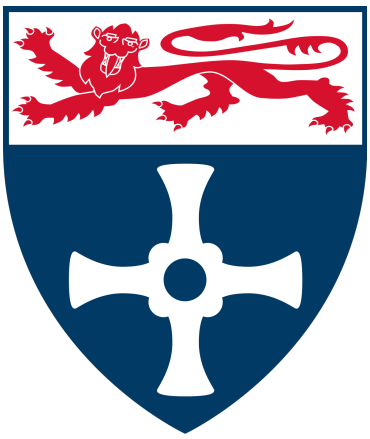
Enhancing Cloud Gaming Experience: Video Quality Prediction with User Activity



Zhaoran Wang

School of Computing
Newcastle University

This dissertation is submitted for the degree of
Doctor of Philosophy

January 2025

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Zhaoran Wang
January 2025

Acknowledgements

During my PhD research, I have been fortunate to receive several remarkable individuals' support, guidance, and involvement. I would like to take this opportunity to express my heartfelt gratitude to them, as their contributions have been invaluable to my academic growth and the successful completion of this thesis.

First, I sincerely thank my esteemed graduate advisor, Prof. Graham Morgan. His unwavering guidance, profound academic insight, and unwavering support have been the cornerstone of my research endeavours. Prof. Morgan's dedication to nurturing my intellectual curiosity and his ability to provide both academic and personal guidance have been truly invaluable. His mentorship has shaped my research skills and instilled in me a passion for pursuing excellence in my academic pursuits.

I would also like to express my deep appreciation to Dr. Gary Ushaw and Dr. Rich Davison for their significant contributions to my work. Their commitment to producing high-quality research, coupled with their active encouragement and expert advice, has been truly inspiring. I am grateful for the countless discussions and collaborations we have had, which have greatly enriched my understanding of the field and pushed me to think critically about my research.

I must also acknowledge the immense support and love I have received from my family members throughout this journey. Their unwavering belief in my abilities, constant encouragement, and endless patience have been the driving force behind my perseverance. I am forever indebted to them for the sacrifices they have made to support my academic aspirations and for being my pillars of strength during challenging times.

Finally, I am grateful to my friends who have stood by me throughout this research journey. Their camaraderie, understanding, and words of encouragement have been a constant source of motivation. The laughter shared experiences, and moments of respite we have enjoyed together have made this journey all the more memorable and enjoyable.

Abstract

With the rapid development of cloud gaming technology over the past twenty years, cloud gaming has begun to challenge traditional gaming modes. Facing an increasing number of cloud gaming platforms, reducing service costs and improving service quality have become key focus areas. This study proposes an innovative cloud game video quality prediction model. It aims to predict the future trend of video quality by analyzing simulated cloud game players' interaction data and cloud game streaming video quality parameters. In our research, by collecting and analyzing multiple sets of relevant data, we established the Linear Time Series Analysis Model and the Recurrent Neural Networks Based Model, and compared them to derive a new prediction model for cloud game streaming video quality. This model not only predicts the future trend of video quality to optimize user experience and service cost, but also derives multiple targeted models to adapt to the specific needs of different game genres through users' behaviors in different games. Through simulation experiments, we verified that these models are effective in various game scenarios and can accurately predict changes in cloud game streaming video quality. The research results demonstrate the applicability and accuracy of our model for cloud game video stream quality prediction, providing cloud game service providers with strategies to optimize video quality based on user behavior dynamically.

In addition, this study explores the potential applications of the prediction model in streaming media and cloud gaming services, providing theoretical support for the future direction and improvement of cloud gaming services. Finally, we anticipate that further refinement of user behavior analysis could enhance the performance of the prediction model and the cloud gaming experience. The outcomes of this research offer strong support for improving the quality and efficiency of cloud gaming services, and have practical application value and theoretical significance.

Key words: Cloud Gaming, Video Quality Prediction, Video Multimethod Assessment Fusion, User Activity Index, Time Series Models, Nonlinear AutoRegressive Model with eXogenous Inputs, Recurrent Neural Networks, Quality of Service, Quality of Experience

Table of contents

Chapter 1

Introduction

1.1 Overview

Cloud gaming, also known as gaming on demand, is a network gaming technology based on cloud computing. [149] This technology allows devices with relatively limited graphic processing and computational power to run high-quality games. In cloud gaming scenarios, games are not run on the player's gaming terminal but are operated on cloud servers. The game scenes are rendered by the cloud servers into video and audio streams, which are then transmitted over the network to the player's gaming terminal. The player's gaming terminal does not need strong graphic computation or data processing capabilities; it only needs the necessary media playback capabilities and the ability to receive player input commands and send them to the cloud server.

Over the past twenty years, cloud gaming has been a promising research area, but it is still in its early stages of development. Although its technology continues to advance, it has not yet been able to challenge the market dominance held by traditional gaming modes. However, as the demands of game content on gaming hardware continue to grow, finding new technologies to meet the ever-changing needs of game players is becoming increasingly important. At the same time, as the pace of modern life accelerates, gamers' demands for a seamless cross-platform gaming experience are also rising. Major technology and gaming companies are increasingly focusing on cloud gaming development, making it an essential component of the gaming entertainment industry and accelerating technological innovation to provide players with a better cloud gaming experience. Therefore, improving the quality of the cloud gaming experience has become a focal point of research.

The core of today's cloud gaming services is built on streaming video platforms, where players receive game content as streaming video. Therefore, dynamic video compression technology is critical to reduce company costs and enhance the cloud gaming experience for players. Most current cloud gaming services still use traditional streaming video compression technology, both in terms of video compression coding and transmission methods, the same as traditional streaming video platforms. However, cloud gaming services are very different from traditional streaming video platforms, both in terms of the content transmitted in cloud gaming streaming

video and the interaction with users. Therefore, based on the streaming video transmission technology, it has become a core focus to improve and create a new type of transmission service for cloud game streaming video based on the characteristics of cloud game service. First, traditional streaming video uses standard compression formats such as H.264/H.265, and most video content is recorded by camera equipment. There is much uncertainty in the change of video quality. But cloud game streaming video is rendered by cloud game servers, and unlike traditional video recording methods, the video quality of cloud game streaming video can also be predicted according to the game content to adjust the compression rate dynamically. [53] In addition, cloud game services are interactive and fundamentally different from traditional streaming video. Both cloud game streaming video content and game data processing depend on user behavioural data.

