mAP@0.5:0.95 (B)	0.73341
Validation Box Loss	0.94240

Validation Class Loss	0.54679
Validation DFL Loss	0.84112

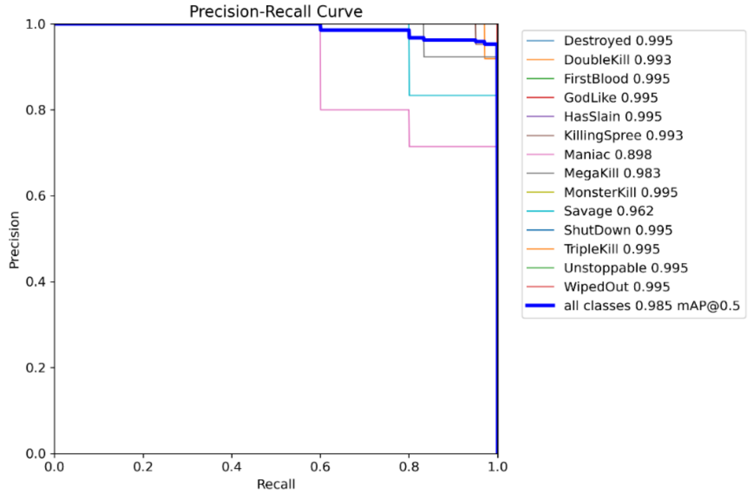
The model achieved a high 0.9 precision, which means nearly 90% of the predicted banners were accurate. Its recall of 0.98 indicates strong sensitivity, successfully identifying almost all actual banners. The mAP@0.5 of 0.98 reflected excellent localization capability at an IoU threshold of 0.5, while the mAP@0.5:0.95 of 0.73 signifies robust performance even under stricter evaluation conditions.

The validation losses for bounding box regression, classification, and distribution focal loss (0.94, 0.55, and 0.84 respectively) demonstrated that the model had a relatively low error rate, though further tuning could help reduce these losses.

To complement the numeric metrics, several visualizations were generated, including:

Figure 34

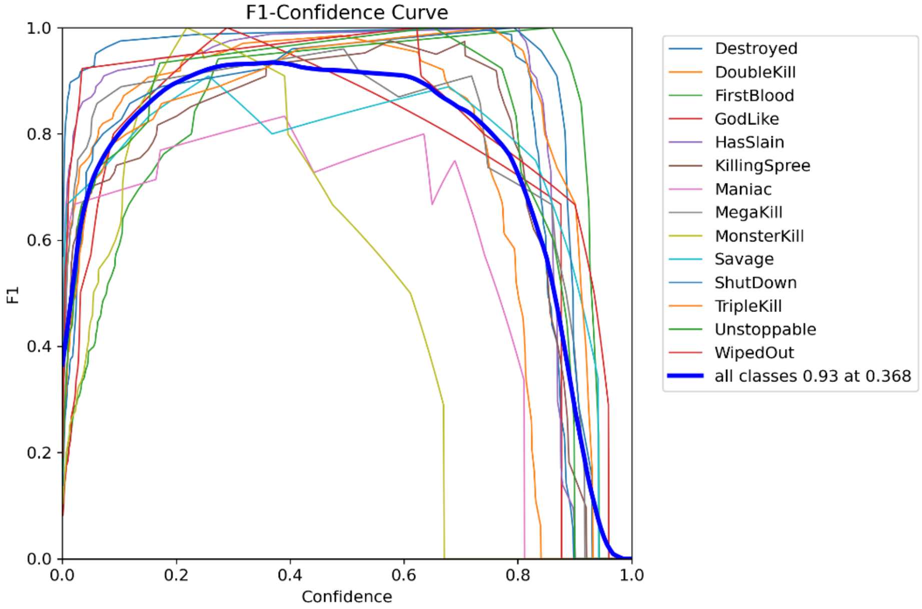
Precision-Recall Curve



- Precision-Recall (PR) Curve – Showed the relationship between precision and recall for all banner classes. The curve is close to the top-right corner, indicating high model reliability.

Figure 35

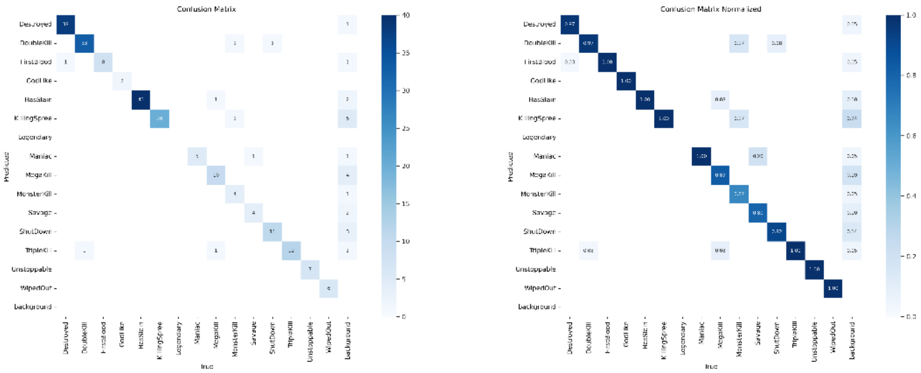
F1 Score Curve



- F1 Score Curve – demonstrated the score that balances precision and recall over the training epochs, indicating consistent performance across training (Ultralytics, 2025).

Figure 36

Confusion Matrix



- Confusion Matrix – Both raw (see Figure 36, left panel) and normalized (see Figure 36, right panel) versions highlighted strong class-wise prediction accuracy. The normalized confusion matrix showed a diagonal dominance, reflecting correct predictions across most classes.

Evaluation Results

Table 5

Summary of Evaluation

Criteria	Mean	Adjectival Rating
A. Functional Suitability		
1. Completeness	3.48	Highly Acceptable
2. Correctness	3.52	Highly Acceptable
3. Suitability	3.48	Highly Acceptable
Criterion Weighted Mean	*3.49*	*Highly Acceptable*
B. Performance Efficiency		
1. Time Behaviour	3.34	Highly Acceptable
2. Resource Utilization	3.34	Highly Acceptable
3. Capacity	3.44	Highly Acceptable
Criterion Weighted Mean	*3.37*	*Highly Acceptable*
C. Compatibility		
1. Co-existence	3.50	Highly Acceptable
Criterion Weighted Mean	*3.50*	*Highly Acceptable*
D. Usability		
1. Appropriate Recognizability	3.56	Highly Acceptable

2. Learnability	3.52	Highly Acceptable
3. Operability	3.64	Highly Acceptable
4. User Error Protection	3.34	Highly Acceptable
5. User Interface Aesthetics	3.40	Highly Acceptable
6. Accessibility	3.58	Highly Acceptable
Criterion Weighted Mean	*3.51*	*Highly Acceptable*
E. Reliability		
1. Maturity	3.28	Highly Acceptable
2. Fault Tolerance	3.38	Highly Acceptable
Criterion Weighted Mean	*3.33*	*Highly Acceptable*
F. Security		
1. Integrity	3.52	Highly Acceptable
Criterion Weighted Mean	*3.52*	*Highly Acceptable*
G. Maintainability		
1. Modifiability	3.56	Highly Acceptable
2. Modularity	3.48	Highly Acceptable
Criterion Weighted Mean	*3.52*	*Highly Acceptable*
H. Portability		
1. Installability	3.56	Highly Acceptable
Criterion Weighted Mean	*3.56*	*Highly Acceptable*
Grand Weighted Mean	**3.47**	**Highly Acceptable**

The evaluation of the developed system was carried out using the ISO/IEC 25010 software quality model, with results gathered through a 4-point Likert scale survey administered to selected respondents. The findings revealed that the system performed exceptionally across all quality characteristics. In terms of functional suitability, which includes completeness (M = 3.48), correctness (M = 3.52), and suitability (M = 3.48), the system achieved a weighted mean of 3.49, which indicated that it satisfied the specified functional requirements and is highly acceptable. Under performance efficiency, the system's time behavior (M = 3.34), resource utilization (M = 3.34), and capacity (M = 3.44) led to a weighted mean of 3.37, suggesting efficient operation with minimal resource usage. For compatibility, the system received a mean rating of 3.50 in terms of co-existence, confirming its seamless integration with other systems.

In the area of usability, the system showed strong performance across several sub-characteristics: appropriate recognizability (M = 3.56), learnability (M = 3.52), operability (M = 3.64), user error protection (M = 3.34), user interface aesthetics (M = 3.40), and accessibility (M = 3.58), culminating in a criterion weighted mean of 3.51. These results imply that the system is user-friendly and easy to use. Regarding reliability, which evaluated the stability and fault tolerance of the system, the maturity (M = 3.28) and fault tolerance (M = 3.38) ratings yielded a weighted mean of 3.33, indicating consistent and dependable performance. The security of the system, measured by integrity, was also deemed highly acceptable with a mean of 3.52, reflecting strong prevention of unauthorized access and data breaches.

In terms of maintainability, which involved ease of modification and modular structure, the system scored 3.56 in modifiability and 3.48 in modularity, with a weighted mean of 3.52, indicating ease of updates and maintenance. The portability of the system, evaluated through installability, was rated at 3.56 highlighting the ease of system deployment across various platforms. Overall, the grand weighted mean was computed at 3.47, which fell under the category of Highly Acceptable. These results affirmed that the system effectively met the comprehensive software quality criteria, demonstrating its readiness for practical implementation and deployment.

Chapter 5

SUMMARY OF FINDINGS, CONCLUSIONS AND RECOMMENDATIONS

This chapter put forward the summary of findings, the conclusions reached based on them, and the recommendations for further enhancement of the system.

Summary of Findings

Based on the test result, the developed application entitled “Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang” can generate video highlights for Mobile Legends: Bang Bang game. The following is the summary of findings of the developed application:

- Test Result

1. **Functional Suitability:** System underwent a thorough perception of its core capabilities, focusing on how users could input video clips, file selection and drag-and-drop methods. This includes evaluating its handling file types and multiple inputs. Generate button functions effectively handled acceptable videos by prompting sorting preferences and showing clear progress updates, while providing appropriate error messages when no valid files were chosen or when unsupported clips are used. The video editor interface allowed users to navigate through clips, keep or delete them with clear visual cues, and enables navigation controls only after user actions. Playback features such as restart, pause, and speed adjustment worked as expected. Lastly, the concatenate function successfully compiled selected clips into a final video file, providing

status updates throughout the process. Overall, the system reliably met the expected functional requirements with an intuitive user experience.

2. **Machine Learning Model Evaluation**

   a. **Long-term Recurrent Convolutional Network (LRCN) Model:** The assessment of the LRCN model's performance highlights the overall accuracy of 80.71% achieved by the test set which was 25% of the dataset partitioned using stratified sampling. The remaining 75% were trained and reached 95% accuracy which indicated the model learned patterns in the data effectively. The accuracy achieved by the test set being higher than the training set is an acceptable balance between fitting the training data and maintaining generalization.

   b. **You Only Look Once version 8 (YOLOv8) Model:** The model was trained and achieved an accuracy of 90% (0.90) in predicting banners. This precision means that 90% of the predicted banners are correct. The model also achieved a recall of 0.98 which means that 98% of the actual banners were predicted. The PR Curve indicated that as the precision and recall is high, the reliability of the model is also high.

- Evaluation Result

1. **Functional Suitability.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its function to generate useful and engaging game highlights accurately reflecting the actual key moments from the MLBB.

2. **Performance Efficiency.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its generating highlights in a reasonable

amount of time. The system also processed highlights using the supported devices' resources efficiently.

3. **Compatibility.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its co-existence alongside other applications running on the devices.
4. **Usability.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its ability that made it easy for the user to recognize the purpose and use of the system. The user also used the system itself effectively. It also protects the user from errors by preventing them or helping the user recover from them.
5. **Reliability.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its performance that is consistent without errors (e.g. incorrect input, interrupted video).
6. **Security.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its integrity on the raw gameplay video uploaded by the user.
7. **Maintainability.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its modifiability based on the response of the user without affecting the system's other features.
8. **Portability.** The evaluators rated the Video Highlight Generator as "Highly Acceptable" regarding its ability to be installed on the supported devices.

Conclusions

Based on the evaluation and finding of the tests, the researchers reached the following conclusions:

1. The Video Highlight Generator was successfully designed to have the following functions such as:
   a. The system allowed the users to upload gameplays, then the system processes them and generate highlights from them with an appealing user interface.
   b. The system allowed the users to opt for sorting the highlights chronologically or by priority, their scores based on the excitement level of the detected banners, before they are displayed to the users.
   c. The system allowed the user to freely review the displayed highlights, having the choice to keep or delete them.
2. The Video Highlight Generator was successfully developed based on the LRCN architecture, combining the implementation of CNN and LSTM to detect team fights. YOLOv8 was also implemented to detect banners from those team fights. Both implementations were trained in Google Colab and then integrated into a Python-based frontend application.
   a. The Python-based GUI application was developed using *customtkinter*, *tkinterdnd2*, *pillow*, *tkVideo*, *opencv-python*, *moviepy*, and others.
3. Based on the eight quality characteristics divided from the product quality aspect of the ISO 25010, the developed system was evaluated to be Highly Acceptable.

Recommendations

The study puts forward the following recommendations for further system enhancements:

1. Provide additional testing to further ensure support for devices such as those without graphics cards.
2. Further optimize runtime on devices with just their processors.
3. Consider further optimizing runtime by providing real-time detection of final highlights.
4. Further optimize runtime by exploring other implementations.
5. Further improve the system's UI by adding more color to the palette and enhancing the overall design. It could be helpful especially if the system is planned to be deployed for general consumers.
6. Consider annotations or indications of transition and timing to ensure which segment is being displayed. These apply to video circulation that will be displaying the final highlights. This could be good for the users who review the highlights. It usually ranges from 1 to 3 users who are assigned to review.
7. Consider generalization of the system by adding more classes and banner data from clips of gameplay videos with other layouts and fonts. Other versions of the game may also be considered.
8. Generalize the system by creating classes and banner data from clips and banners of other games like MLBB.

APPENDIX A

EVALUATION SHEET

Technological University of the Philippines
College of Science
Computer Studies Department

Name (optional): ____

Demographic:
○ MLBB Player or Esports Manager
○ MLBB Video Highlights Editor (Creatives Committee)
○ Esports Organization Officer
○ Others: ____________________

Direction: Please evaluate the project "Video Highlight Generator utilizing a Convolutional Neural Network (CNN) combined with a Long Short-Term Memory (LSTM) network For Mobile Legends: Bang Bang" by using the given scale and placing a checkmark (✓) under the corresponding numerical rating.

4 – Highly Acceptable 3 – Very Acceptable 2 – Acceptable 1 – Not Acceptable

INDICATORS	**4**	**3**	**2**	**1**
A. Functional Suitability				
Completeness The system provides all the necessary features and functionalities to generate game highlights from Mobile Legends: Bang Bang.				
Correctness The highlights generated accurately reflect the actual key moments from the gameplay.				
Sustainability The system is appropriate for the task of creating useful and engaging Mobile Legends Highlights.				
B. Performance Efficiency				
Time Behavior The system processes and generates highlights in a reasonable amount of time.				
Resource Utilization The system efficiently uses my device's resources (e.g., CPU, GPU, Memory) during the processing.				
Capacity The system handles large or multiple video files without crashing or slowing down.				

C. Compatibility				
Co – Existence The system works well alongside other applications running on my devices.				
D. Usability				
Appropriateness Recognizability It is easy to recognize the purpose and use of the system when I first open it.				
Learnability I can learn how to use the system quickly without much help or instructions.				
Operability The system is easy to navigate and operate.				
User Error Protection The system prevents errors or helps me recover from mistakes easily.				
User Interface Aesthetics The system visually appealing and well-designed.				
Accessibility Even though I have different levels of experience and abilities, I can still use the system effectively.				
E. Reliability				
Maturity The system performs consistently without unexpected errors.				
Fault Tolerance The system can recover or continue working after minor issues (e.g., incorrect input, interrupted video)?				
F. Security				
Integrity I feel my data (uploaded videos) is safe and not altered or misused by the system?				
G. Maintainability				
Modifiability The system can be easily updated or improved based on feedback or changing needs.				
Modularity The system is organized in a way that makes its parts easy to understand or change independently.				
H. Portability				
Instability The system was easy to install and set up on my device.				

Comments/Suggestions:

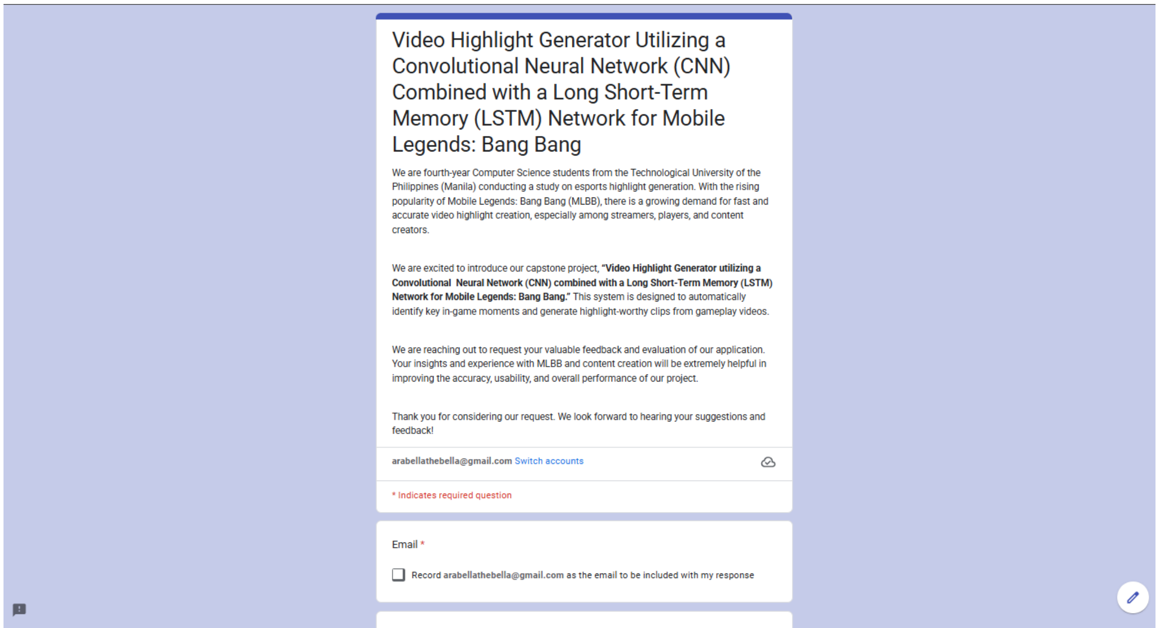
____

____

____

APPENDIX B

SURVEY QUESTIONNAIRE



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

We are fourth-year Computer Science students from the Technological University of the Philippines (Manila) conducting a study on esports highlight generation. With the rising popularity of Mobile Legends: Bang Bang (MLBB), there is a growing demand for fast and accurate video highlight creation, especially among streamers, players, and content creators.