Therefore, this study establishes a new research direction to address the shortcomings of current cloud game services. By analysing the User Activity Index(UAI) and adding it to the analysis of different kinds of game content, we can grasp the changes in cloud game streaming video content. These interaction behaviour data are defined, standardised, and analysed using linear models and neural networks to find out the characteristics of cloud game user interaction behaviours and changes in cloud game streaming video content. Ultimately, we can judge and predict the changes in cloud game streaming video quality parameters through the Cloud Gaming User Activity Index(UAI) to provide reliable data support for dynamically adjusting the compression of cloud game streaming video, which ultimately reduces the company's costs and enhances the players' cloud gaming experience under the condition of limited resources.

1.2 Contribution

The core of cloud gaming services is still composed of streaming video transmission technology. Nowadays, when transmitting streaming video, dynamic compression algorithms are applied to optimise transmission efficiency, thereby reducing transmission costs and maximising the quality of the user experience. To better assess the quality of user experience, streaming service providers like Netflix have proposed definitions for video quality parameters. By calculating the difference between the video on the user side and the original video data, the Video Multimethod Assessment Fusion(VMAF) is obtained, which is used to provide feedback and optimise the dynamic compression algorithm. It is well known that traditional streaming video content transmission is broadcast from the server to the client and is not affected by the subjective behaviour of the user on the client side. However, in the context of cloud gaming, the streaming video content transmitted between the cloud and the user end has interactivity, meaning that the user's control input to the game has a significant impact on the content of the cloud gaming streaming video, which inevitably affects the calculation of VMAF. Therefore, studying the influence of the Cloud Gaming User Activity Input(UAI) on VMAF and providing a reference for improving the calculation method of VMAF, thereby better evaluating the user's cloud gaming experience quality, is undoubtedly a precious research topic. Based on this, this

thesis investigates the relationship between the Cloud Gaming User Activity Input and VMAF to predict changes in VMAF through UAI. The main contributions of this thesis are as follows:

1. We have proposed a User Activity Index (UAI) based on game content and user behaviour. This index, derived from a comprehensive analysis of the results, provides a practical tool for assessing the user's activity level during a gaming session. By analysing user input behaviour on the keyboard, mouse, and controller while playing games, we have also briefly analysed the relationship between changes in various types of game content and control inputs, further enhancing the practicality of our research.

2. We employed linear model theory to model the relationship between UAI and VMAF. Since VMAF is a comprehensive indicator influenced by multiple factors, we hypothesise that its variation within a game session follows a stationary stochastic process. By analysing its auto-correlation properties, the order and parameters of the linear model are estimated, and a prediction model for UAI-VMAF is established. However, when we conduct experiments to simulate the linear models, it is proved that because of the too complex relation between UAI and VMAF, the linear models almost cannot be used to make predictions. However, this study works to lay the foundation for the next stage.

3. We utilised recurrent neural networks to model the relationship between UAI and VMAF. Based on the assumption that the variation of VMAF within a game session follows a stationary stochastic process, NARX models are established by collecting data from different types of games. Due to the dynamic nature of UAI and VMAF sequence data, the delay parameter settings in NARX are crucial. We thoroughly analysed the design principles of time series delay and guided the correct selection of the delay settings in NARX.

4. In the experimental stage, we first defined the metrics for evaluating the prediction performance of the models. Then, we selected multiple cloud gaming streaming videos for analysis, including different types of games, such as action and card games. By combining VMAF with the corresponding user activity index of the game content, we successfully verified the significant impact of user behaviour on the quality of cloud gaming streaming videos. Based on different game types and compression rates, we chose appropriate NARX parameters and successfully achieved UAI-VMAF prediction using NARX. The experiments revealed that for action games, due to the frequent control inputs, the UAI changes were relatively gradual, and the periodicity of the corresponding VMAF variation pattern was not significant. As a result, the training and simulation effects of the NARX model could have been improved. However, after data augmentation on the dataset, the results improved considerably.

In summary, we conducted simulations and comparisons on datasets of multiple game types and compression bitrates using time series models and non-linear autoregressive models with exogenous inputs. Through the simulation results, we can better determine the impact of the user activity index on cloud gaming video quality parameters. Furthermore, we can use

the simulation models to predict cloud gaming video quality parameters based on future user behaviour indicators, ultimately providing theoretical guidance for the dynamic compression of streaming videos. These findings have theoretical significance and practical implications for reducing transmission costs and improving user experience quality, ultimately meeting the needs of cloud gaming service platforms.

1.3 Chapter Outline

In this thesis, all chapters focus on how to improve the quality of cloud gaming services, aiming to clarify the core research content of how to predict the quality of cloud gaming video streaming through the user activity index, as well as the contribution of this research to the field of cloud gaming. The thesis structure flow chart is as Fig 1.1.

Introduction This chapter provides a concise overview of the evolution and structure of cloud gaming services, highlighting the unique challenges they face. It also presents the cutting-edge aspects of contemporary cloud gaming services and the innovative contribution of our research to enhancing their quality of experience.

Background This chapter is divided into three parts. The first part mainly introduces the development history, infrastructure, advantages, and disadvantages of cloud gaming, lays the foundation for the background of this study, and introduces the importance of industry development and research. The second part introduces the background of the collection method of cloud game user operation behaviour and conducts a detailed analysis of different game types, thereby laying a theoretical foundation for the User Activity Index (UAI) defined in the subsequent research. The third part briefly introduces the development of streaming video, the core technology of today's cloud gaming services. It leads to the definition of video quality parameters for cloud gaming streaming video. Through the detailed introduction of video quality parameters and the discussion of Video Multimethod Assessment Fusion(VMAF), we can better understand the theoretical basis of the research background of this thesis, thereby further determining the following research content and direction.