We are excited to introduce our capstone project, **"Video Highlight Generator utilizing a Convolutional Neural Network (CNN) combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang."** This system is designed to automatically identify key in-game moments and generate highlight-worthy clips from gameplay videos.

We are reaching out to request your valuable feedback and evaluation of our application. Your insights and experience with MLBB and content creation will be extremely helpful in improving the accuracy, usability, and overall performance of our project.

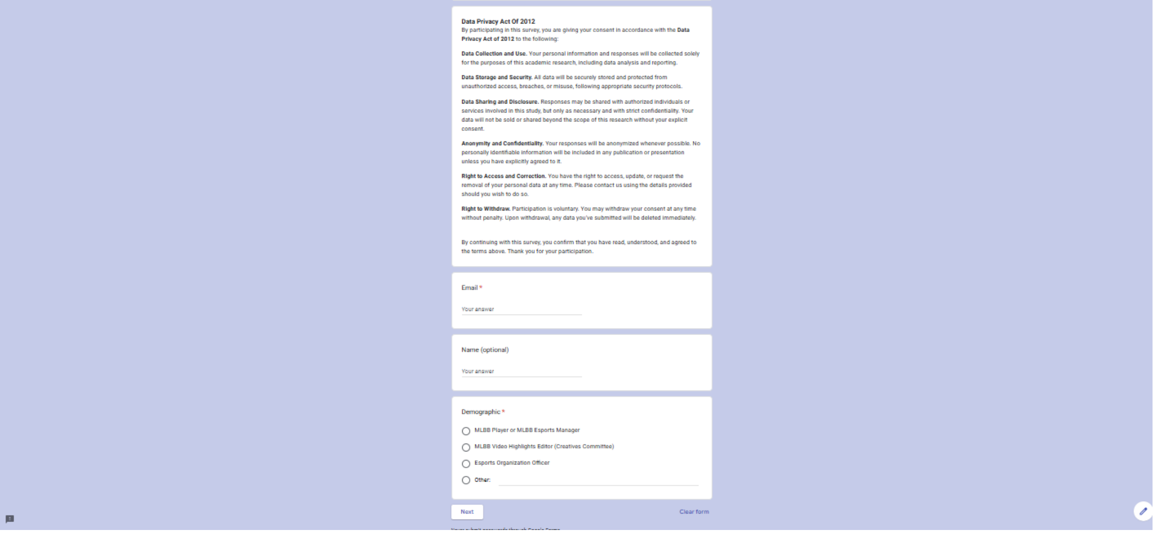
Thank you for considering our request. We look forward to hearing your suggestions and feedback!

arabellathebella@gmail.com Switch accounts

* Indicates required question

Email *

□ Record **arabellathebella@gmail.com** as the email to be included with my response



Data Privacy Act Of 2012
By participating in this survey, you are giving your consent in accordance with the **Data Privacy Act of 2012** to the following:

Data Collection and Use. Your personal information and responses will be collected solely for the purposes of this academic research, including data analysis and reporting.

Data Storage and Security. All data will be securely stored and protected from unauthorized access, breaches, or misuse, following appropriate security protocols.

Data Sharing and Disclosure. Responses may be shared with authorized individuals or services involved in this study, but only as necessary and with strict confidentiality. Your data will not be sold or shared beyond the scope of this research without your explicit consent.

Anonymity and Confidentiality. Your responses will be anonymized whenever possible. No personally identifiable information will be included in any publication or presentation unless you have explicitly agreed to it.

Right to Access and Correction. You have the right to access, update, or request the removal of your personal data at any time. Please contact us using the details provided should you wish to do so.

Right to Withdraw. Participation is voluntary. You may withdraw your consent at any time without penalty. Upon withdrawal, any data you've submitted will be deleted immediately.

By continuing with this survey, you confirm that you have read, understood, and agreed to the terms above. Thank you for your participation.

Email *

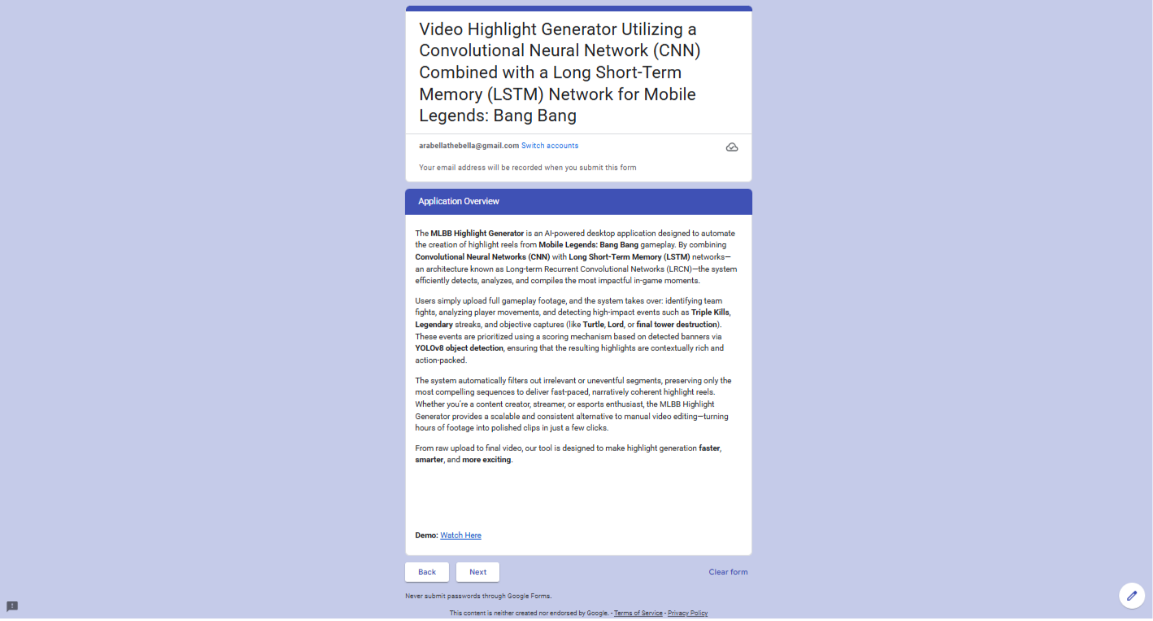
Your answer

Name (optional)

Your answer

Demographic *

○ MLBB Player or MLBB Esports Manager
○ MLBB Video Highlights Editor (Creatives Committee)
○ Esports Organization Officer
○ Other: __________



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

arabellathebella@gmail.com Switch accounts

Your email address will be recorded when you submit this form

Application Overview

The **MLBB Highlight Generator** is an AI-powered desktop application designed to automate the creation of highlight reels from **Mobile Legends: Bang Bang** gameplay. By combining **Convolutional Neural Networks (CNN)** with **Long Short-Term Memory (LSTM)** networks—an architecture known as Long-term Recurrent Convolutional Networks (LRCN)—the system efficiently detects, analyzes, and compiles the most impactful in-game moments.

Users simply upload full gameplay footage, and the system takes over: identifying team fights, analyzing player movements, and detecting high-impact events such as **Triple Kills**, **Legendary** streaks, and objective captures (like **Turtle**, **Lord**, or **final tower destruction**). These events are prioritized using a scoring mechanism based on detected banners via **YOLOv8 object detection**, ensuring that the resulting highlights are contextually rich and action-packed.

The system automatically filters out irrelevant or uneventful segments, preserving only the most compelling sequences to deliver fast-paced, narratively coherent highlight reels. Whether you're a content creator, streamer, or esports enthusiast, the MLBB Highlight Generator provides a scalable and consistent alternative to manual video editing—turning hours of footage into polished clips in just a few clicks.

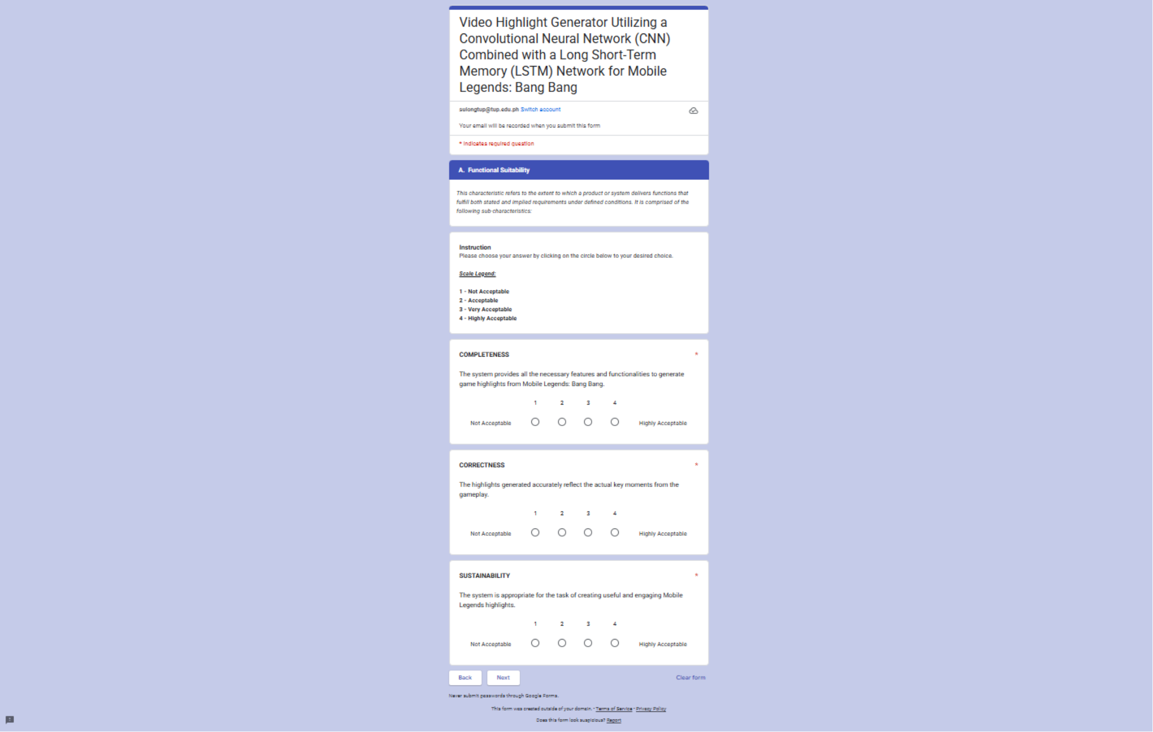
From raw upload to final video, our tool is designed to make highlight generation **faster**, **smarter**, and **more exciting**.

Demo: Watch Here

Back Next Clear form

Never submit passwords through Google Forms.

This content is neither created nor endorsed by Google. - Terms of Service - Privacy Policy



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

A. Functional Suitability

This characteristic refers to the extent to which a product or system delivers functions that fulfill both stated and implied requirements under defined conditions. It is comprised of the following sub-characteristics:

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

Scale Legend:

1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

COMPLETENESS *

The system provides all the necessary features and functionalities to generate game highlights from Mobile Legends: Bang Bang.

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

CORRECTNESS *

The highlights generated accurately reflect the actual key moments from the gameplay.

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

SUSTAINABILITY *

The system is appropriate for the task of creating useful and engaging Mobile Legends highlights.

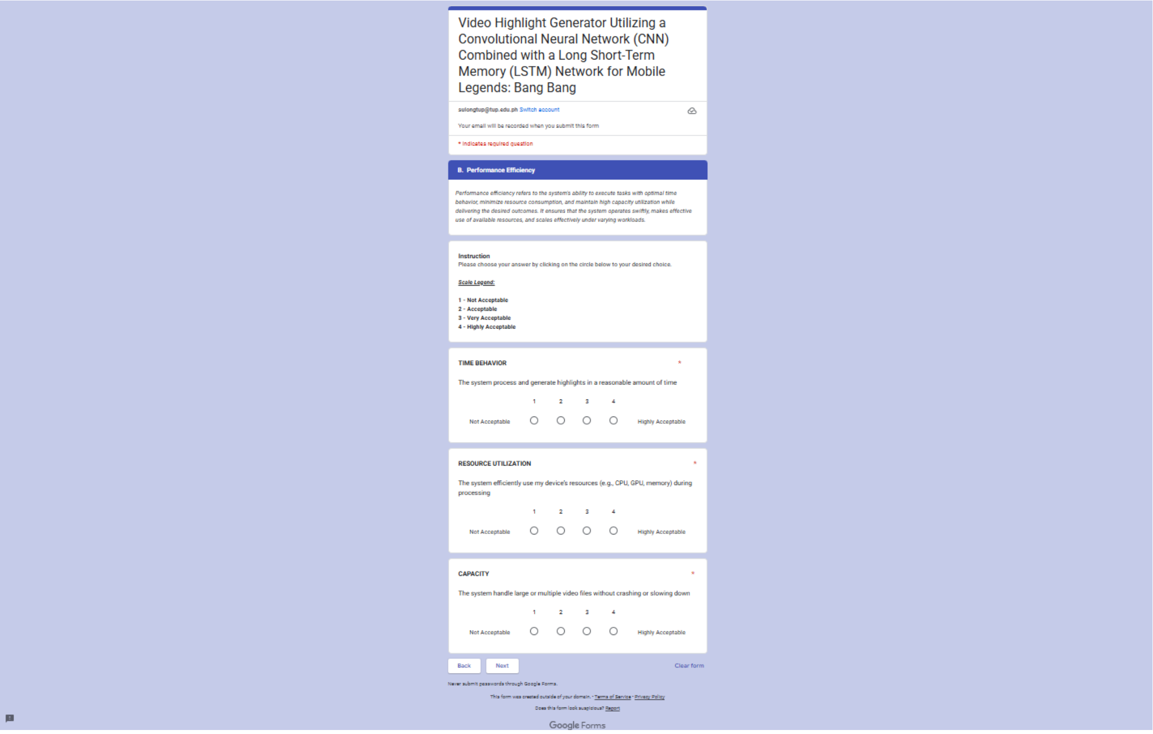
	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

Back Next Clear form

Never submit passwords through Google Forms.

This form was created outside of your domain. - Terms of Service - Privacy Policy

Does this form look suspicious? Report



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

B. Performance Efficiency

Performance efficiency refers to the system's ability to execute tasks with optimal time behavior, minimize resource consumption, and maintain high capacity utilization while delivering the desired outcomes. It ensures that the system operates swiftly, makes effective use of available resources, and scales effectively under varying workloads.

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

Scale Legend:

1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

TIME BEHAVIOR *

The system process and generate highlights in a reasonable amount of time

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

RESOURCE UTILIZATION *

The system efficiently use my device's resources (e.g., CPU, GPU, memory) during processing

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

CAPACITY *

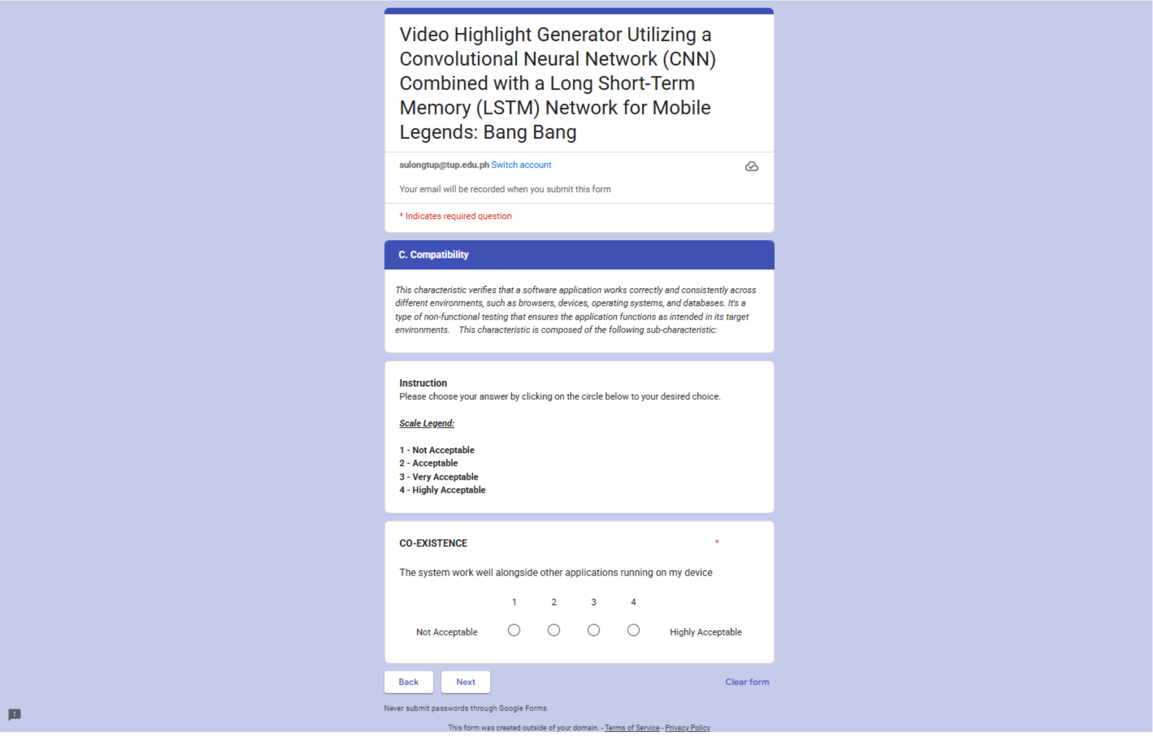
The system handle large or multiple video files without crashing or slowing down

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

Never submit passwords through Google Forms.