Methodology This chapter is structured into two main sections. The first section integrates the background chapter with the core architecture of relevant cloud gaming services to define the User Activity Index parameter. It also details the process of collecting user behaviour index and VMAF parameters, providing a robust data source for the experimental analysis. The second section applies the theory of cloud gaming services to explain the Time Series Model and Non-linear AutoRegressive Model with eXogenous Inputs (NARX), particularly their application in predicting cloud gaming video quality parameters. It also defines the relevant parameters and model evaluation indicators, laying the groundwork for subsequent simulation and experimental results.

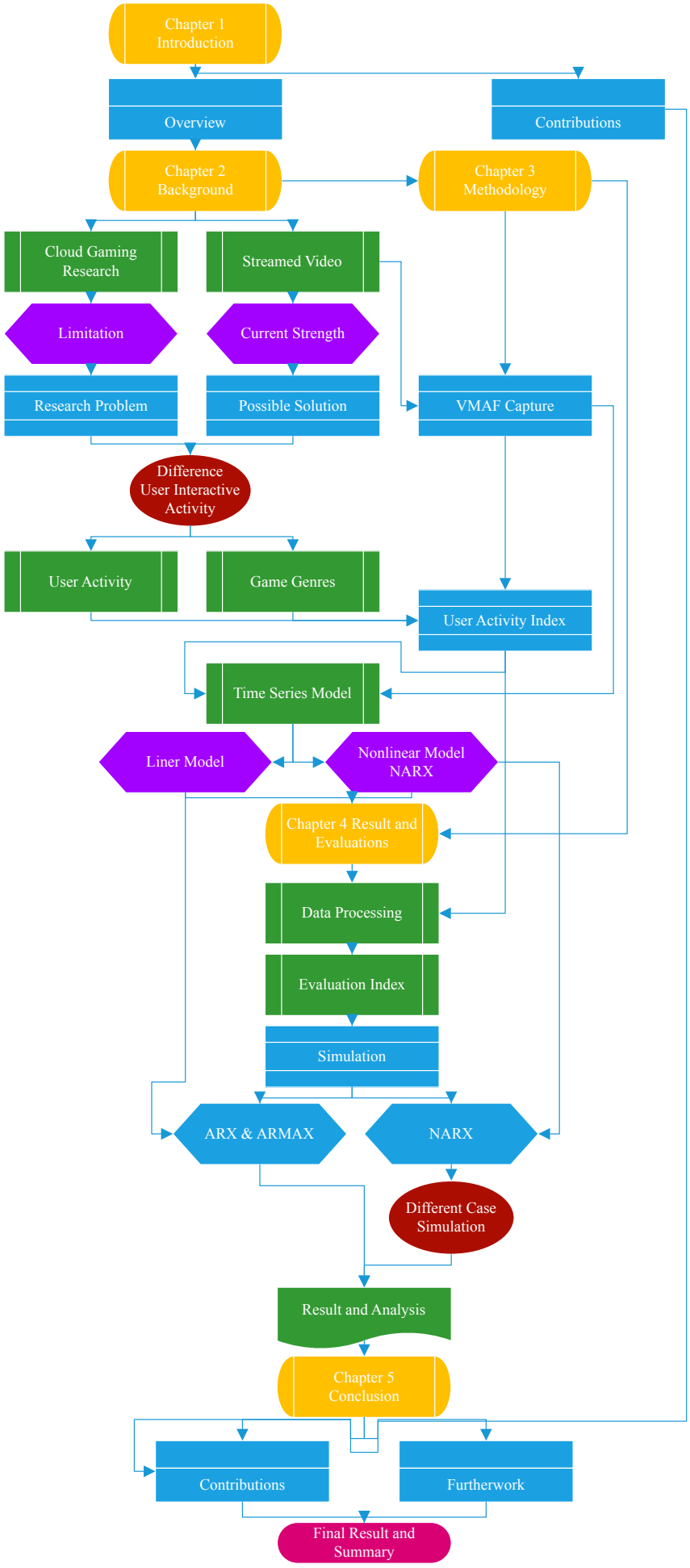


Fig. 1.1 Thesis Structure Flow Chart

Results and Evaluations This chapter can be divided into two parts. In the first part, we introduce the detailed processing of experimental research data and the normalisation of data based on the research background and research methods. In the second part, we introduce the result evaluation indicators we established based on the research on the Time Series Model and Non-linear AutoRegressive Model with eXogenous Inputs (NARX) in the methods chapter. After establishing the indicators, we simulated the two models and evaluated the simulation results to draw experimental conclusions.

Conclusions This chapter consolidates the findings of this research, emphasising the practicality of predicting cloud game streamed video quality parameters through the User Activity Index. It restates the significance of this research in enhancing the quality of experience in cloud gaming. The study's limitations are also outlined to guide future research.

Appendix The appendix contains supplementary materials, including part of the original data collected and used in the experiment and part of the pseudo-code, to help readers better understand the research content and contribution of this study.

Chapter 2

Background

2.1 Overview

In this chapter, we start with the history and architecture of cloud gaming services and briefly summarise the development history of cloud gaming and the basic architecture of today's cloud gaming services. Afterwards, by understanding the advantages and disadvantages of today's cloud gaming services, we can further deepen our understanding of the core research direction of this thesis and thus understand the core background of cloud gaming services. Next, this chapter briefly introduces the characteristics of streaming video and the concept of video quality parameters. Introducing video quality parameters in detail to better explain the characteristics and usage of VMAF lays the background theoretical foundation for the research direction of this thesis. Finally, this chapter introduces various game activity collection methods and the characteristics of these different collection methods. It combines the characteristics of various game genres to conduct an in-depth discussion of user behaviour in different games.

2.2 Cloud Gaming Background

2.2.1 History

Cloud gaming originated in the early 2000s. With the continuous improvement of Internet bandwidth and the advancement of cloud server cluster technology, service provision based on cloud computing was gradually applied. Therefore, the idea of running the game on a cloud server cluster and transmitting the content to the user in real-time via the Internet was finally realised. [19] At that time, G-Cluster first publicly demonstrated the prototype of its cloud gaming platform. Although the demonstration effect at the time was not ideal, the technology was still in its early stages, and development was basic, with delays and unsatisfactory image quality. However, the G-Cluster platform has also successfully transmitted game footage to IPTV and mobile device platforms through the network, verifying the possibility that games can be processed and transmitted remotely. In 2003, G-Cluster began to commercialise the cloud gaming platform. Its initial sales were still based on the traditional direct one-time sale of

complete games or leasing games, and the available areas were also very limited. Although a subscription-based game service was later launched, it could still not be promoted on a large scale. It was unfamiliar to the public due to network delays and substandard data transmission speeds.

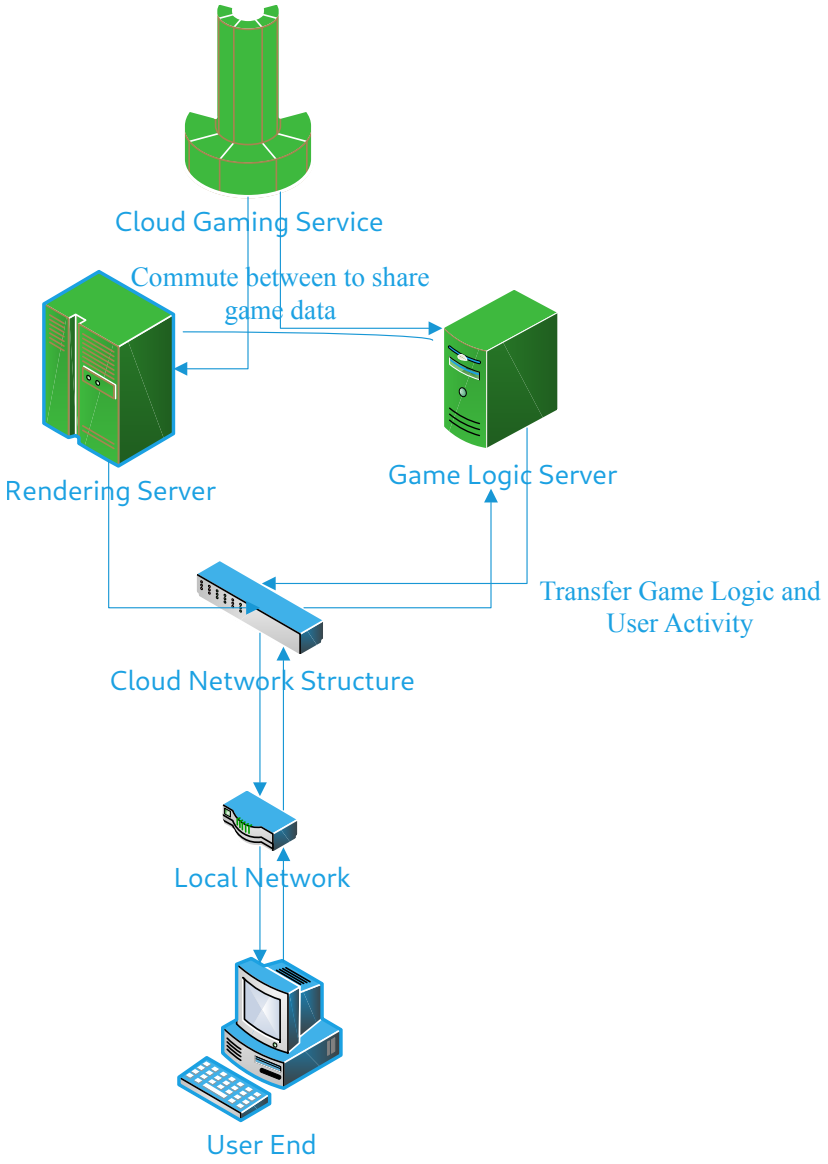


Fig. 2.1 Current Cloud Gaming Structure Example

It was not until 2009 that the launch of the OnLive platform marked the first large-scale commercial attempt at cloud gaming services. Benefiting from increased internet bandwidth and efficient video compression technology, OnLive enabled subscribers to experience real-time cloud gaming services and effectively reduced gaming latency. [72] OnLive also offered various other features, such as allowing players to record gameplay or watch others play games. Although OnLive successfully brought cloud gaming services into public view, they were limited by network connection quality and failed to gain support from most players. [23] Due to the high costs of maintaining and expanding servers, the company eventually faced a financial crisis. It ceased its cloud gaming services in 2015, marking the end of an era for cloud gaming but having a lasting impact on the industry, providing valuable experience and lessons for later services

like Google Stadia and NVIDIA GeForce NOW. OnLive's technological innovations continue to support today's cloud gaming technology.

Shortly after the closure of the OnLive platform, Sony launched the PlayStation NOW service, utilising its acquired Gaikai cloud technology to give players remote access to the old PlayStation game library. In 2023, Sony also launched the PlayStation Portal handheld device, which allows players to remotely access their PlayStation games via an internet connection, achieving a low-latency, high-quality gaming experience. With the introduction of the PlayStation NOW service, mainstream gaming companies officially entered the cloud gaming market, ushering in a new era for cloud gaming technology and development.

The same year PlayStation NOW was introduced, NVIDIA launched the GeForce NOW cloud gaming service, leveraging its powerful GPU service network and hardware strength to provide an outstanding end-to-end cloud gaming streaming service. GeForce NOW not only allows users to access games via a subscription model but also lets players access free cloud games. In 2019, Google launched the Stadia cloud gaming service. Based on the advantage of owning the streaming video platform YouTube and having a large number of cloud servers worldwide, Stadia offered an excellent cloud gaming service experience with high resolution, high bitrate, and high-quality video transmission. However, this also caused a sharp rise in the cost of Stadia's cloud services, leading to its cessation in 2023. Meanwhile, Microsoft's xCloud, as part of Xbox Game Pass, enabled Xbox Game Pass subscribers to stream games on various devices. Thanks to a rich game library and affordable prices, Xbox Game Pass gained widespread popularity among players, and Microsoft xCloud thus came into the view of many players, successfully integrating the cloud gaming service model into the traditional gaming ecosystem. Additionally, as the largest streaming video service platform, Netflix has also begun venturing into the cloud gaming field. Currently, users can experience over 80 different cloud games on various Netflix clients, such as TVs and smartphones. As the cloud gaming market expands with the entry of major manufacturers, the industry's development is also influenced by the advancement of 5G technology. The low latency and high transmission rates of 5G networks hold promising potential for cloud gaming, offering a more seamless and immersive gaming experience. [154] However, despite the market growth, cloud gaming still grapples with several challenges. Network stability remains a significant hurdle, as latency and internet speed often hinder players' experiences. The cloud gaming content library is also not as extensive as traditional game service providers, and some games may not perform well on cloud gaming platforms. Lastly, the cost of cloud gaming services poses a significant barrier to the industry's development, and finding ways to reduce costs and improve service quality is a key focus.