This form was created outside of your domain. - Terms of Service - Privacy Policy

Does this form look suspicious? Report



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

C. Compatibility

This characteristic verifies that a software application works correctly and consistently across different environments, such as browsers, devices, operating systems, and databases. It's a type of non-functional testing that ensures the application functions as intended in its target environments. This characteristic is composed of the following sub-characteristic:

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

Scale Legend:

1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

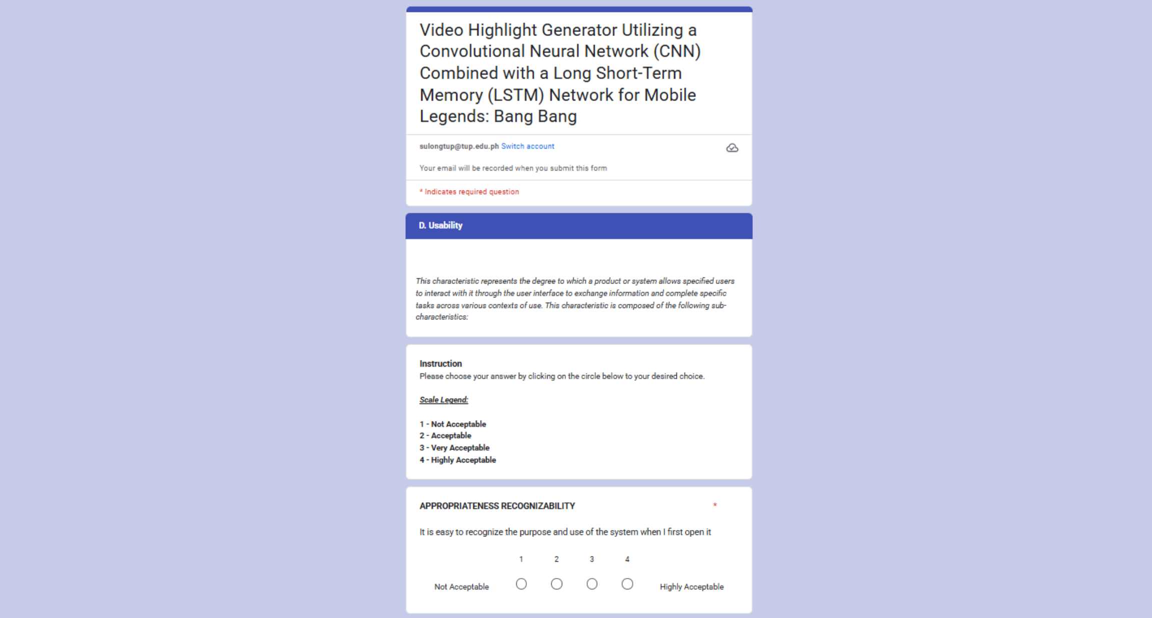
CO-EXISTENCE *

The system work well alongside other applications running on my device

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

Never submit passwords through Google Forms.

This form was created outside of your domain. - Terms of Service - Privacy Policy



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

D. Usability

This characteristic represents the degree to which a product or system allows specified users to interact with it through the user interface to exchange information and complete specific tasks across various contexts of use. This characteristic is composed of the following sub-characteristics:

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

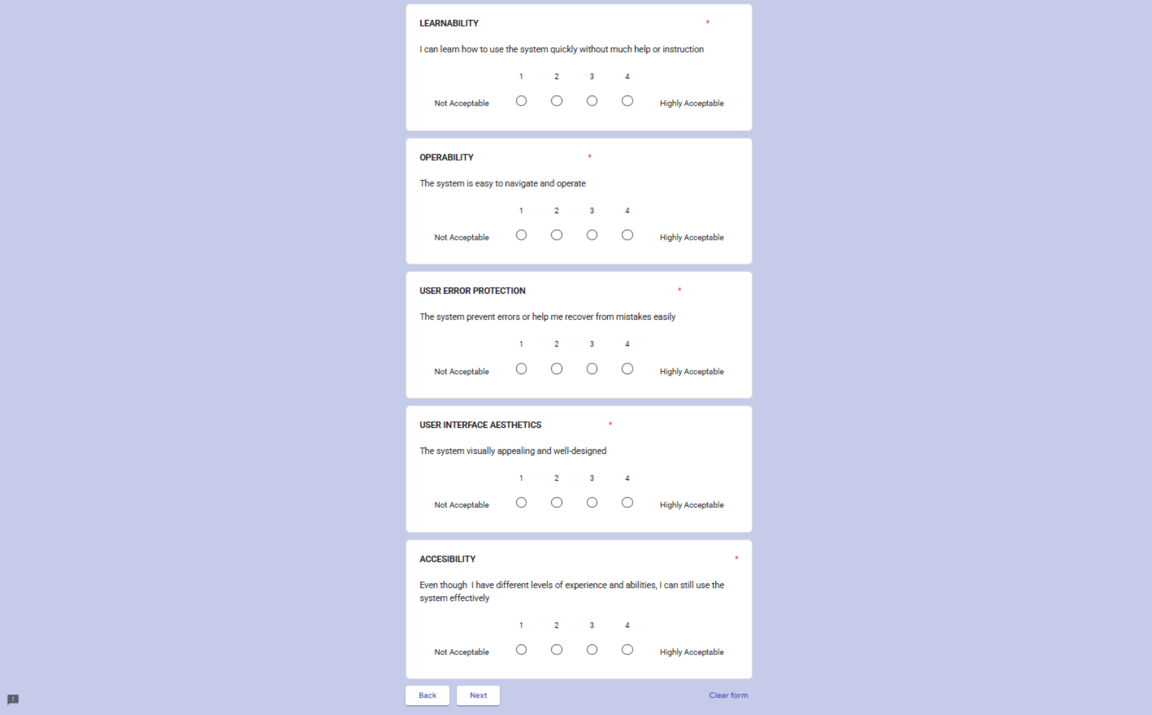
Scale Legend:

1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

APPROPRIATENESS RECOGNIZABILITY *

It is easy to recognize the purpose and use of the system when I first open it

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable



LEARNABILITY *

I can learn how to use the system quickly without much help or instruction

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

OPERABILITY *

The system is easy to navigate and operate

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

USER ERROR PROTECTION *

The system prevent errors or help me recover from mistakes easily

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

USER INTERFACE AESTHETICS *

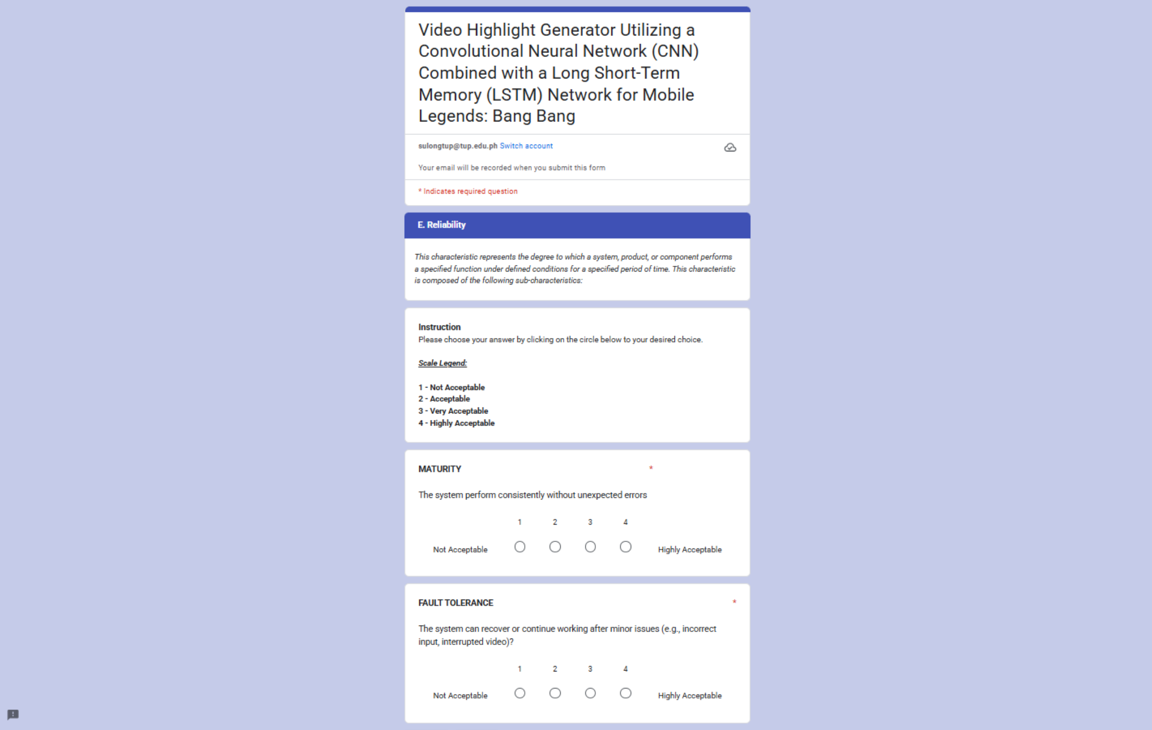
The system visually appealing and well-designed

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

ACCESIBILITY *

Even though I have different levels of experience and abilities, I can still use the system effectively

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

E. Reliability

This characteristic represents the degree to which a system, product, or component performs a specified function under defined conditions for a specified period of time. This characteristic is composed of the following sub-characteristics:

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

Scale Legend:

1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

MATURITY *

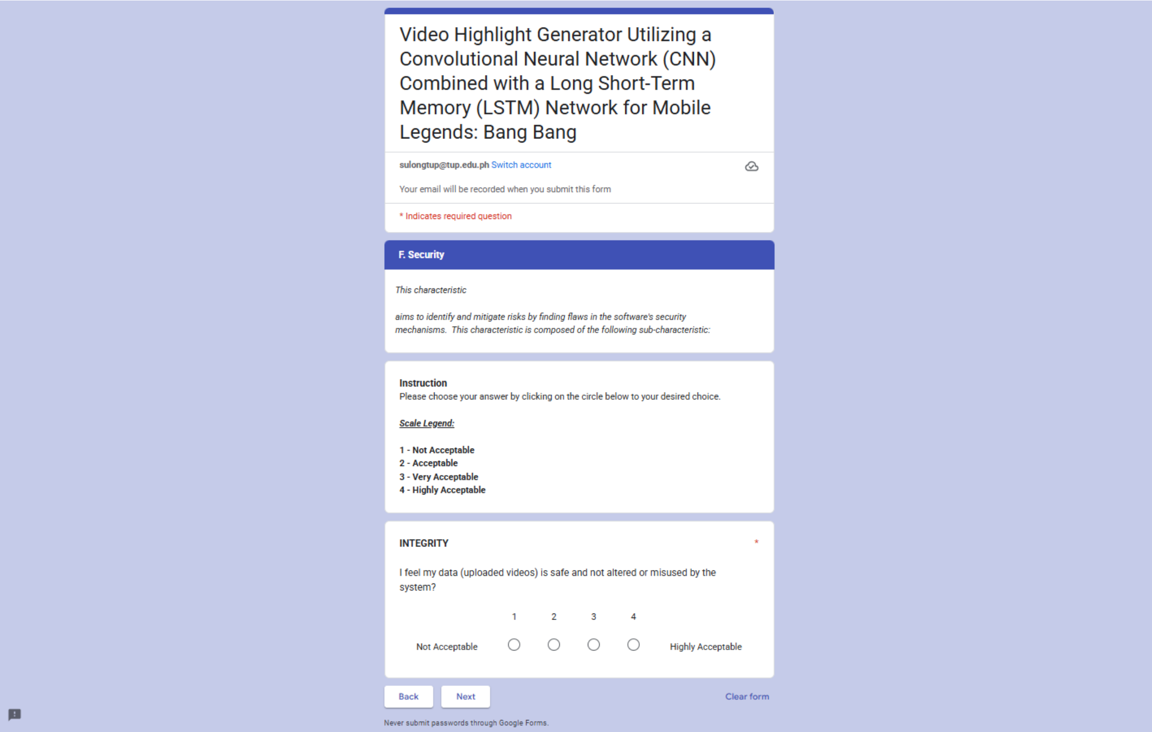
The system perform consistently without unexpected errors

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

FAULT TOLERANCE *

The system can recover or continue working after minor issues (e.g., incorrect input, interrupted video)?

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

F. Security

This characteristic

aims to identify and mitigate risks by finding flaws in the software's security mechanisms. This characteristic is composed of the following sub-characteristic:

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

Scale Legend:

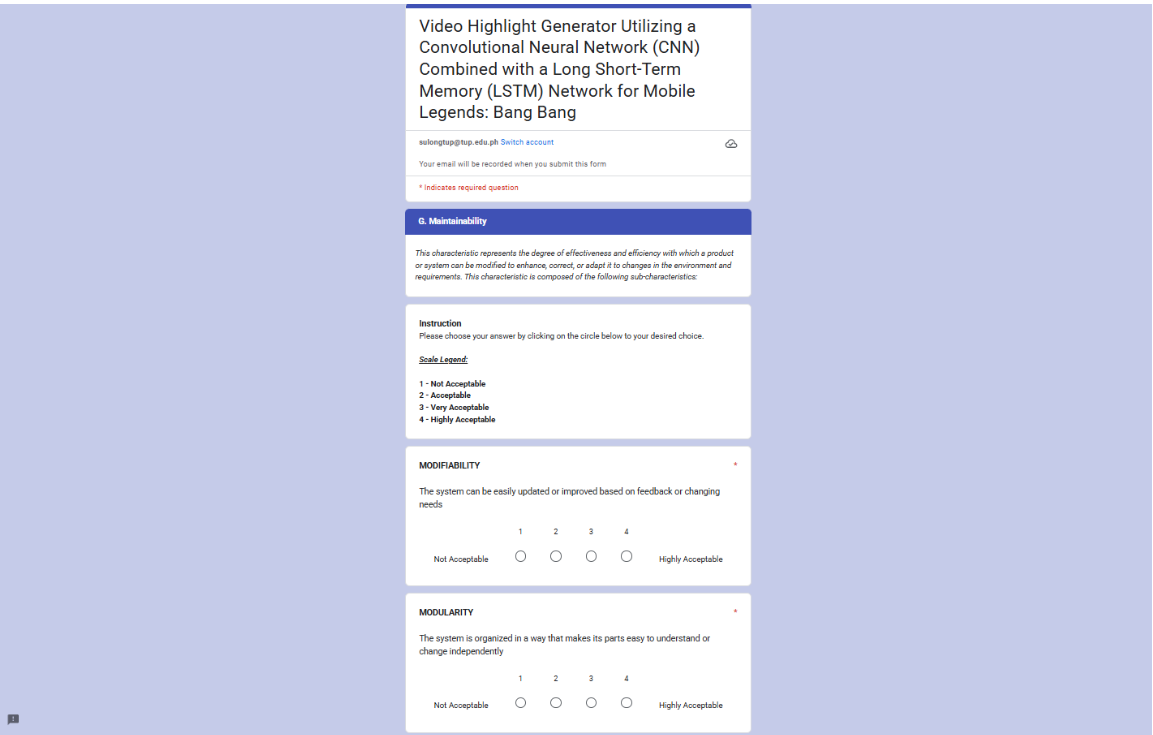
1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

INTEGRITY *

I feel my data (uploaded videos) is safe and not altered or misused by the system?

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

Never submit passwords through Google Forms.



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

G. Maintainability

This characteristic represents the degree of effectiveness and efficiency with which a product or system can be modified to enhance, correct, or adapt it to changes in the environment and requirements. This characteristic is composed of the following sub-characteristics:

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

Scale Legend:

1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

MODIFIABILITY *

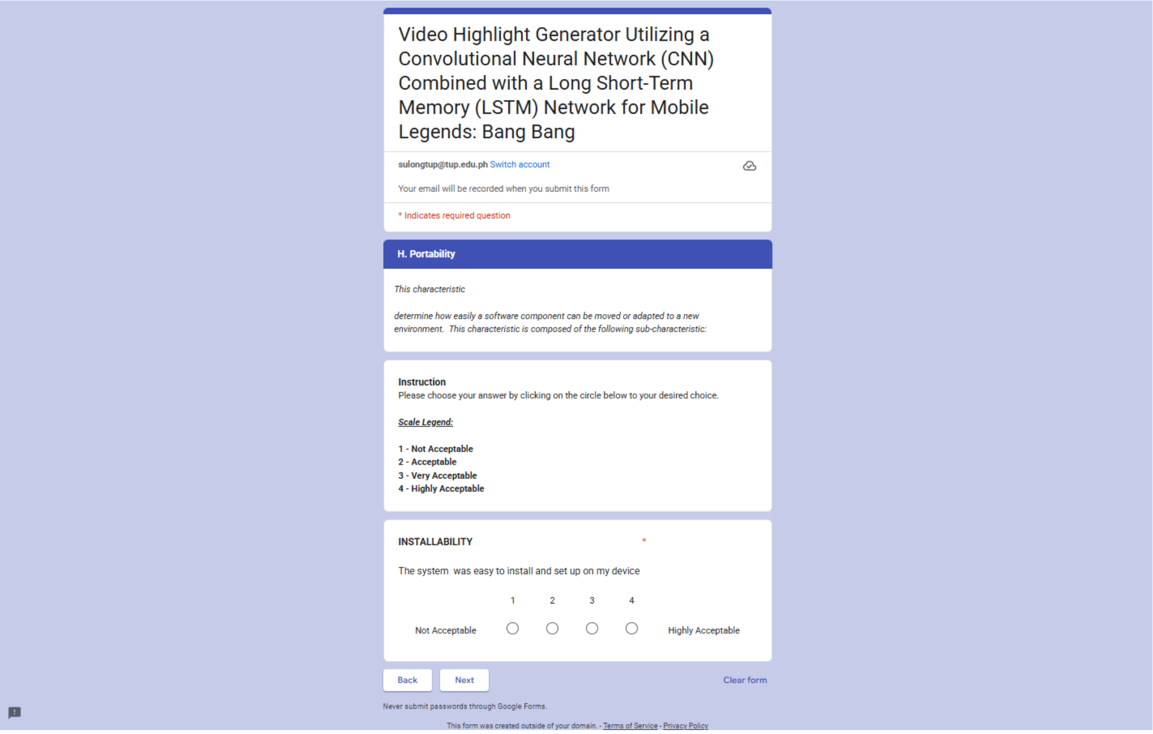
The system can be easily updated or improved based on feedback or changing needs

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

MODULARITY *

The system is organized in a way that makes its parts easy to understand or change independently

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

* Indicates required question

H. Portability

This characteristic

determine how easily a software component can be moved or adapted to a new environment. This characteristic is composed of the following sub-characteristic:

Instruction
Please choose your answer by clicking on the circle below to your desired choice.