2.2.2 Current Development

At present, the overall architecture of cloud gaming services consists of a content layer, platform layer, network layer, and terminal layer, as shown in the Fig.: 2.2 below.

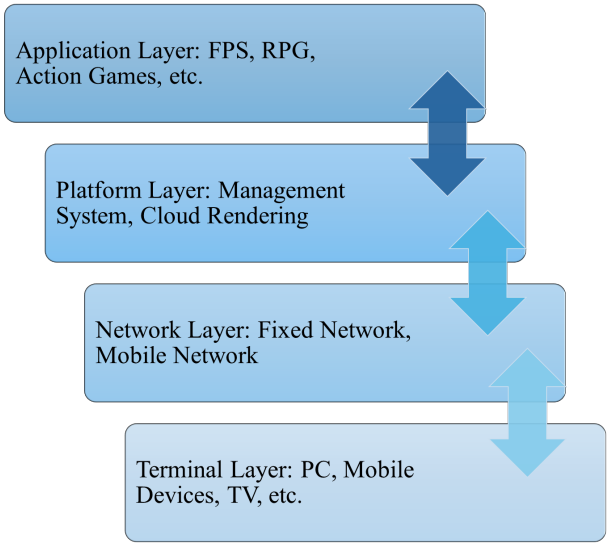


Fig. 2.2 Cloud Gaming Architecture

The content layer is primarily the game content source responsible for the online game provider. Generally speaking, most current games can be transplanted to cloud game servers, but there are still no native cloud games developed specifically for the cloud game service architecture. [113] Therefore, current cloud game services cannot fully take advantage of the cloud game architecture.

Platform Layer of Cloud Gaming Services

The platform layer primarily includes a management system and cloud rendering technology. The management system is mainly responsible for user and content management and the scheduling and allocation of related hardware resources. At the same time, cloud rendering technology is a critical component of cloud gaming. Cloud gaming rendering servers are equipped with high-performance GPUs to achieve the high-quality image rendering required for modern games. While the cloud game management system processes the game logic, the rendering server will perform geometric processing, rasterisation, colouring, post-processing, and other processes on the game scene. [49] Especially since today's games generally have high real-time characteristics, multiple groups of cloud game rendering servers will collaborate to perform parallel calculations, thereby reducing processing delays and increasing processing speed. Once rendering is complete, the game frames are passed to the video encoder and the game scene is converted into streaming video content. [112] In this step, the original data of the game video will be compressed to a size that is easy to transmit. The encoder usually uses encoding formats such as H.264 or H.265, which are efficient through frame motion compensation, frame content prediction, image blocking, entropy coding, etc. algorithm to obtain the best size streaming video without sacrificing the original video quality of the game. [68] At the same time, due to the real-time nature of streaming video, the encoder will adjust the bit rates of video transmission promptly according to network conditions, trying to prioritise the smooth operation of cloud games when the network is limited. In addition, dealing with cloud gaming input delay is crucial to the platform layer. When users feed data back to the server, the input delay can sometimes

be huge due to unstable network fluctuations. [89] In order to better solve this problem, some cloud game platforms adopt game-based content prediction algorithms. Although cloud game service providers cannot predict the player's following input very well, they can still render in advance based on specific game content scenarios. Content may be needed for the next scene, thereby minimising rendering time and making up for the loss caused by delay. Finally, the management system at the platform layer also needs to achieve dynamic server load balancing. Due to the fluctuation in the number of players and the different computing requirements of different games, the management system needs to allocate the load of the cloud game rendering server effectively in real-time. [64] In today's cloud gaming platforms, server virtualisation technology can better address the load issue. A physical server will be divided into several virtual environments through GPU pass-through and containerisation technology. Each environment can be responsible for different cloud game rendering tasks to achieve maximum dynamic allocation of server resources.

Network Layer of Cloud Gaming Services

The network layer mainly involves the network environment where users are located, which can usually be roughly divided into wired network or mobile network environments. In order to improve users' network environment, more and more network providers are beginning to provide all-optical networks to users. The all-optical network means that all connections from the cloud to the network layer to the terminal use optical fibre technology, and the network layer is simplified as much as possible to achieve network flatness. In terms of network forwarding, distributed OLT is used to achieve business switching and separation, DSCP is used to replace CoS to implement network slicing and unique bearer slices are established for cloud gaming services to achieve zero network jitter, thereby providing lower latency. [57] Finally, new fibre optic switches are used at the user end, and technologies such as 10Gbps network cards or WiFi7 wireless routers are added to achieve the optimal cloud gaming network environment. However, the current all-optical network architecture still proves difficult to popularise, the cost is still very high, and it basically only exists in theoretical experimental environments. Therefore, using an all-optical network architecture is not the first choice for cloud gaming service providers to provide higher service quality. In the mobile network environment, 5G technology brings unprecedented bandwidth to cloud games. By using millimetre wave frequencies, cloud gaming services can transmit high-quality streaming videos without the restriction of data bottlenecks. [91] At the same time, in the 5G network architecture, through the dense deployment of small base stations and the application of multi-access edge computing (MEC), cloud game delays can be effectively reduced, and the QUIC protocol can effectively solve the packet loss problem in online games. The full name of QUIC is Quick UDP Internet Connection, which is implemented based on UDP. The handshake process is simple, congestion control is improved, and multiplexing is enabled. QUIC also has the feature of Forward Error Correction (FEC). QUIC adds redundant data to each data packet. When packet loss occurs, the redundant data of other packets can directly recover the lost packet. This ensures data integrity and avoids delays introduced by TCP retransmissions.

Finally, 5G network slicing technology can provide customised virtual network slicing for cloud gaming platforms to optimise the network environment further. However, like the wired network environment, the cost of deploying a large number of 5G base stations and 5G edge computing servers is very high, and today's cloud gaming service providers cannot adopt this solution. [134]

User Layer of Cloud Gaming Services

There is the user layer of cloud gaming services, which is also a link that is often overlooked in the cloud gaming service architecture. Although cloud gaming processing and rendering all rely on the cloud game rendering server, the client still needs to decode the received streaming video. The user-side decoding speed and secondary rendering efficiency determine the smoothness of the user-side game. For example, PlayStation Portal is a client device uniquely customised by Sony for the PlayStation cloud gaming experience. Compared with previous traditional client devices, it can play cloud games from PlayStation more smoothly. In addition, the user-side input mechanism is also critical to the quality of cloud gaming services. Whether users choose a keyboard, mouse, controller, or touchscreen device as a game controller, these input devices' response speed and accuracy determine the immediacy of cloud gaming services. Collecting input data with minimal delay and sending it to the cloud game server is also essential in improving the quality of cloud game services. [14] Finally, in order to ensure the real-time nature of cloud games, audio and video synchronisation on the client side is also crucial. In addition to using today's buffering technology, some cloud game clients use pre-caching technology to preload part of the audio and video content to the user in advance to improve the cloud game service quality experience further.

Combining the above cloud game service architecture, we can find that today's cloud game services have developed into a service model that closely integrates multiple system architectures, software, and hardware. In addition to further improving the smoothness of cloud gaming services, reducing the operating costs of users and cloud gaming service providers has also become the core content of cloud gaming development.

2.2.3 Strengths and Limitations

The advent of cloud gaming services has brought significant advancements to the gaming industry. [122] However, these services are still in their early stages, with several challenges that need to be addressed. In this analysis, we will delve into the advantages and disadvantages of cloud gaming services, shedding light on their current state and future potential.

Strengths

First, for users, cloud gaming services are highly accessible, and users can access high-definition game content at any time without being limited by hardware devices. At present, many large-scale games still require players' high-end devices. In order to experience the full content of such games, most players need to purchase expensive hardware to support the operation of

these games. The current mainstream processors and graphics cards are updated every two years, which places significant financial pressure on users. In addition, more high-end hardware can only be used on PCs, and more mobile phones and laptops are needed to meet the hardware standards of high-definition games. The emergence of cloud gaming services can address these issues effectively. Users can play games anytime and anywhere through the Internet without being restricted by hardware. At the same time, users can also play games on various operating devices, such as TVs and mobile phones, and achieve seamless switching, completely unaffected by hardware restrictions. Second, with the increase in hardware requirements for traditional games and the improvement of picture quality, the storage space occupied by traditional games is also increasing sharply. [150] Currently, the average size of mainstream large-scale games is about 100GB, and some games have reached 200-300GB due to a lack of optimisation. This means that users still need 30 minutes to 1 hour to download and install games under extremely high network conditions, and it will also take up a lot of hard disk space for users. For areas with poor network environments, users may wait a day or even longer to download and install. Cloud games perfectly eliminate these obstacles. Cloud game service updates are uniformly managed by the server side, which can always ensure that users have the latest version. Users can directly start the game without waiting, enhancing the immediacy of user games and improving the quality of user experience. In addition to providing users with a more immediate experience, cloud game services can optimise transmission quality through algorithms and dynamically adjust users' visual and auditory experience, which is also one of the specific contents discussed in this study. [124] Finally, cloud game services benefit from the community nature of online game service platforms and their advantages, such as not being restricted by space, which can bring users a better social experience. Users can watch friends' game sessions in real-time from the first-person perspective through the platform, share game content, and even operate at the same time and join friends' ongoing games anytime and anywhere. In addition, in some countries, Internet cafe culture is very popular, and some users meet in Internet cafes to play games together. Cloud gaming services can bring a better gaming experience to these users because there are no device restrictions, users can choose more diverse places for game gatherings, and the user's gaming experience can be enhanced.

For game developers, cloud gaming services also bring new content to game design and development. First, because cloud gaming reduces the restrictions on hardware requirements, game developers can develop more complex and content-intensive large-scale games. Games can have higher-quality graphics content and more complex physical simulation calculations and no longer worry about the performance of users' devices. In addition, game developers no longer have to worry about compatibility issues in the development of cloud games. Cloud gaming services are all deployed on cloud servers. A game can be easily compatible with different operating systems and device environments, and developers no longer need to perform special optimisations for individual systems, which can significantly reduce development costs and time. At the same time, in cloud gaming services, all game updates and maintenance will be completed on the cloud gaming server, and all game data will be stored on the cloud server. This

can effectively reduce the cost incurred by game developers when updating, and developers can better control the access and distribution of games, reducing game piracy. [9] Moreover, cloud game services can implement a rolling update mechanism through the virtualisation technology of cloud game servers. Under the premise that the game service is not suspended, different game servers can be updated in batches to achieve an uninterrupted and non-stop update mechanism. Regarding game anti-cheating, game developers can deploy all content on the server side, and user devices are only used to receive and send operation instructions. On the server side, game developers can carefully analyse and supervise all data in real-time, thereby effectively controlling the generation of game cheating. At the same time, because all game data is deployed in the cloud, cheating programs cannot modify game files and storage data, significantly increasing the difficulty of developing traditional cheating programs, which effectively eliminates game cheating. At the same time, game developers can also better monitor user in-game purchases. In traditional game services, false transactions, credit card fraud, etc., have been essential problems in in-game purchase services in recent years. Thanks to the high inheritance of cloud services and the application of related blockchain technology, all transactions and in-game purchase services are traceable in cloud game services, which can effectively prevent risks such as transaction data leakage and transaction fraud. [65] Finally, cloud game services can also better expand the market scope of game developers. With the application of cloud gaming services, eliminating device and geographical restrictions can enable more users to access high-quality games. Through the game community features of online cloud gaming service platforms, game developers can more easily promote their works, provide more trial and marketing content, attract more users, and improve purchase conversion rates. Game developers can also collect more accurate player behaviour data, optimise game content, and give players a better gaming experience.