Scale Legend:

1 - Not Acceptable
2 - Acceptable
3 - Very Acceptable
4 - Highly Acceptable

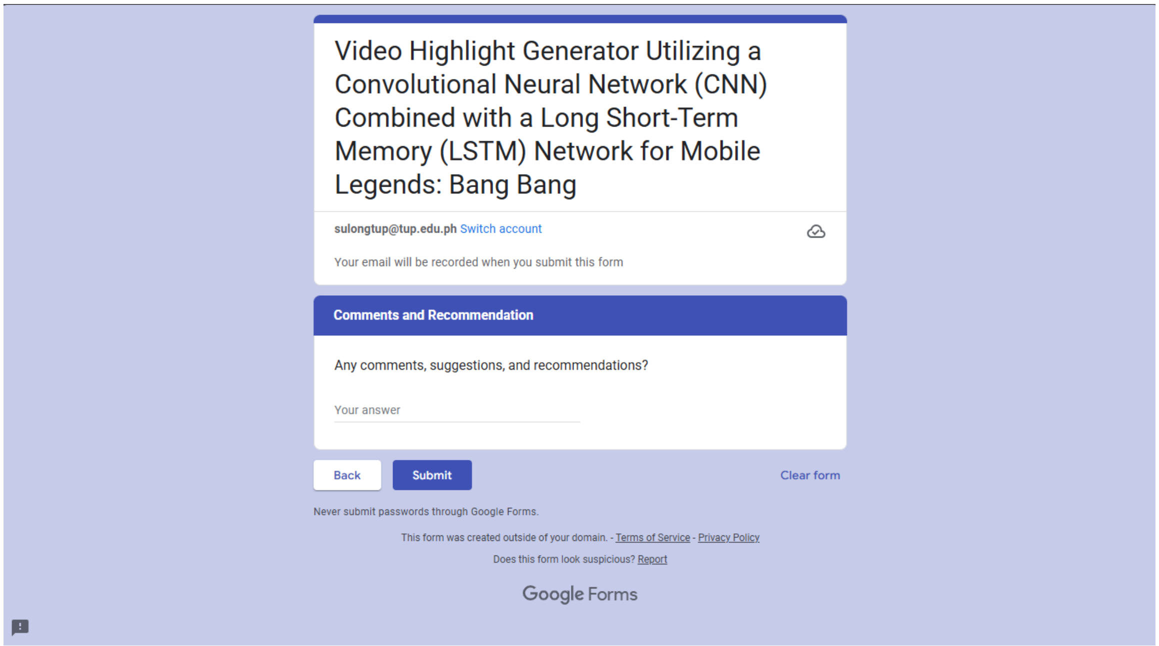
INSTALLABILITY *

The system was easy to install and set up on my device

	1	2	3	4	
Not Acceptable	○	○	○	○	Highly Acceptable

Never submit passwords through Google Forms.

This form was created outside of your domain. - Terms of Service - Privacy Policy



Video Highlight Generator Utilizing a Convolutional Neural Network (CNN) Combined with a Long Short-Term Memory (LSTM) Network for Mobile Legends: Bang Bang

sulongtup@tup.edu.ph Switch account

Your email will be recorded when you submit this form

Comments and Recommendation

Any comments, suggestions, and recommendations?

Your answer

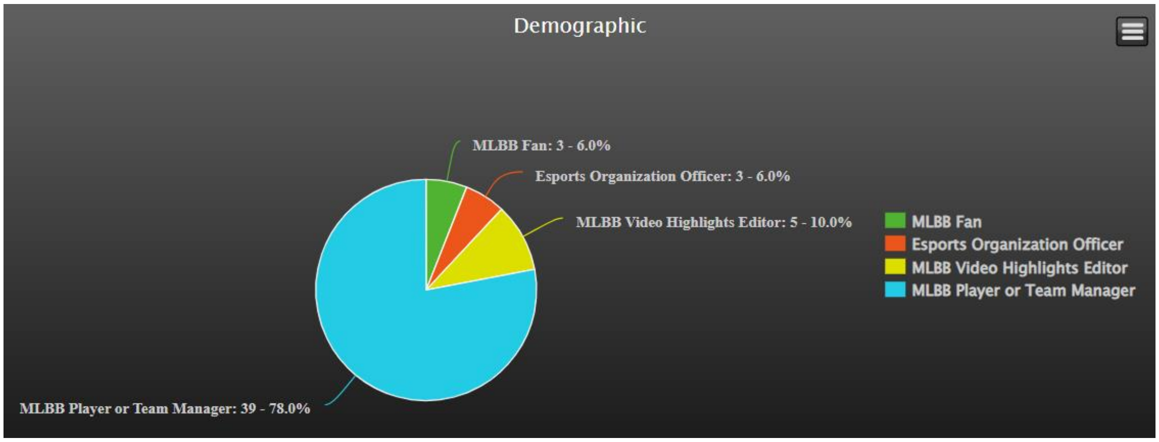
Never submit passwords through Google Forms.

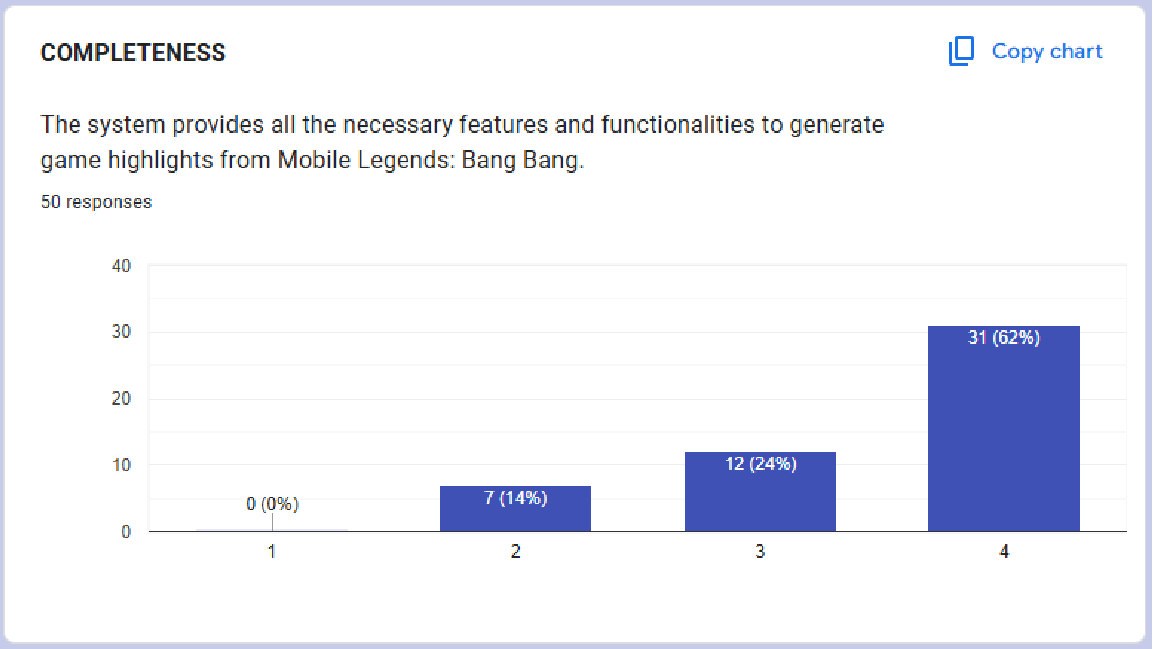
This form was created outside of your domain. - Terms of Service - Privacy Policy

Does this form look suspicious? Report

APPENDIX C

RESULTS SHEETS

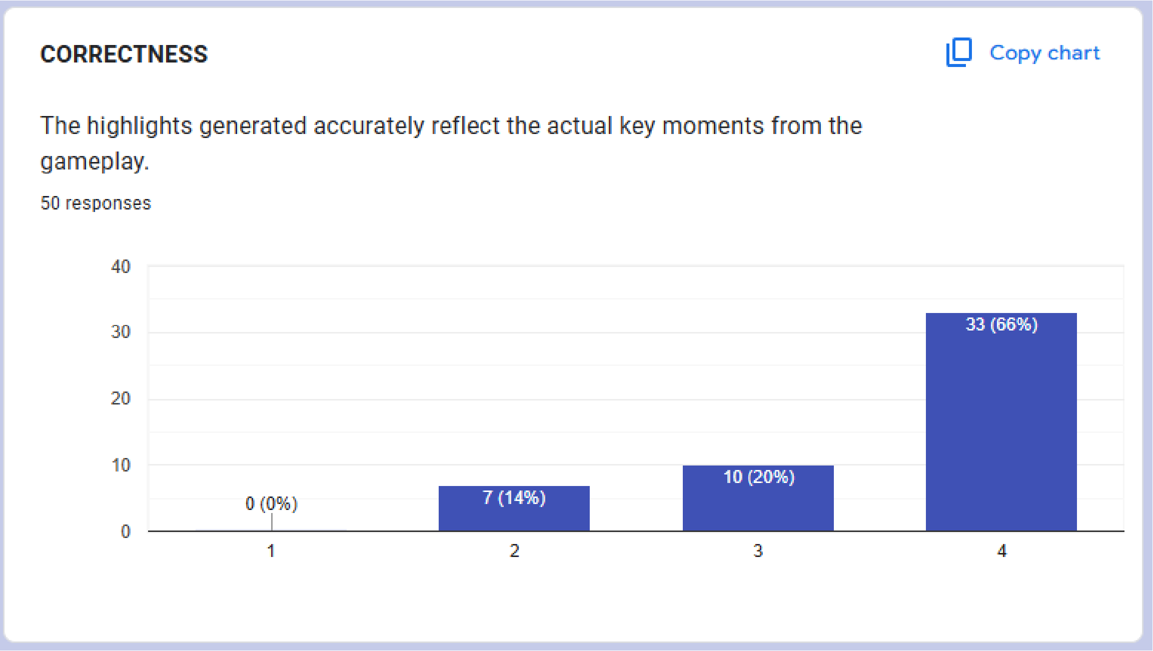




COMPLETENESS

The system provides all the necessary features and functionalities to generate game highlights from Mobile Legends: Bang Bang.

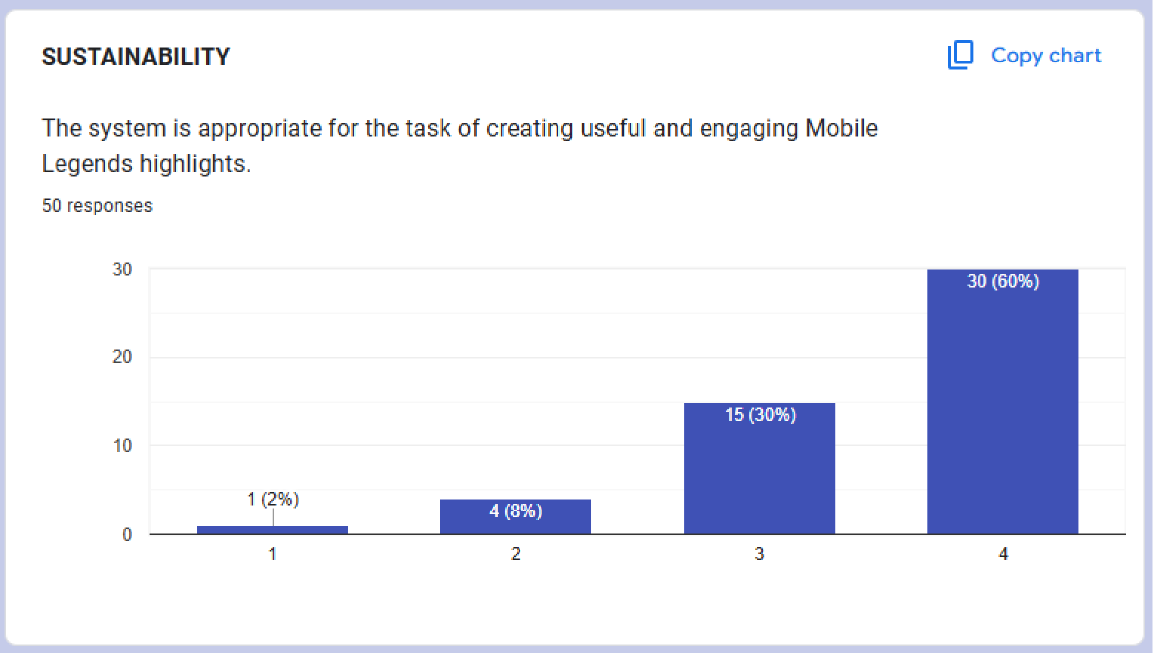
50 responses



CORRECTNESS

The highlights generated accurately reflect the actual key moments from the gameplay.

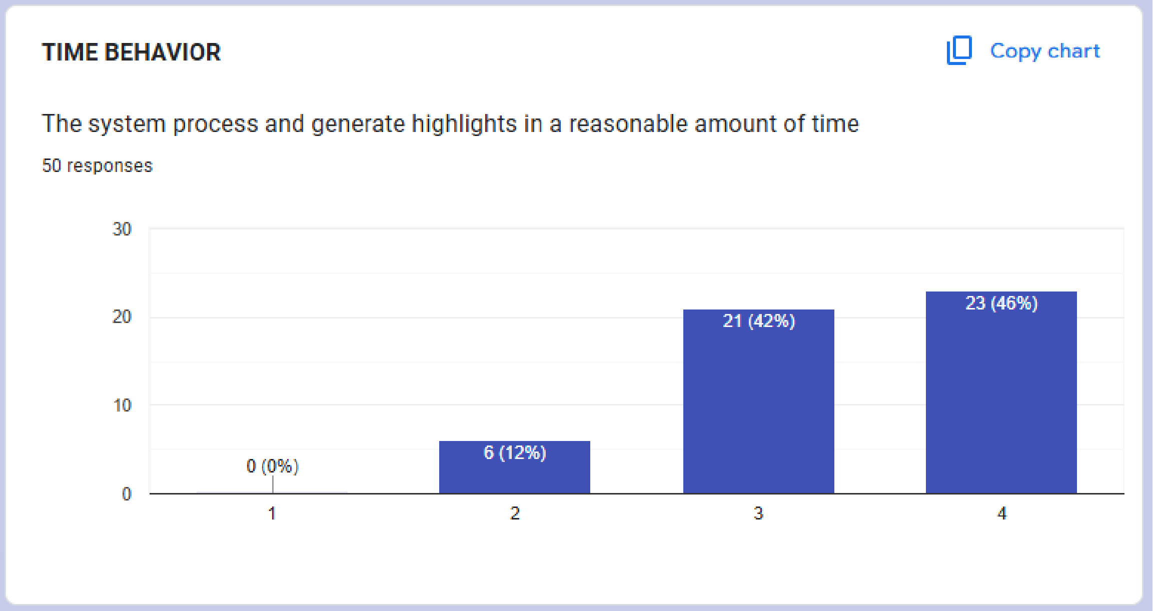
50 responses



SUSTAINABILITY

The system is appropriate for the task of creating useful and engaging Mobile Legends highlights.

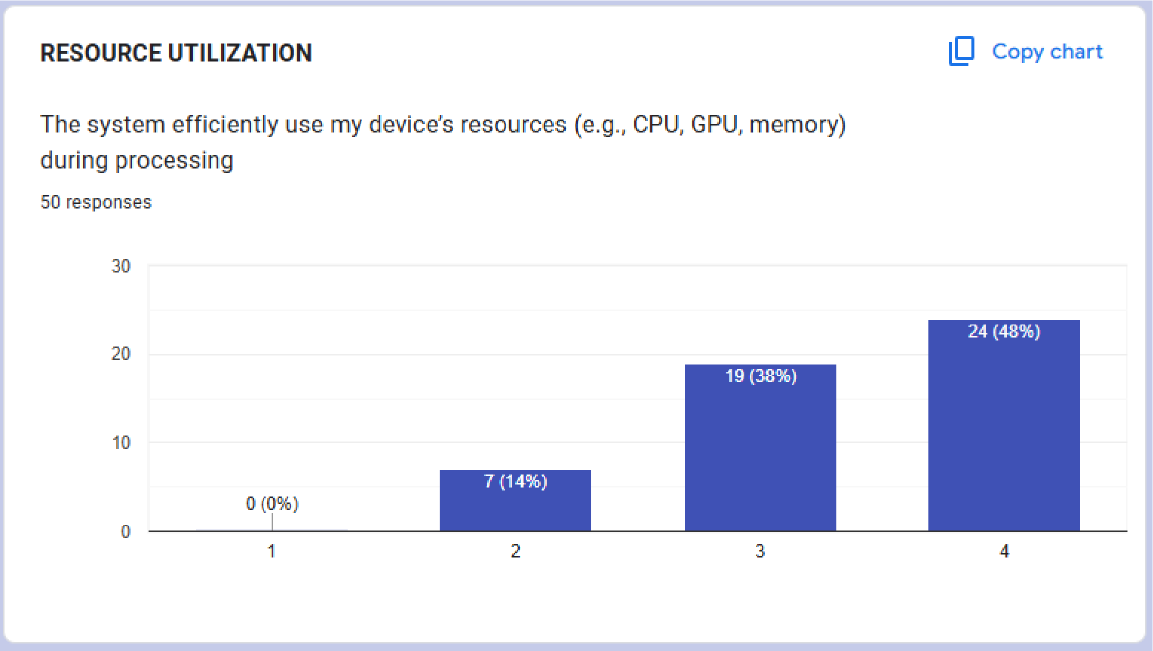
50 responses



TIME BEHAVIOR

The system process and generate highlights in a reasonable amount of time

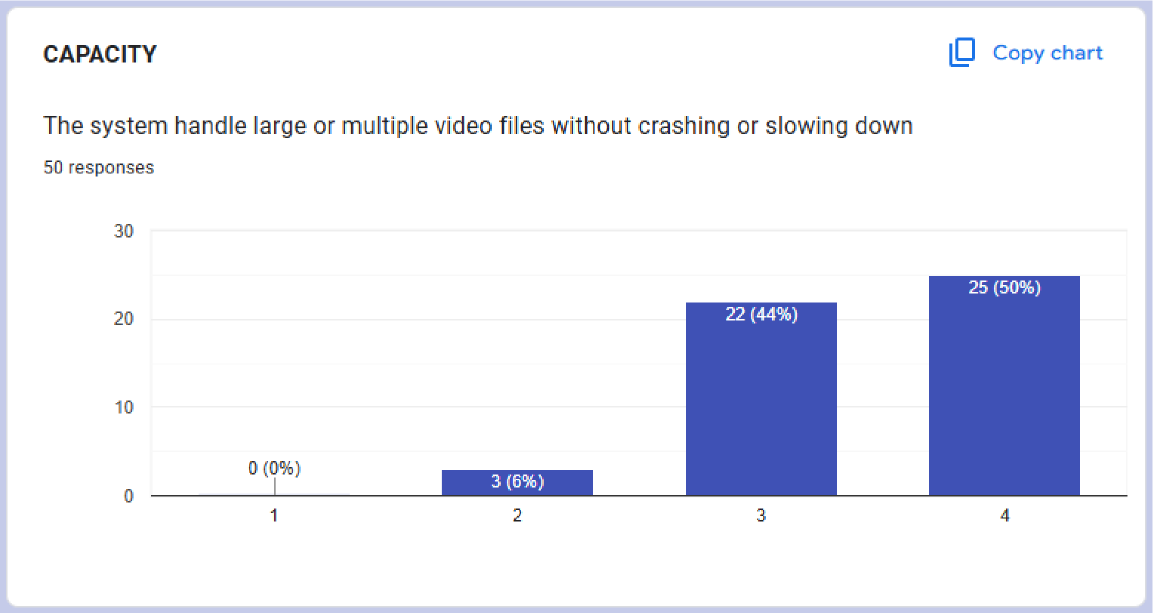
50 responses



RESOURCE UTILIZATION

The system efficiently use my device's resources (e.g., CPU, GPU, memory) during processing

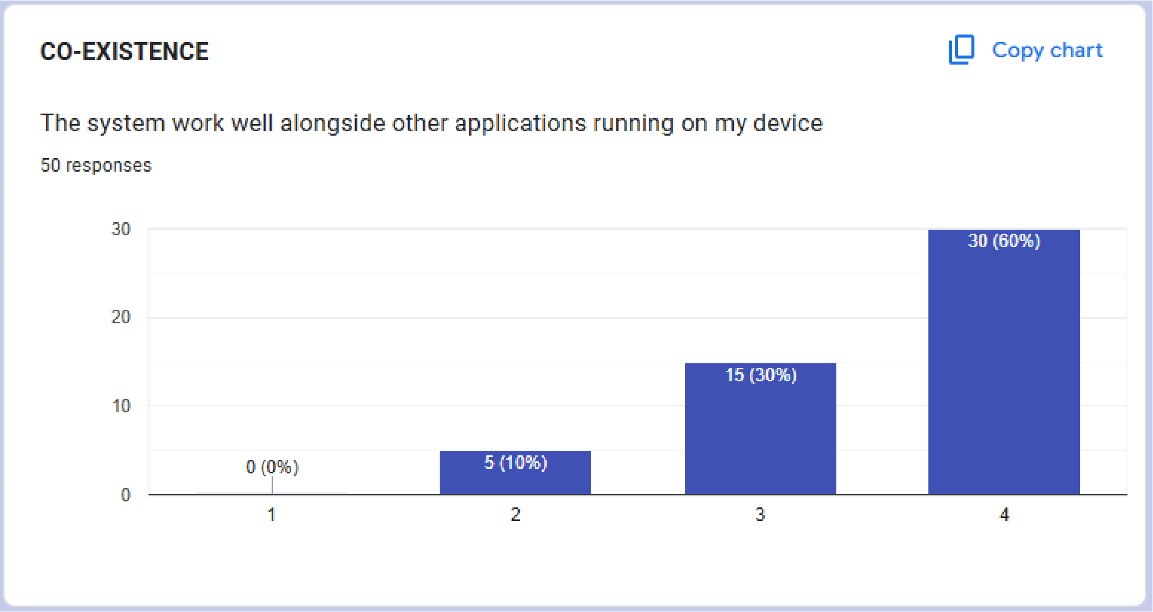
50 responses



CAPACITY

The system handle large or multiple video files without crashing or slowing down

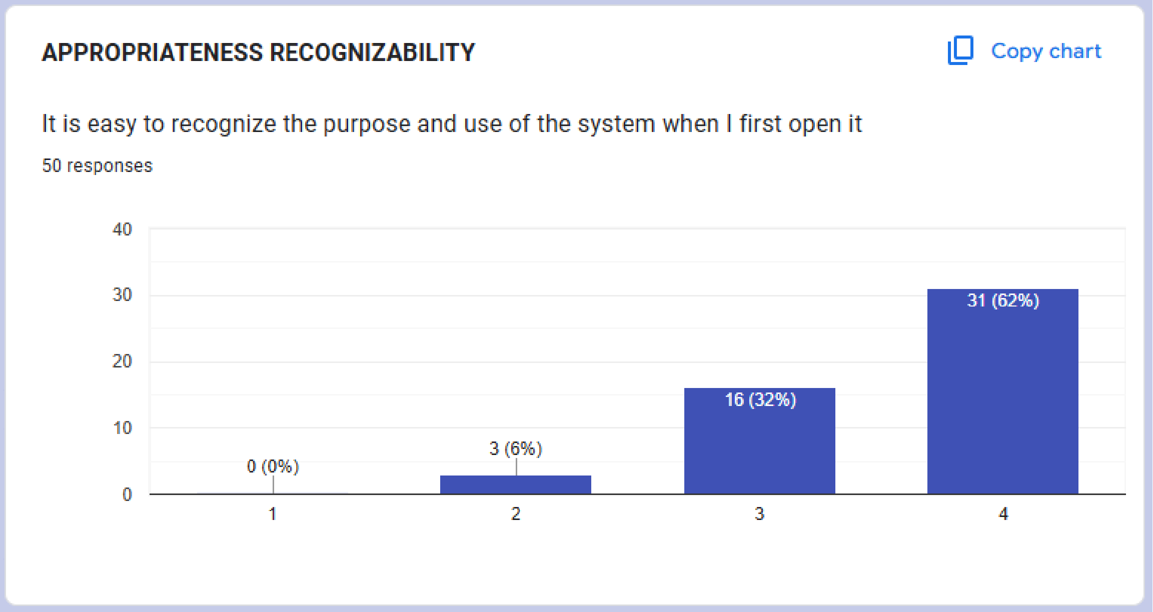
50 responses



CO-EXISTENCE

The system work well alongside other applications running on my device

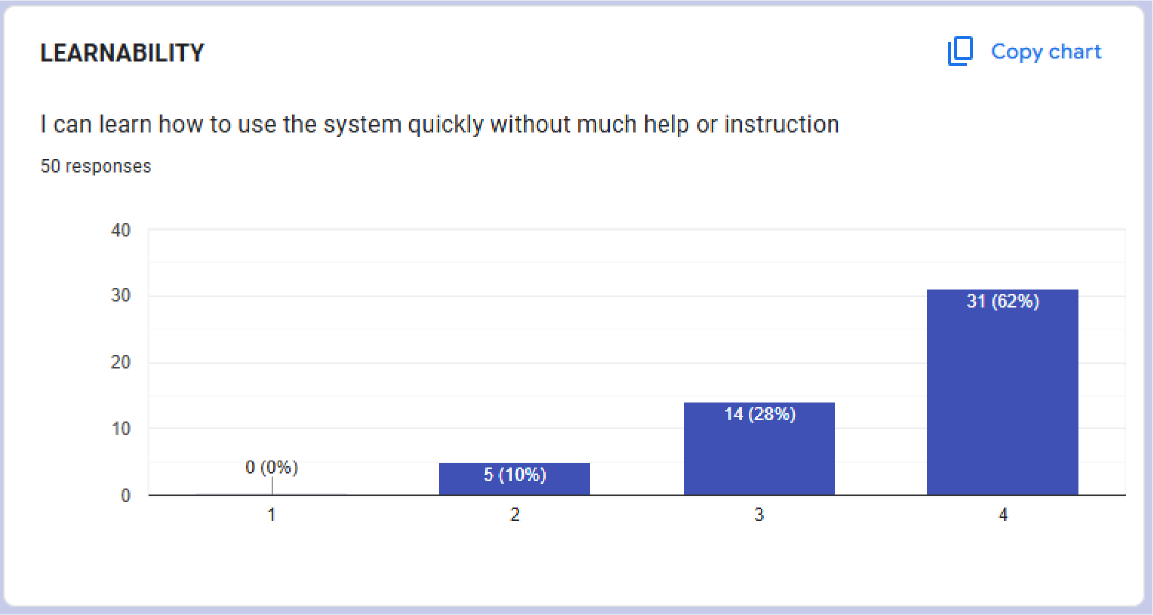
50 responses



APPROPRIATENESS RECOGNIZABILITY

It is easy to recognize the purpose and use of the system when I first open it

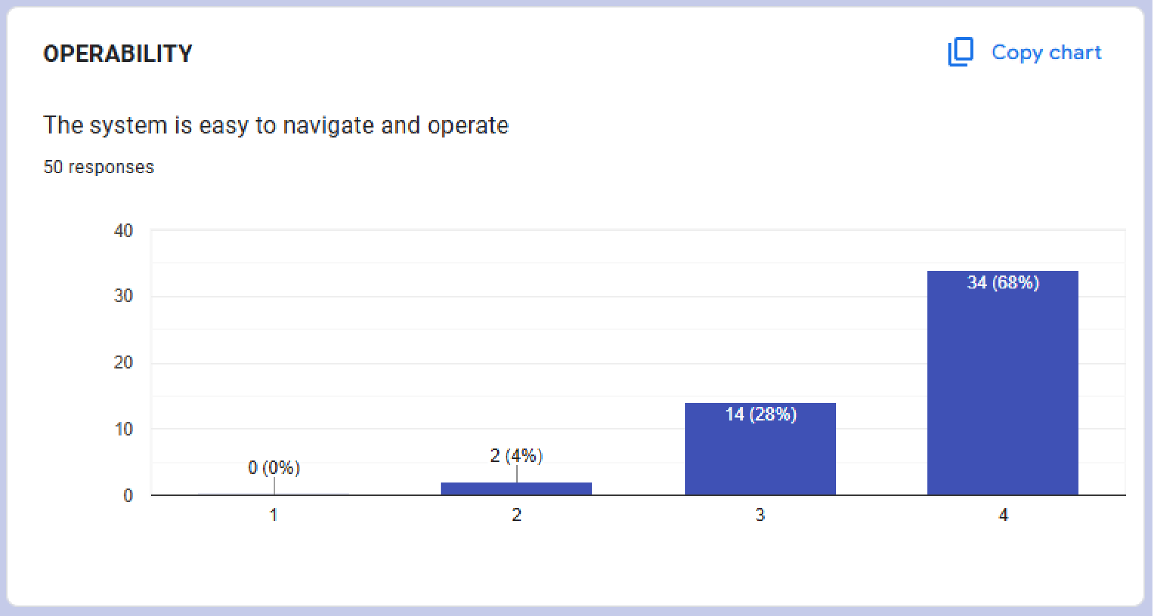
50 responses



LEARNABILITY

I can learn how to use the system quickly without much help or instruction

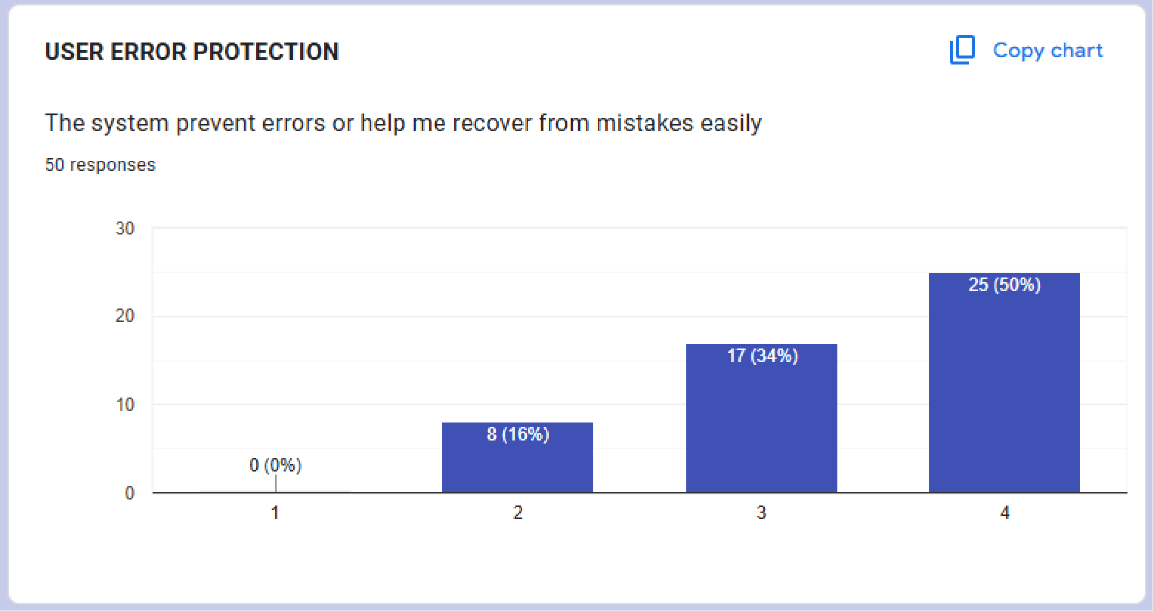
50 responses



OPERABILITY

The system is easy to navigate and operate

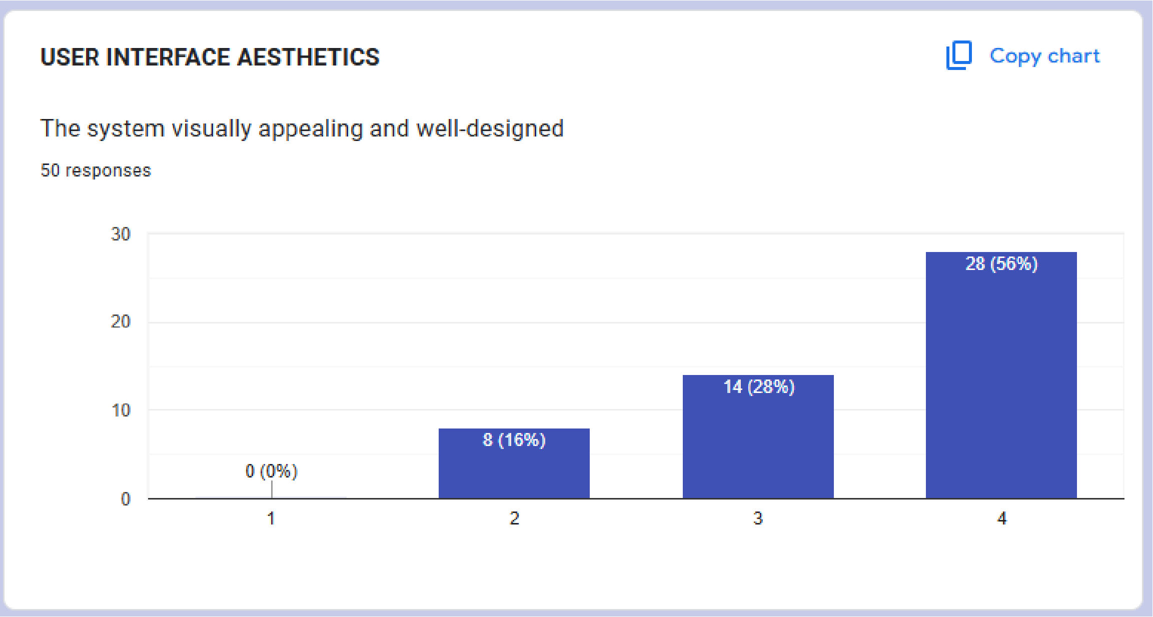
50 responses



USER ERROR PROTECTION

The system prevent errors or help me recover from mistakes easily

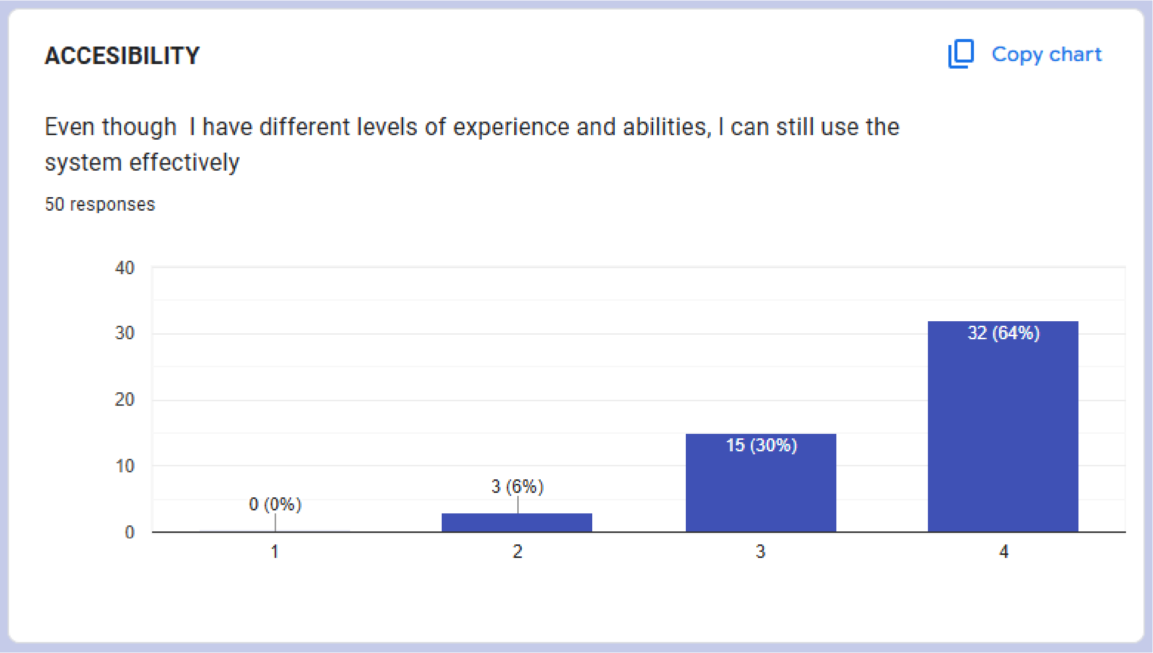
50 responses



USER INTERFACE AESTHETICS

The system visually appealing and well-designed

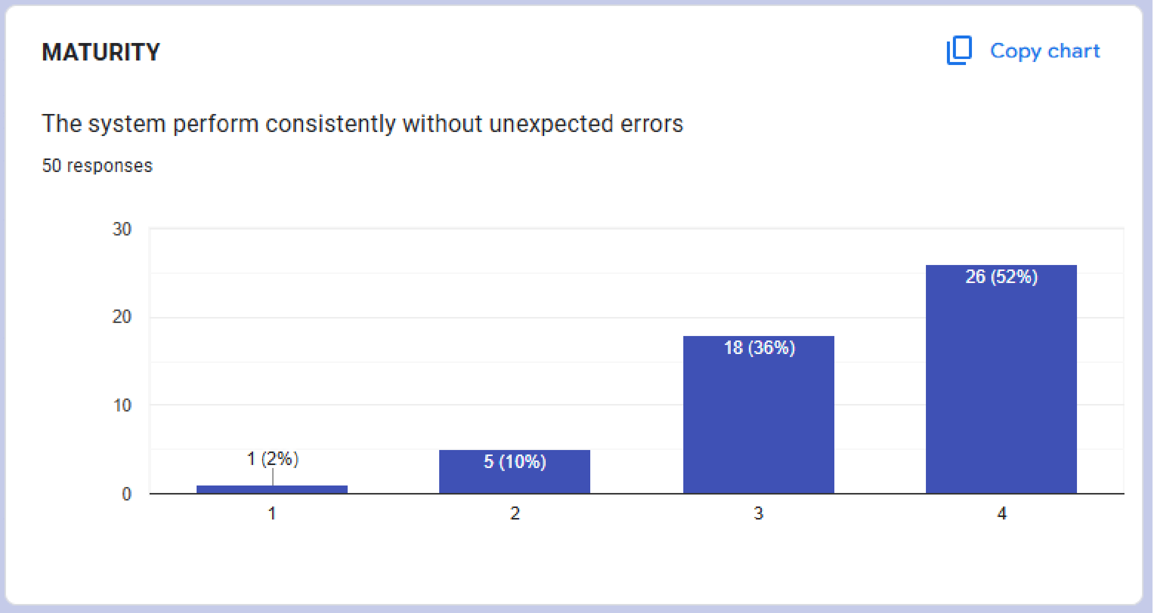
50 responses



ACCESIBILITY

Even though I have different levels of experience and abilities, I can still use the system effectively