Limitations

The previous section shows that cloud gaming services have brought revolutionary innovations to the gaming industry. However, because cloud gaming services are still an emerging field, many challenges remain.

First, for users, the core requirement of cloud gaming services is a stable and high-speed Internet connection. When the network connection is unstable or the bandwidth is insufficient, the video compression rate will increase, and the quality of the game screen received by users will be significantly reduced because the core of the cloud gaming service uses streamed video technology. At the same time, players will suffer from all the problems of traditional online games, such as delays and connection interruptions. In addition, because cloud gaming servers may be deployed in multiple regions, the physical distance between users and cloud gaming servers is sometimes very far. [104] This also leads to increased latency in cloud gaming services, especially for games requiring fast response, such as first-person shooters or racing simulation games; the user experience will be significantly reduced. When users need to play cloud games on mobile networks, the cost of mobile network traffic will also become an issue that needs

attention. Because cloud gaming services are based on streaming technology, and most games require high-resolution and high-bitrate streaming video transmission, users will consume data, increasing the cost of using cloud gaming services. Since the streaming compression technology currently used by cloud gaming services does not optimise the cloud gaming services themselves, and the compression technologies of different platforms are different, the compatibility of cloud gaming services is limited by traditional streaming compression technology, which causes certain troubles for users. In addition to user network costs, game purchase and ownership issues are points that need attention in cloud gaming services. Most cloud gaming services adopt a subscription access system. Users do not own a copy of the game. [1] When users no longer want to use cloud gaming services or need to change cloud gaming service providers, the original games will be removed from the user's game library, and users will lose access to the games, causing certain troubles. In addition, since the current cost of cloud gaming services is still high, the pricing of cloud gaming services for users is also high, and long-term subscriptions to cloud gaming services have become a considerable expense for users. For some users who often play small games, the final cost may be greater than the investment in hardware equipment. Finally, due to the architecture of cloud gaming services, a large amount of user game data and operation data will be transmitted to cloud storage in real-time. The security and privacy of cloud data are also important issues. Suppose the security measures of cloud gaming service providers are not sufficient. In that case, user data is very likely to be leaked, causing some users to worry about the personal privacy issues of cloud gaming services.

For cloud game operators and developers, there are still many issues that need to be solved in cloud game service technology. First, the initial investment cost of cloud game services is often very high, including purchasing and constructing large high-performance server clusters, leasing network facilities and equipment maintenance, which is impossible for some small game content providers. Therefore, the current cloud game service providers are all large enterprises, which has also reduced industry competitiveness and could be more conducive to the long-term development of cloud game services. In addition to the basic investment in server hardware equipment, network costs are a significant problem for cloud game services. [118] Both large service providers and small start-ups face this problem. Because cloud game services need to establish server clusters within a certain physical distance from the user group to ensure that the user's physical delay is within an acceptable range, cloud game services need to deploy their server clusters in various places around the world, and find corresponding network providers in different countries and regions to provide network bandwidth for cloud game services. Therefore, the subsequent maintenance costs of cloud game services are still very high. In addition, due to policy differences in various countries and regions, deploying cloud game services globally and providing consistent services has become more complex. Similarly, different countries and regions have different policies on game content, and specific games also involve different agent distributors in different regions. Therefore, cloud gaming service providers need to spend more time and money to deal with the issue of game copyright. Finally, due to the great volatility of demand for cloud gaming services, service providers need many cloud computing resources to

allocate server loads to cope with user access during peak periods. [157] When the game version is updated or a new game is released, the number of users accessing the cloud gaming server may be several times the usual number, and the bandwidth pressure will increase simultaneously. How to achieve elastic allocation and load balancing of servers and bandwidth is an important issue facing current cloud gaming services.

Taking Google Stadia as an example, the Stadia service platform can bring users extremely low latency and high-quality game video images. However, the Stadia service also relies on Google's large number of data centres and cloud service facilities worldwide. These hardware foundations guarantee an excellent user experience, but this also brings huge network and server costs to Google. Although the top price of Google Stadia is already at a high level, it still cannot achieve a balance between income and expenditure for cloud gaming services. Ultimately, Google Stadia completely shut down its cloud gaming service in 2023. Amazon Luna is another example, which relies on a wide range of AWS cloud service infrastructure due to the high scalability of AWS cloud services. Amazon can effectively expand or shrink its cloud gaming service deployment according to the number of users. However, this scalability often comes at a high cost, and in order to meet the peak traffic period when the game is released, other services may be sacrificed to meet the needs of cloud gaming services. At the same time, because Amazon is not a traditional game service provider, Amazon has no game content foundation, and finding cloud games that can be transplanted has become one of its main difficulties. Amazon tried to create its game team to create a new game for cloud gaming services, but due to the lack of technical and experience support, users did not support this project, and Amazon eventually stopped the development of native cloud games based on cloud gaming services. Microsoft xCloud can still provide users with stable cloud gaming services thanks to the advantages of the Microsoft Azure cloud platform. At the same time, based on the fact that Microsoft currently has a large number of game development studios, it has improved the stable content guarantee for its cloud gaming platform. NVIDIA GeForce NOW cloud gaming service relies on efficient server performance. [123] Thanks to its excellent GPU processors and large AI server hardware, NVIDIA has a large advantage in hardware costs and can provide smooth cloud gaming services for players in some regions. In addition to these two traditional cloud gaming service providers, Netflix has started deploying its cloud gaming services. Because the core of cloud gaming services is still streaming video technology, as the best streaming video provider, Netflix has very advanced dynamic compression technology and is further optimising dynamic compression technology through video quality parameters, which brings potential advantages for Netflix to reduce the operating costs of cloud gaming services.

Through the above cases and analysis, we can find that the main problems currently faced by cloud gaming services are all related to network and service costs. [105] How to dynamically adjust the quality of cloud gaming streaming video content received by users under limited network resources so as to enhance the user's Quality of Experience while reducing service costs has become the main problem in the development of cloud gaming services. The following research in this thesis also aims to provide an innovative solution to this problem.