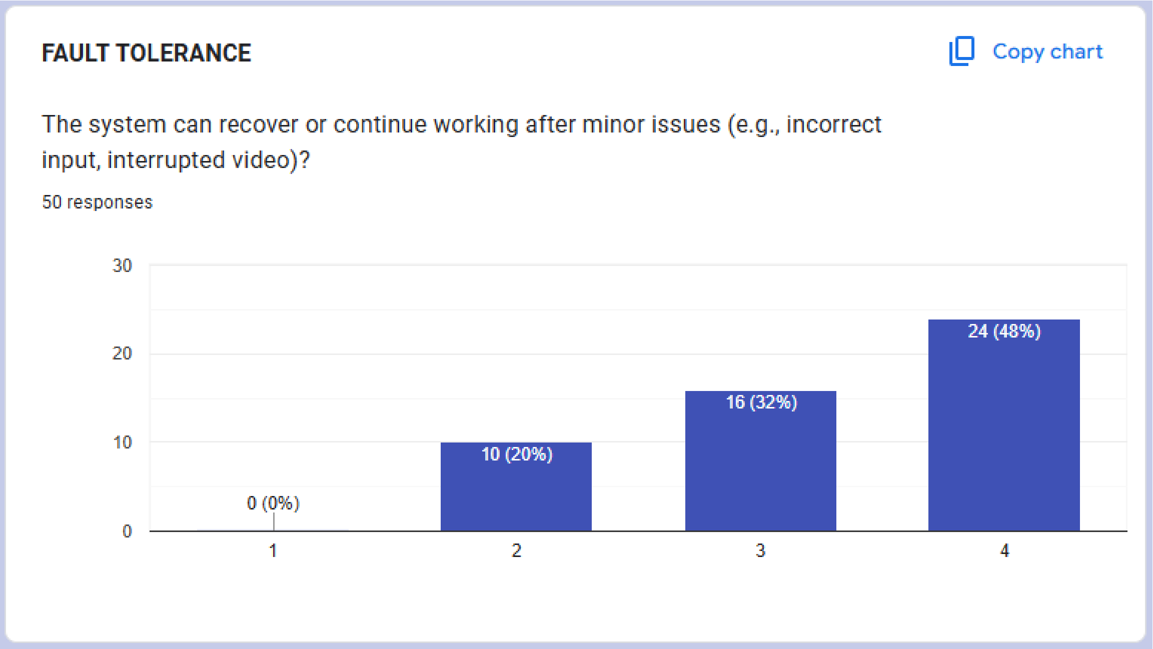
50 responses



MATURITY

The system perform consistently without unexpected errors

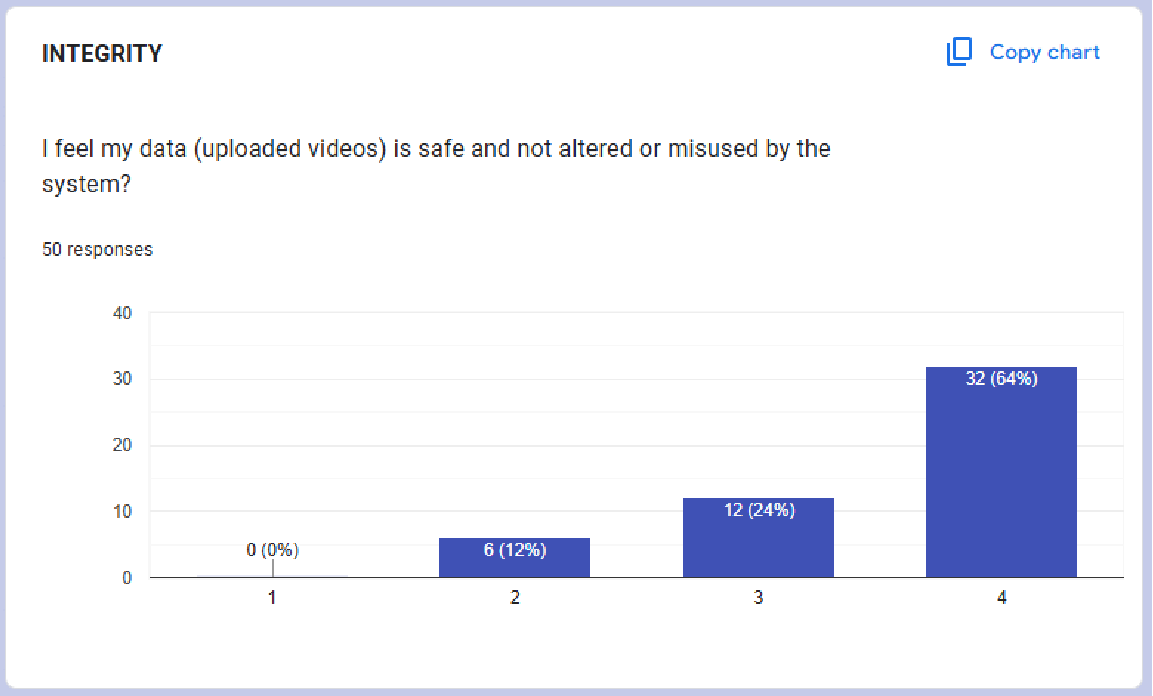
50 responses



FAULT TOLERANCE

The system can recover or continue working after minor issues (e.g., incorrect input, interrupted video)?

50 responses

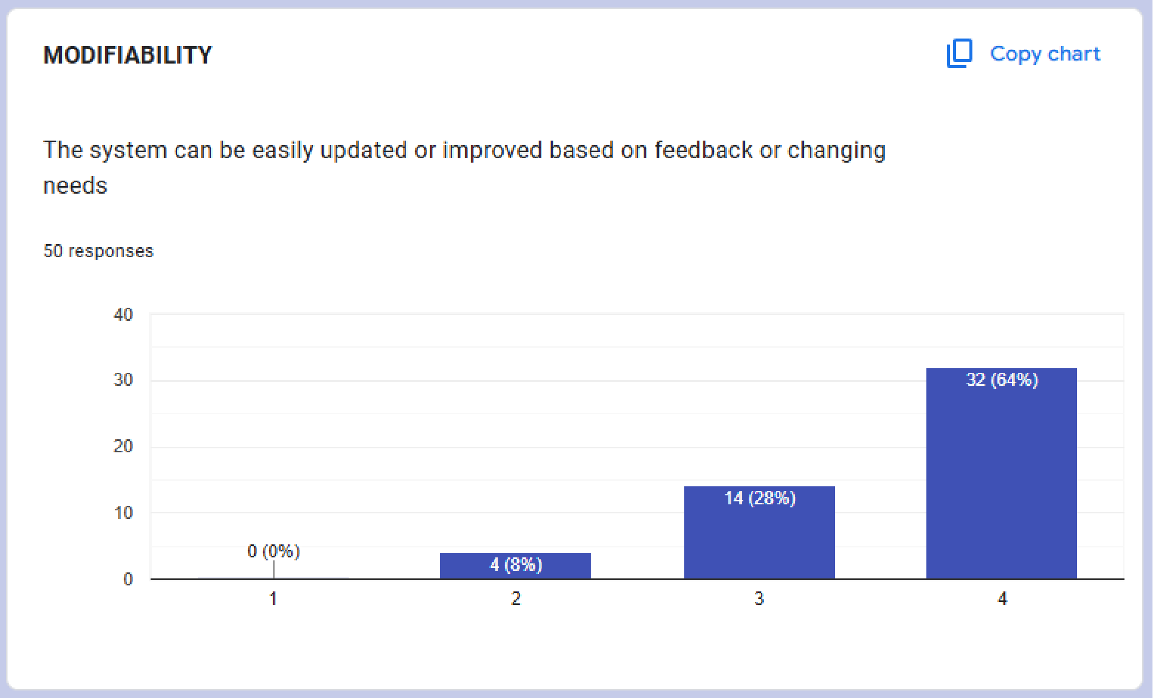


INTEGRITY

Copy chart

I feel my data (uploaded videos) is safe and not altered or misused by the system?

50 responses

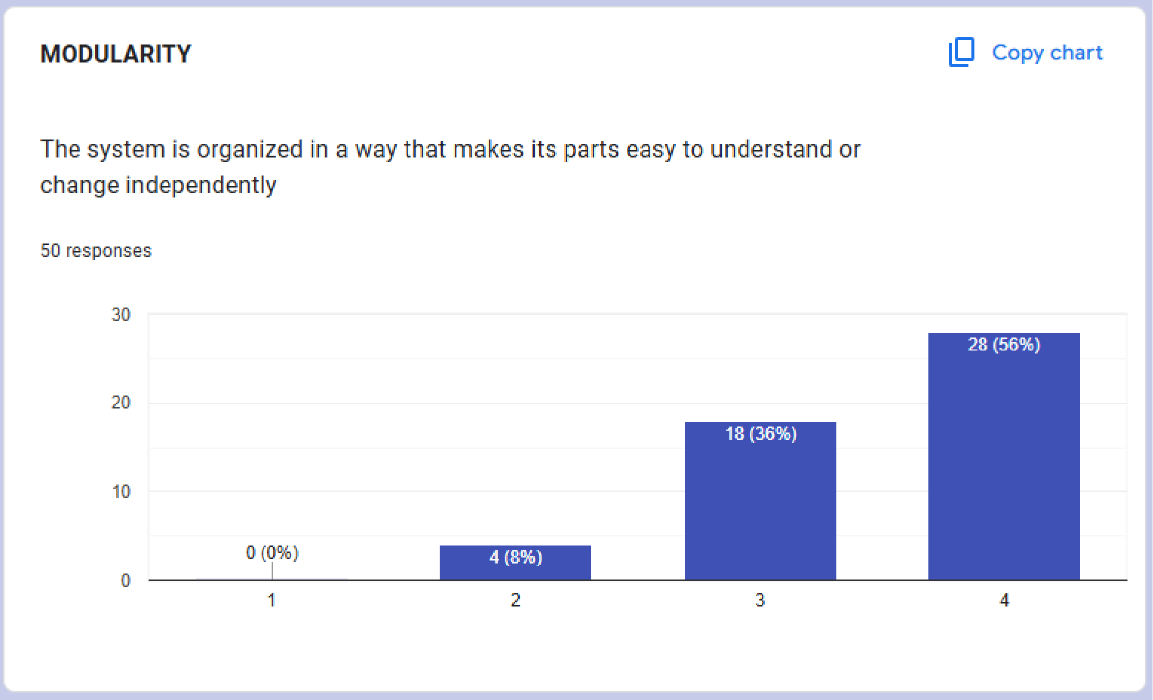


MODIFIABILITY

Copy chart

The system can be easily updated or improved based on feedback or changing needs

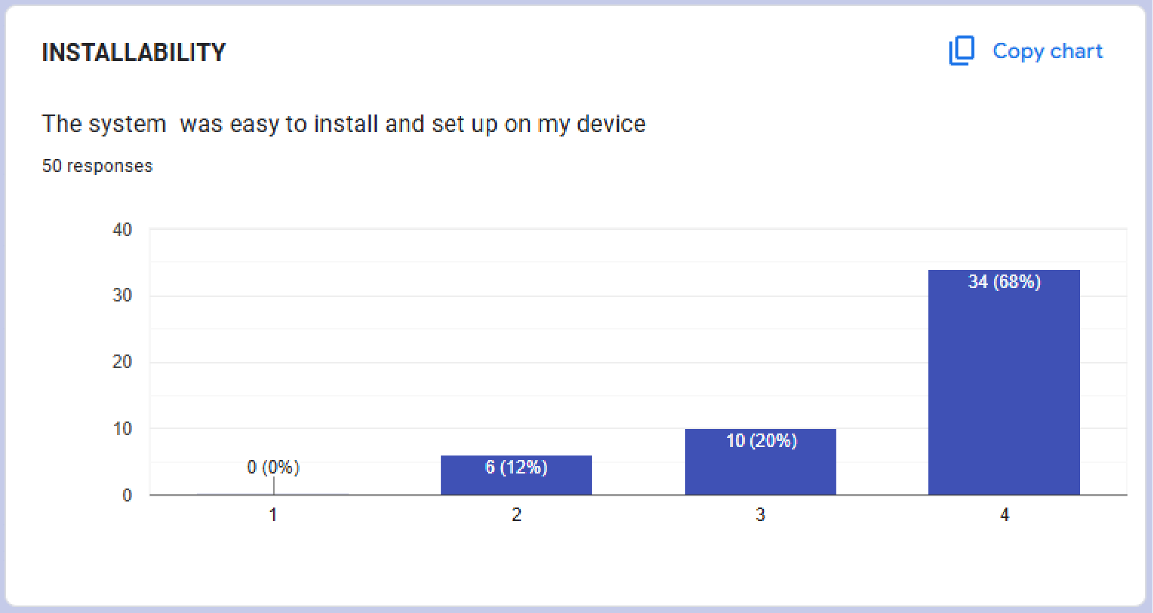
50 responses



MODULARITY

The system is organized in a way that makes its parts easy to understand or change independently

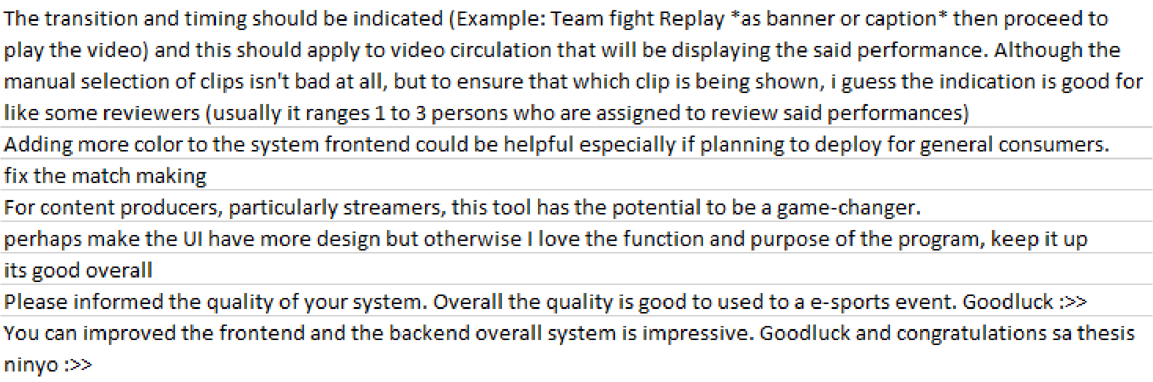
50 responses



INSTALLABILITY

The system was easy to install and set up on my device

50 responses



The transition and timing should be indicated (Example: Team fight Replay *as banner or caption* then proceed to play the video) and this should apply to video circulation that will be displaying the said performance. Although the manual selection of clips isn't bad at all, but to ensure that which clip is being shown, i guess the indication is good for like some reviewers (usually it ranges 1 to 3 persons who are assigned to review said performances)

Adding more color to the system frontend could be helpful especially if planning to deploy for general consumers.

fix the match making

For content producers, particularly streamers, this tool has the potential to be a game-changer.

perhaps make the UI have more design but otherwise I love the function and purpose of the program, keep it up

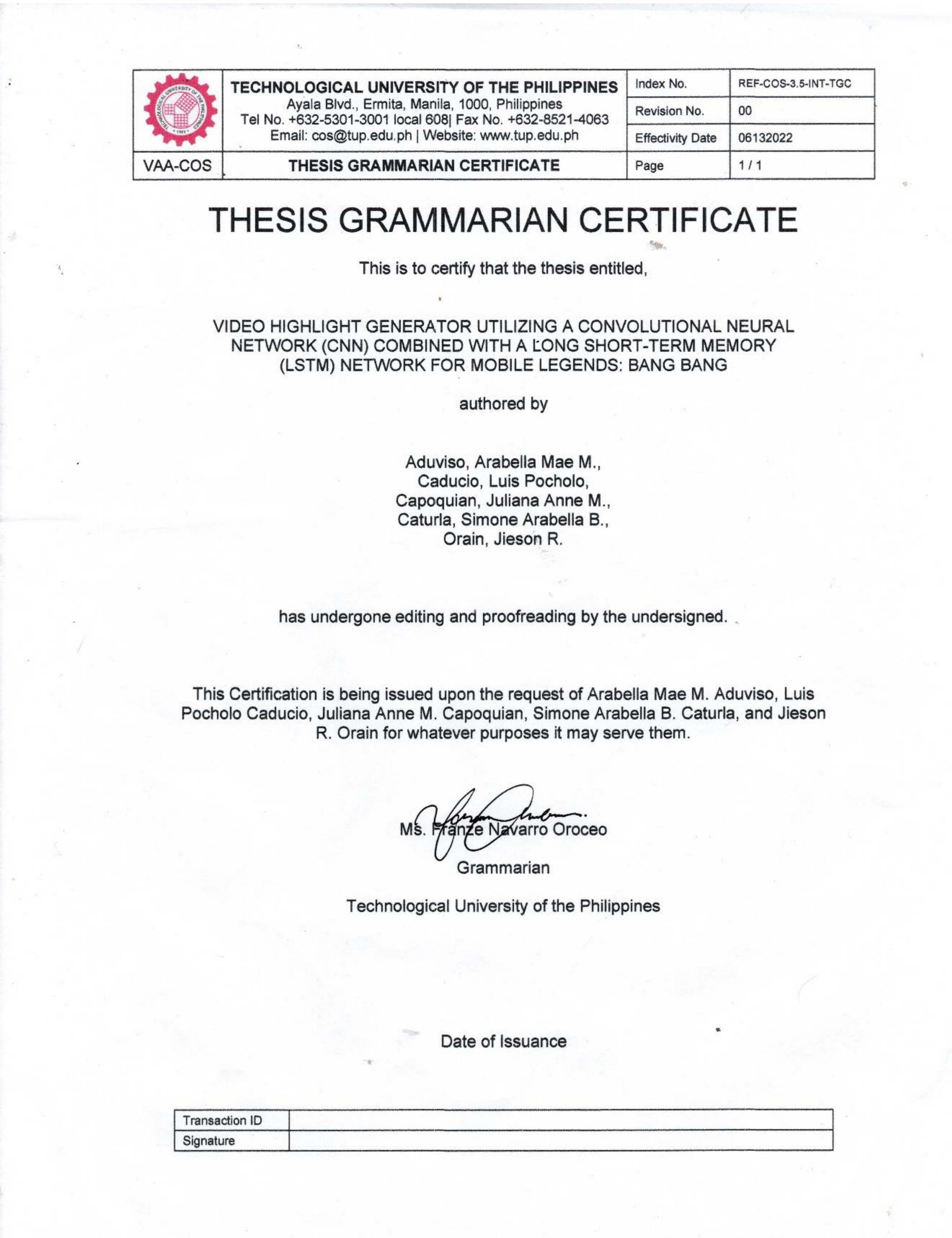
its good overall

Please informed the quality of your system. Overall the quality is good to used to a e-sports event. Goodluck :>>

You can improved the frontend and the backend overall system is impressive. Goodluck and congratulations sa thesis ninyo :>>

APPENDIX D

THESIS GRAMMARIAN CERTIFICATION



	TECHNOLOGICAL UNIVERSITY OF THE PHILIPPINES Ayala Blvd., Ermita, Manila, 1000, Philippines Tel No. +632-5301-3001 local 608| Fax No. +632-8521-4063 Email: cos@tup.edu.ph | Website: www.tup.edu.ph	Index No.	REF-COS-3.5-INT-TGC
		Revision No.	00
		Effectivity Date	06132022
VAA-COS	**THESIS GRAMMARIAN CERTIFICATE**	Page	1 / 1

THESIS GRAMMARIAN CERTIFICATE

This is to certify that the thesis entitled,

VIDEO HIGHLIGHT GENERATOR UTILIZING A CONVOLUTIONAL NEURAL NETWORK (CNN) COMBINED WITH A LONG SHORT-TERM MEMORY (LSTM) NETWORK FOR MOBILE LEGENDS: BANG BANG

authored by

Aduviso, Arabella Mae M.,
Caducio, Luis Pocholo,
Capoquian, Juliana Anne M.,
Caturla, Simone Arabella B.,
Orain, Jieson R.

has undergone editing and proofreading by the undersigned.

This Certification is being issued upon the request of Arabella Mae M. Aduviso, Luis Pocholo Caducio, Juliana Anne M. Capoquian, Simone Arabella B. Caturla, and Jieson R. Orain for whatever purposes it may serve them.

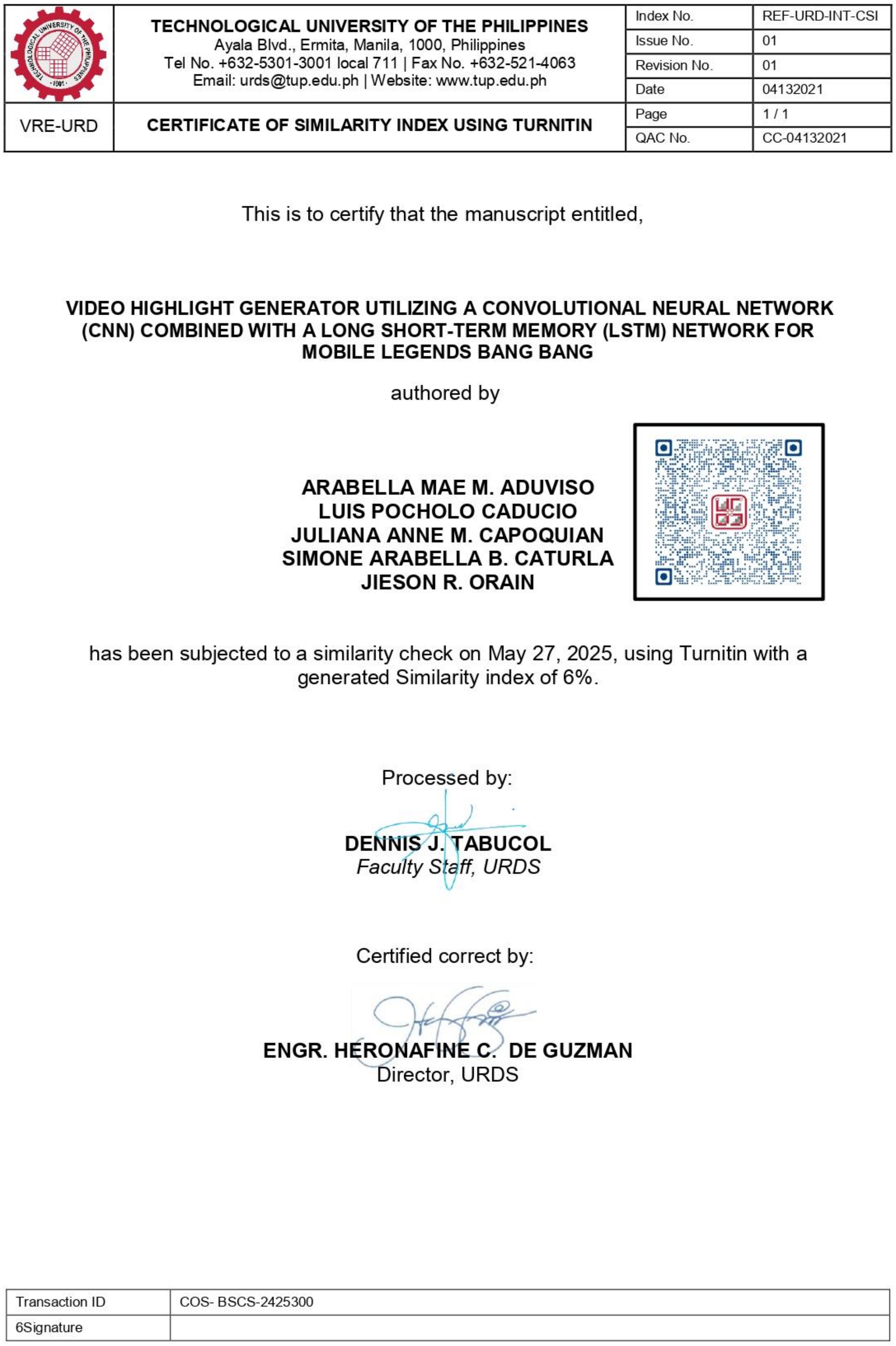
Ms. Franze Navarro Oroceo
Grammarian
Technological University of the Philippines

Date of Issuance

Transaction ID	
Signature	

APPENDIX E

CERTIFICATE OF SIMILARITY INDEX USING TURNITIN



	TECHNOLOGICAL UNIVERSITY OF THE PHILIPPINES Ayala Blvd., Ermita, Manila, 1000, Philippines Tel No. +632-5301-3001 local 711 | Fax No. +632-521-4063 Email: urds@tup.edu.ph | Website: www.tup.edu.ph	Index No.	REF-URD-INT-CSI
		Issue No.	01
		Revision No.	01
		Date	04132021
VRE-URD	**CERTIFICATE OF SIMILARITY INDEX USING TURNITIN**	Page	1 / 1
		QAC No.	CC-04132021

This is to certify that the manuscript entitled,

VIDEO HIGHLIGHT GENERATOR UTILIZING A CONVOLUTIONAL NEURAL NETWORK (CNN) COMBINED WITH A LONG SHORT-TERM MEMORY (LSTM) NETWORK FOR MOBILE LEGENDS BANG BANG

authored by

ARABELLA MAE M. ADUVISO
LUIS POCHOLO CADUCIO
JULIANA ANNE M. CAPOQUIAN
SIMONE ARABELLA B. CATURLA
JIESON R. ORAIN

has been subjected to a similarity check on May 27, 2025, using Turnitin with a generated Similarity index of 6%.

Processed by:

DENNIS J. TABUCOL
Faculty Staff, URDS

Certified correct by:

ENGR. HERONAFINE C. DE GUZMAN
Director, URDS

Transaction ID	COS- BSCS-2425300
6Signature	

References

Abdelhalim, B., & Titouna, F. (2023). Automatic Sports Video Classification Using CNN-LSTM Approach. 6th International Hybrid Conference on Informatics and Applied Mathematics. https://www.researchgate.net/publication/381740998_Automatic_Sports_Video_Classification_Using_CNN-LSTM_Approach

Academy, C. (2024, July 10). Mobile Legends Beginner Guide: Characters, Ranks, Mechanics | Coins.ph. Coins Academy. https://coins.ph/academy/mobile-legends-beginner-guide-characters-ranks-mechanics/

Accuracy, Precision and Recall. (n.d.). Padhai Time. Retrieved May 14, 2025, from https://padhaitime.com/Machine-Learning/Accuracy,-Precision-and-Recall

Ahmed, U. (2024). Explore VIDIZMO's Indexer in video content management system. Enterprise Video Streaming Solutions for Businesses, Enterprises, Government, Local, State Government, Healthcare, Education, Law Enforcement Agencies, Justice, Public Safety, Manufacturing, Financial & Banking Industry. https://blog.vidizmo.com/video-indexing-software

Ahmed, Y. (2024). MLBB Player Count (2024): How many people play the game? https://www.Esports.net/news/mobile-games/mlbb-player-count/

Alake, R. (2024, December 4). *Loss Functions in Machine Learning Explained*. DataCamp. Retrieved May 14, 2025, from https://www.datacamp.com/tutorial/loss-function-in-machine-learning

Alder, D. (2024, November 3). *Esports industry earnings — 2025 growth statistics | LEVVVEL*. LEVVVEL. https://levvvel.com/esports-statistics/

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*(1). https://doi.org/10.1186/s40537-021-00444-8

Anishnama. (2023, April 28). Understanding LSTM: architecture, pros and cons, and implementation. Medium. https://medium.com/@anishnama20/understanding-lstm-architecture-pros-and-cons-and-implementation-3e0cca194094

Announcer. (2022, February 18). Mobile Legends: Bang Bang Wiki. Retrieved May 9, 2025, from https://mobile-legends.fandom.com/wiki/Announcer#First_Blood

Antonio. (2021, December 28). *65 Terms in Mobile Legends and Their Meanings, Complete!* VCGamers News. Retrieved May 9, 2025, from https://www.vcgamers.com/news/en/terms-in-mobile-legends/

Beuzen, T. (n.d.). Chapter 5: Introduction to Convolutional Neural Networks. https://www.tomasbeuzen.com/deep-learning-with-pytorch/chapters/chapter5_cnns-pt1.html

Bharathi. (2025, April 11). *Confusion Matrix for Multi-Class Classification*. Analytics Vidhya. Retrieved May 14, 2025, from https://www.analyticsvidhya.com/blog/2021/06/confusion-matrix-for-multi-class-classification/

Biswal, A. (2024, July 16). Top 10 deep learning algorithms you should know in 2024. Simplilearn.com. https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm#:~:text=%EE%80%80Deep%20learning%EE%80%81%20uses%20artificial%20neural%20networks

Bretous, M. (2024, July 31). 6 Short-Form Video Trends Marketers Should Watch in 2024 [New Data]. 2024 HubSpot State of Marketing Report. https://blog.hubspot.com/marketing/short-form-video-trends

Britton, J. (n.d.). What is ISO 25010? Perforce Software. https://www.perforce.com/blog/qac/what-is-iso-25010

Canlas, R. B., Piad, K. C., & Lagman, A. C. (2021). An ISO/IEC 25010 Based Software Quality Assessment of a Faculty Research Productivity Monitoring and Prediction System. ICIT 2021: IoT and Smart City, 2, 238–242. https://doi.org/10.1145/3512576.3512619

Carneiro, T., Da Nobrega, R. V. M., Nepomuceno, T., Bian, G., De Albuquerque, V. H. C., & Filho, P. P. R. (2018). Performance analysis of Google Colaboratory as a tool for accelerating deep learning applications. IEEE Access, 6, 61677–61685. https://doi.org/10.1109/access.2018.2874767

Cartoonbase. (2023, October 10). Micro videos: snackable social media content – Cartoonbase. https://cartoonbase.com/en/resources/lexicon/micro-videos/

Ciaglia, F., Zuppichini, F. S., Guerrie, P., McQuade, M., & Solawetz, J. (2022). RoboFlow 100: a Rich, Multi-Domain Object Detection Benchmark. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2211.13523

Chung T, Sum S, Chan M, Lai E, Cheng N. Will Esports result in a higher prevalence of problematic gaming? A review of the global situation. J Behav Addict. 2019 Sep

1;8(3):384-394. doi: 10.1556/2006.8.2019.46. Epub 2019 Sep 25. PMID: 31553236; PMCID: PMC7044624.

Dao, L. M. (2024, September 17). The beginner’s guide to getting good at Mobile Legends: Bang Bang. ONE Esports. https://www.oneEsports.gg/mobile-legends/mobile-legends-beginners-guide/

Das, A., & Mishra, H. (2024). The Effect of Reels on Attention among Young and Middle-Aged Adults. https://ijip.in/wp-content/uploads/2024/07/18.01.031.20241203.pdf

Dawood, M. (2023, June 4). Convolutional neural networks (CNNs) in image and video processing. Medium. https://muhammaddawoodaslam.medium.com/convolutional-neural-networks-cnns-in-image-and-video-processing-552da8422604

Deepia. (2024, May 5). Convolutional Neural Networks | Deep Learning Animated [Video]. YouTube. https://www.youtube.com/watch?v=8NbXFoVk1mo

Ding, J., Li, Y., Li, Y., & Jin, D. (2018). Click versus Share: A Feature-driven Study of Micro-Video Popularity and Virality in Social Media. In Society for Industrial and Applied Mathematics eBooks (pp. 198–206). https://doi.org/10.1137/1.9781611975321.23

Dofitas, C., Gil, J., & Byun, Y. (2024). Multi-Directional Long-Term Recurrent Convolutional network for road situation recognition. *Sensors*, *24*(14), 4618. https://doi.org/10.3390/s24144618

DPG. (2022, April 21). MLBB terms you need to know: First Blood, Savage, Godlike, and more. *Codashop Blog*. Retrieved May 9, 2025, from https://news.codashop.com/ph/mlbb-terms-you-need-to-know-first-blood-savage-godlike-and-more/

f1 Score Definition. (n.d.). Encord. Retrieved May 14, 2025, from https://encord.com/glossary/f1-score-definition/

Feddewar, Palash & Deshmukh, Bharti & Ijmtst, Editor. (2022). Convolution Neural Network (CNN) for Video Processing: A Survey. International Journal for Modern Trends in Science and Technology. 8. 147-152. 10.46501/IJMTST0801025.

Fernandes, B., Silva, F., Alaiz-Moreton, H., Novais, P., Neves, J., & Analide, C. (2020). Long Short-Term Memory Networks for traffic flow forecasting: exploring input variables, time frames and Multi-Step approaches. Informatica, 31(4), 723–749. https://doi.org/10.15388/20-infor431

Francesco. (2024, April 9). 📷 RoboFlow 100: a Multi-Domain Object Detection Benchmark. Roboflow Blog. https://blog.roboflow.com/roboflow-100/

Freenergi, A. (2021, December 10). Convolutional neural network for object recognition and detection. Medium. https://medium.com/@ringlayer/convolutional-neural-network-for-object-recognition-and-detection-126a22af8975

Game ended - Slang meaning and examples - FastSlang. (n.d.). https://www.fastslang.com/game-ended

GeeksforGeeks. (2024a, May 18). Fully Connected Layer vs Convolutional Layer. GeeksforGeeks. https://www.geeksforgeeks.org/fully-connected-layer-vs-convolutional-layer/

GeeksforGeeks. (2024b, August 26). Difference between machine learning and deep learning. GeeksforGeeks. https://www.geeksforgeeks.org/difference-between-machine-learning-and-deep-learning/

GeeksforGeeks. (2024c, October 10). Introduction to Convolution Neural Network. GeeksforGeeks. https://www.geeksforgeeks.org/introduction-convolution-neural-network/

GeeksforGeeks. (2024d, November 19). Softmax activation function in neural networks. GeeksforGeeks. https://www.geeksforgeeks.org/the-role-of-softmax-in-neural-networks-detailed-explanation-and-applications/

GeeksforGeeks. (2024e, November 19). *Softmax activation function in neural networks*. GeeksforGeeks. https://www.geeksforgeeks.org/the-role-of-softmax-in-neural-networks-detailed-explanation-and-applications/

GeeksforGeeks. (2025a, April 26). What is NonMaximum Suppression? GeeksforGeeks. https://www.geeksforgeeks.org/what-is-non-maximum-suppression/

GeeksforGeeks. (2025b, April 9). What is an API (Application Programming Interface). GeeksforGeeks. https://www.geeksforgeeks.org/what-is-an-api/

GeeksforGeeks. (2025c, April 5). *Activation functions in Neural Networks*. GeeksforGeeks. https://www.geeksforgeeks.org/activation-functions-neural-networks/

GeeksforGeeks. (2025d, February 27). *Understanding the confusion matrix in machine learning*. Retrieved May 14, 2025, from https://www.geeksforgeeks.org/confusion-matrix-machine-learning/

GeeksforGeeks. (2025e, May 23). Introduction to recurrent neural networks. GeeksforGeeks. https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/

GeeksforGeeks. (2025f, April 5). What is LSTM Long Short Term Memory? GeeksforGeeks. https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/

GeeksforGeeks. (2025g, March 4). *What is Adam Optimizer?* GeeksforGeeks. https://www.geeksforgeeks.org/adam-optimizer/

Google Colab. (n.d.). *Colaboratory. Google*. Retrieved November 12, 2024, from https://research.google.com/colaboratory/faq.html

Hindarto, D. (2023). Exploring YOLOV8 Pretrain for Real-Time detection of Indonesian Native fish species. SinkrOn, 8(4), 2776–2785. https://doi.org/10.33395/sinkron.v8i4.13100

Huang, H., Wang, B., Xiao, J., & Zhu, T. (2024). Improved small-object detection using YOLOv8: A comparative study. Applied and Computational Engineering, 41(1), 80–88. https://doi.org/10.54254/2755-2721/41/20230714

Ibrahim, M. S., Muralidharan, S., Deng, Z., Vahdat, A., & Mori, G. (2016). A Hierarchical Deep Temporal Model for Group Activity Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2016.217

iMerit. (2022, November 2). Using neural networks for video classification. https://imerit.net/blog/using-neural-networks-for-video-classification-blog-all-pbm/

Intelligent Prediction of Separated Flow Dynamics using Machine Learning. (2024). *Journal of Applied Fluid Mechanics,* 18(2). https://doi.org/10.47176/jafm.18.2.2910

Introduction to Cloud TPU. (n.d.). Google Cloud. https://cloud.google.com/tpu/docs/intro-to-tpu

Kang, S., & Lee, J. (2020). An E-sports rp using win-loss probability model. SAC ’20: Proceedings of the 35th Annual ACM Symposium on Applied Computing. https://doi.org/10.1145/3341105.3373894

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). Large-Scale Video Classification with Convolutional Neural Networks. 2014 IEEE Conference on Computer Vision and Pattern Recognition. https://doi.org/10.1109/cvpr.2014.223

Keita, Z. (2023). Classification in Machine Learning: An Introduction. https://www.datacamp.com/blog/classification-machine-learning

Keras: The high-level API for TensorFlow. (n.d.). TensorFlow. https://www.tensorflow.org/guide/keras

Khalil, S., Shah, H. A., & Bednarik, R. (2025). Improving microsurgical suture training with automated phase recognition and skill assessment via deep learning. *Computers in Biology and Medicine*, *192*(0010–4825), 110238. https://doi.org/10.1016/j.compbiomed.2025.110238

Khan, A. A., & Shao, J. (2022). SPNet: A deep network for broadcast sports video highlight generation. Computers & Electrical Engineering, 99, 107779. https://doi.org/10.1016/j.compeleceng.2022.107779

Khan, M., Mehran, M. T., Haq, Z. U., Ullah, Z., Naqvi, S. R., Ihsan, M., & Abbass, H. (2021). Applications of artificial intelligence in COVID-19 pandemic: A comprehensive review. Expert Systems With Applications, 185, 115695. https://doi.org/10.1016/j.eswa.2021.115695

Kobis, D. C., & Tomatala, M. F. (2020). STUDENTS' PERCEPTIONS ON MOBILE LEGENDS: BANG-BANG (MLBB) AS MEDIUM TO LEARN ENGLISH. LINGUA JURNAL ILMIAH, 16(2), 22–38. https://doi.org/10.35962/lingua.v16i2.52

Kromydas, B. (2023, January 18). Convolutional Neural Network: A complete guide. LearnOpenCV – Learn OpenCV, PyTorch, Keras, Tensorflow With Code, & Tutorials. https://learnopencv.com/understanding-convolutional-neural-networks-cnn/

Kundu, R. (2024, July 2). Video classification: methods, use cases, tutorial. V7. https://www.v7labs.com/blog/video-classification-guide

Lohokare, A., Shah, A., & Zyda, M. (2020). Deep Learning Bot for League of Legends. *Proceedings of the Sixteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-20)*, 322–327. https://doi.org/10.1609/aiide.v16i1.7449

Mawalia, K. A. (2020). The impact of the mobile legend game in creating virtual reality. http://dx.doi.org/10.20473/ijss.v12i2.22908. https://www.researchgate.net/publication/346703962_The_impact_of_the_Mobile_Legend_game_in_creating_virtual_reality

McKeever, G. (2024, January 8). What is Fault Tolerance? | Creating a Fault Tolerant System | Imperva. Learning Center. https://www.imperva.com/learn/availability/fault-tolerance/

moviepy. (2025, January 10). PyPI. https://pypi.org/project/moviepy/

Nair, G. (2023, August 23). Activation function in neural networks: SigmoiD, TANH, RELU, Leaky RELU, Parametric RELU, ELU, SoftMax, GELU | Medium. Medium.

https://medium.com/@gauravnair/the-spark-your-neural-network-needs-understanding-the-significance-of-activation-functions-6b82d5f27fbf#c8a2

Narayan, G. (2023, August 10). Explained: How Video indexing optimizes video search and streaming experience. ImageKit.io Blog. https://imagekit.io/blog/video-indexing/

Oladipo, T. (2023). A Short-Form video explainer (and five tips for using it). Buffer: All-you-need Social Media Toolkit for Small Businesses. https://buffer.com/resources/short-form-video/

Pardeshi, S. (2023, March 25). CNN-LSTM Architecture and Image Captioning - Analytics Vidhya - Medium. *Medium.* https://medium.com/analytics-vidhya/cnn-lstm-architecture-and-image-captioning-2351fc18e8d7

Pokharel, P. (2021, June). How and Why We're Making Video Editing as Easy as Editing Text. Reduct. https://reduct.video/blog/video-editing-as-easy-as-editing-text

Python releases for Windows. (n.d.). Python.org. https://www.python.org/downloads/windows/

Pyun, H., Jang, W. (Eric), Lee, G., Ryu, Y., Hwang, H., & Jeong, J. (2023). Determinants of Esports Highlight Viewership: The Case of League of Legends Champions Korea. Proceedings of the 56th Hawaii International Conference on System Sciences. https://www.cs.ucdavis.edu/~yjlee/projects/eccv2018-stmn.pdf

Qmii. (2023, October 9). Understanding ISO 25010: Navigating the seas of software quality. https://www.qmii.com/understanding-iso-25010-navigating-the-seas-of-software-quality/

Ringgo. (2024, September 11). *How to Destroy Turret Mobile Legends (ML).* Esportsku. Retrieved May 9, 2025, from https://en.Esportsku.com/how-to-destroy-turret-mobile-legends-ml/

Robb, J., Garner, T., Collins, K., & Nacke, L. E. (2017). The Impact of Health-Related User Interface Sounds on Player Experience. Simulation & Gaming, 48(3), 402–427. doi:10.1177/1046878116688236

Robbins, R. (2023, July 26). ISO 25010. TheTechnologyVault.com. https://thetechnologyvault.com/iso-25010-software-quality-standard

Roboflow. (2025). *Roboflow Universe: Computer Vision Datasets*. https://universe.roboflow.com/

Rossel, J. D. (2021, March 30). Mobile Legends Player count: How many people still play ML. AFK Gaming. https://afkgaming.com/mobileEsports/news/7319-mobile-legends-player-count-how-many-people-still-play-ml

Sanagapati, P. (2020, June 8). A simple CNN model beginner guide !!!!!! Kaggle. https://www.kaggle.com/code/pavansanagapati/a-simple-cnn-model-beginner-guide

Sevilla, J. (2025, January 27). 3 pro tips on how to secure the Turtle in Mobile Legends: Bang Bang. ONE Esports. https://www.oneEsports.gg/mobile-legends/3-pro-tips-secure-turtle-mlbb/

Singh, K. (2022, October 17). The complete Guide for Beginners MLBB (MOBILE LEGENDS BANG BANG). LinkedIn. Retrieved October 3, 2024, from https://www.linkedin.com/pulse/complete-guide-beginners-mlbb-mobile-legends-bang-karishma-singh/

Sharma, V., Gupta, M., Kumar, A., & Mishra, D. (2021). Video Processing using Deep Learning Techniques: A Systematic Literature review. IEEE Access, 9, 139489–139507. https://doi.org/10.1109/access.2021.3118541

Shepherd, J. (2024, April 23). 30 vital video marketing statistics you need to know in 2024. The Social Shepherd. https://thesocialshepherd.com/blog/video-marketing-statistics

Saxena, S. (2025, May 1). What is LSTM? Introduction to Long Short-Term Memory. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/

Sridevi, M., & Kharde, M. (2020). Video summarization using highlight detection and pairwise deep ranking model. Procedia Computer Science, 167, 1839–1848. https://doi.org/10.1016/j.procs.2020.03.203

Teamfight - Slang Meaning and examples - FastSlang. (n.d.). https://www.fastslang.com/teamfight

TensorFlow. (n.d.). TensorFlow. https://www.tensorflow.org/

Trilles, C. (2025, April 18). Step-by-step guide on how to secure Lord in Mobile Legends and win almost every time. ONE Esports. https://www.oneEsports.gg/mobile-legends/how-to-secure-lord-mobile-legends/

Tunguia, J. & Staff Team. (2024). Mobile Legends (MLBB) Farming Guide: Here’s how to dominate every match. GamingonPhone. https://gamingonphone.com/guides/mobile-legends-farming-guide/

Turtle. (2025, April 10). Mobile Legends: Bang Bang Wiki. Retrieved May 9, 2025, from https://mobile-legends.fandom.com/wiki/Turtle

Uddin, Md. A., Talukder, Md. A., Uzzaman, M. S., Debnath, C., Chanda, M., Paul, Islam, Md. M., Khraisat, A., Alazab, A., & Aryal, S. (2024). Deep learning-based human activity recognition using CNN, ConvLSTM, and LRCN. International Journal of Cognitive Computing in Engineering, 5(2666–3074), 259–268. https://doi.org/10.1016/j.ijcce.2024.06.004

Ultralytics. (2024, November 7). YOLOV8. Ultralytics YOLO Docs. https://docs.ultralytics.com/models/yolov8/

Ultralytics. (2025). *F1-Score - Discover the importance of the F1-score in machine learning! Learn how it balances precision and recall for optimal model evaluation.* https://www.ultralytics.com/glossary/f1-score#:~:text=The%20F1-Score%20is%20a%20widely%20used%20metric%20in,balances%20two%20other%20important%20metrics%3A%20precision%20and%20recall

Video Highlight Reels: 5 Best practices for sharing insight. (n.d.). https://www.usertesting.com/blog/creating-effective-video-highlight-reels

W3Schools.com. (n.d.). https://www.w3schools.com/python/numpy/numpy_intro.asp

Waqas, M., & Humphries, U. W. (2024). A critical review of RNN and LSTM variants in hydrological time series predictions. *MethodsX*, 13(2215–0161), 102946. https://doi.org/10.1016/j.mex.2024.102946

What is a GPU? Graphics processing units defined. (n.d.). Intel. https://www.intel.com/content/www/us/en/products/docs/processors/what-is-a-gpu.html

YOLOv8 Object Detection Model: What is, How to Use. (2025). https://roboflow.com/model/yolov8

RESEARCHERS' PROFILE

Arabella Mae M. Aduviso

arabellamae.aduviso@tup.edu.ph • 0956-046-2371 • linkedin.com/in/arabella-mae-aduviso/

EDUCATION

Technological University of the Philippines **August 2025**
B.S. in Computer Science (Non-STEM) *Ermita, Manila*

TECHNICAL SKILLS

Languages: C/C++, Java, TypeScript, JavaScript, Python, R, SQL
Technologies: Flask, Node.js, Express.js, React.js, REST APIs, Google Cloud Platform (GCP), Tailwind CSS, Networking (TCP/IP, client-server models)
Tools: VS Code, Netbeans. Jupyter Notebook, Google Colab, Google Cloud, MySQL, Vite, Qt Designer, Git

PROJECTS

Information Management System | MERN, Tailwind CSS **May 2025**

- Developed a client management system for a coworking space to handle virtual office subscriptions, including subscription history tracking and document submissions per client.
- Integrated an automated email reminder system for upcoming subscription expiries, built with modern web technologies and a responsive, user-friendly interface.

Tomographic Image Reconstruction | Python **Jun. 2024**

- Applied the Maximum Likelihood - Expectation Maximization (ML-EM) algorithm to automate tomographic image reconstruction, a technique commonly used in medical CT scans. The algorithm iteratively minimized the difference between measured and estimated data, resulting in clearer and more accurate images. This demonstrated the effectiveness of statistical reconstruction methods in enhancing image processing tasks.

Simulation for COVID-19 Progression | R, ggplot2 **Jun. 2024**

- Utilized a real-world COVID-19 dataset (March 2020) with demographic and health status information to simulate virus progression using Agent-Based Modeling (ABM), visualizing trends in infection severity, regional spread, and sex-based outcomes, identifying higher severity in male patients and age-specific infection differences.
- Produced visual analyses to support public health responses and intervention strategies, while addressing model limitations like behavioral assumptions and data biases, highlighting the need for ongoing refinement.

Market Basket Analysis on the Sales Invoice | Python **Jan. 2024**

- Conducted Market Basket Analysis using Apriori algorithm on sales invoice data from CMC Rice Center and General Merchandise to identify frequently purchased item combinations.
- Utilized association rule mining results to recommend cross-selling strategies and product bundling for optimized sales and enhanced customer experience.

Luis Pocholo Caducio

luispocholo.caducio@tup.edu.ph • (956) 158-5855 • linkedin.com/in/luis-pocholo-caducio-781697273

EDUCATION

Technological University of the Philippines Manila **August 2025**
B.S. in Computer Science (Non-Stem) *San Marcelino St., Ermita, Manila*

Colegio de San Juan de Letran Manila **July 2021**
Information Communications Technology (TVL-ICT) *Muralla St., Intramuros, Manila*

TECHNICAL SKILLS

Languages: C/C++, JavaScript, TypeScript, Python, SQL, HTML, CSS
Technologies: Node.js, Express.js, React.ts, REST APIs, Tailwind CSS, Vite, Git, React.js
Tools: VS Code, Jupyter Notebook, MySQL, Vite, Qt Designer, Git

PROJECTS

Information Management System | MERN, Tailwind CSS **May 2025**

- Developed an information management system for a company to track virtual office subscriptions, including subscription history and document submissions per company's client.
- Implemented an auto-send email system in EmailJS for expiring subscriptions, built with modern web technologies and a responsive, user-friendly interface.

Market Basket Analysis on the Sales Invoice | Python **Jan. 2024**

- Conducted Association Analysis using Apriori Algorithm on sales invoice data from a general merchandise to identify frequently purchased item combinations.
- Used association rule mining results to recommend cross-selling strategies and product bundling for optimized sales and enhanced customer experience.

Restaurant Billing System | Python **Jul. 2023**

- Developed a restaurant billing system that is a point of sales system which displays the restaurant's menu, takes orders from its customers, and tracks its daily sales.
- Utilized Qt Designer's drag and drop feature to design the system's GUI.

Juliana Anne M. Capoquian

julianaanne.capoquian@tup.edu.ph • 0926-757-3141 • linkedin.com/in/juliana-capoquian-1b88422b7

EDUCATION

Technological University of the Philippines Manila **August 2025**
B.S. in Computer Science (Non- Stem) *San Marcelino Ermita, Manila*

TECHNICAL SKILLS

Languages: C, **C++,** Java, Python, Dart, Visual Basic, HTML, CSS
Technologies: Flutter, MobX, Git, PyQt, WordPress
Tools: Android Studio, Netbeans, VS Code, Postman, Google Cloud Platform (GCP), Jupyter Notebook, RoboFlow

PROJECTS

Eeazy App User Login | Dart, Flutter **Apr. 2025**

- Created the front-end code for the Eeazy mobile application's user login using Flutter. Consisted of user input validation and logic.

Minfolio Website | WordPress, CSS **Mar. 2025**

- Utilized WordPress, Helium Theme, and other plugins to create a website which contains information and blog posts.
- Responsive to mobile and tablet devices.

BasketXpertPOS | Python, PyQt5 **Jun. 2024**

- Developed a Point-of-Sale system for an existing salon business in Pasig City. The system simplified operations through personalized UI, improved client experiences with the system's ability to accept cashless transactions and gave essential insights into consumer behavior by the automation of Daily Sales Report.
- Integration of Machine Learning to discover customer behavioral patterns and provide personalized offers through Market Basket Analysis.

Tomographic Image Reconstruction | Python **Jun. 2024**

- Applied the Maximum Likelihood - Expectation Maximization (ML-EM) algorithm to automate tomographic image reconstruction, a technique commonly used in medical CT scans. The algorithm iteratively minimized the difference between measured and estimated data, resulting in clearer and more accurate images. This demonstrated the effectiveness of statistical reconstruction methods in enhancing image processing tasks.

Simone Arabella Caturla

simone.caturla@tup.edu.ph • 0961-718-2948 • linkedin.com/in/simone-caturla

EDUCATION

Technological University of the Philippines Manila **August 2025**
B.S. in Computer Science (Non-Stem) *San Marcelino St., Ermita, Manila*

Colegio de San Juan de Letran Manila **July 2021**
Information Communications Technology (TVL-ICT) *Muralla St., Intramuros, Manila*

TECHNICAL SKILLS

Languages: JavaScript, Python, HTML, CSS
Tools: VS Code, Tkinter, Vite, Git, React.js
Skills: UI Design, Web Development, Quality Assurance Testing, Technical Writing, Front-End Development

EXPERIENCE

UI Designer and Quality Assurance Tester **Dec. 2024**
Senate of the Philippines | Pasay City

- Designed the UI for both the user and admin side of a website.
- Designed the UI for a mobile application
- Created user manuals for multiple applications
- Tested and evaluated multiple applications to identify bugs and improve user experience.

PROJECTS

A Market Basket Analysis on the Sales Invoices of a General Merchandise Store | Python **Jan. 2024**

- Conducted Association Analysis using Apriori Algorithm on sales invoice data from a general merchandise to identify frequently purchased item combinations.
- Used association rule mining results to recommend cross-selling strategies and product bundling for optimized sales and enhanced customer experience

Tic-Tac-Toe Agent by Alpha-Beta Pruning Minimax | Python **Jan. 2024**

- This project involves creating a TicTacToe playing AI using the Alpha-Beta Pruning Minimax algorithm. The aim is to showcase effective AI decision-making in TicTacToe through the application of advanced algorithms.

Jieson R. Orain

jieson.orain@tup.edu.ph • 0939-762-5083 • linkedin.com/in/jieson-orain-256806287

EDUCATION

Technological University of the Philippines Manila **August 2025**
B.S. in Computer Science (Non- Stem) *San Marcelino Ermita, Manila*

Asian Institute of Computer Studies (Bacoor) **July 2021**
Information Communications Technology (TVL-ICT) *Salinas, Bacoor, Cavite*

TECHNICAL SKILLS

Languages: C, C++, Java, Python, HTML
Tools: Vs Code, PyCharm, Git, Figma, Adobe Photoshop
Skills: Project Management, Web Designing, Figma Basics for UI/UX, Video Editing, Graphic Designing

SOFT SKILLS

Soft Skills: Rapport Building, Project Management, Communications

PROJECTS

Manila Zoo Animal Identifier | Python, Java **January 2024**

- Manila Zoo Animal Classifier aligns with contemporary conservation goals by promoting awareness about endangered species and their conservation status. Beyond its educational value, it enhances the overall visitor experience, making trips to the Manila Zoo more interactive, engaging, and technologically relevant. In essence, this serves as a bridge between technology and wildlife conservation, emphasizing the pivotal role of innovation in fostering a sense of responsibility and connection with the animal kingdom.

General Apportionment Generator| Python **January 2024**

- The General Apportionment Generator is a practical tool designed to implement various methods for fair representation, employing Python and PyQt5 for an efficient user experience. Utilizing Python's 'fractions' and 'math' module, along with the NumPy library, it ensures precise calculations. The graphical user interface (GUI), crafted with PyQt5, facilitates easy exploration and implementation of various apportionment methods.