2.3 Streamed Video

2.3.1 Current Development

Streaming video technology, also known as streamed video, is a highly efficient technology that enables users to receive and play video content in real-time. This process eliminates the need to download the entire video file, significantly reducing waiting time and enhancing the user experience. Currently, streaming video is extensively utilised on major online video platforms and some IPTV. Streaming video technology can divide large videos into many small video

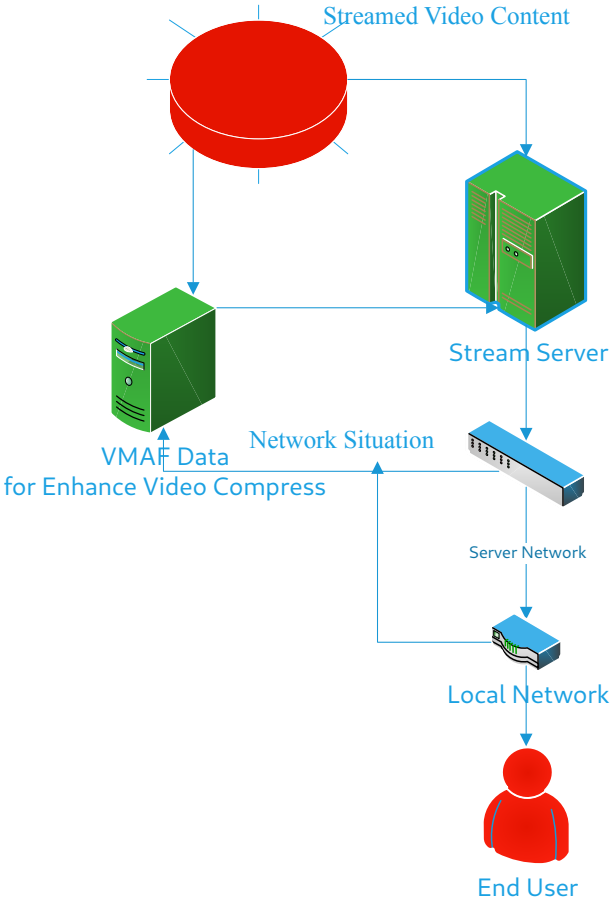


Fig. 2.3 Example Streamed Video Platform Structure

data packets and send these data packets directly to users through the network while processing the original video. The user end will immediately process these data packets and decode the video, thus realising the concept of immediate playback without receiving all the video content. Among them, video encoding compression, data packet segmentation and adaptive bit rate streaming are the core contents of this technology. Many factors in today's network environment affect the user's bandwidth and network quality. [50] Compressing and encoding the original video content can significantly reduce the data volume, thereby optimising the transmission efficiency. The commonly used streaming video encoding formats are H.264, H.265 (HEVC), VP9, etc. [135] These formats can effectively guarantee video quality and reduce data volume. By compressing and encoding the original video, the video server can divide the data into many small segments, usually in seconds, to better follow the changes in the network environment for

data transmission and reduce the waiting time of users. [3] Finally, the emergence of adaptive bitrate streaming technology has solved the problem of network fluctuations very well. This technology allows players to select pre-processed videos of different qualities according to users' network conditions. When the user's network fluctuates and the conditions are poor, the player usually automatically reduces the video quality to avoid the interruption caused by the user waiting for the video to load, thereby effectively improving the user experience.

In addition to the above three key technologies, the transmission protocol of streaming video is also very critical. Different network transmission protocols can ensure that data packets are effectively and losslessly transmitted. The first is the HTTP Live Streaming (HLS) protocol, the most widely used one. [48] Apple developed this protocol and supports adaptive bitrate streaming technology. It can be very well applied to multiple devices and complex network environments. Dynamic Adaptive Streaming over HTTP (DASH) is an open standard known as MPEG-DASH. [8] It also supports adaptive bitrate streaming technology. [12] In addition, the DASH protocol also allows data to be received from multiple sources simultaneously, which can more effectively buffer video package files for users. In addition to the two protocols mentioned above, the Real-Time Messaging Protocol (RTMP) is used on many live broadcasts and video platforms. Macro-media originally developed RTMP, which has the characteristics of low latency. However, due to the widespread use of HLS and DASH, most video servers no longer use RTMP. For example, Netflix, the world's largest streaming video service provider, mainly uses Dynamic Adaptive Streaming over HTTP (MPEG-DASH) protocol, combined with ABR (Adaptive Bitrate), to give users the best online viewing experience for video. [146]

Netflix currently uses two main encoding formats, VP9 and H.265 (HEVC). The VP9 format is well known for its high compatibility. This format can run well on different devices, operating systems, and browsers. It can effectively improve bandwidth efficiency and perform low-bitrate, high-quality video transmission in a limited network environment. H.265 is aimed at handling a large number of high-definition resources. H.265 is based on the H.264 encoding and has a more innovative compression algorithm. [44] It can efficiently and quickly process 4K high-definition video resources, but it also has high requirements for user device hardware. In addition to the encoding format, the distribution network used by Netflix is the key to its ability to provide a high-quality video viewing experience. Netflix's self-built content delivery network (CDN) is also known as Open Connect. Open Connect is designed to store and deliver Netflix's entire video library. It allows Netflix to deploy streaming video content separately to servers closer to the user's geographic location. By selecting the server based on the network environment and geographic location, users can always obtain video content with low latency, high speed, and high quality. This technology is also the core of Netflix's ability to provide high-quality streaming video services worldwide. At the same time, Netflix user clients can dynamically adjust the buffer size to adapt to the different network environments in which user devices may be used and optimise playback decoding efficiency. On the server side, Netflix will also pre-encode each video content in multiple qualities so that users can quickly obtain the required video version on various devices or network environments. Finally, Netflix also actively uses machine learning

and big data to analyse users' streaming video viewing experience. By responding to the analysis of video content and video quality to improve video encoding, the quality of user video viewing is ensured, and the video quality parameter indicator of Video Multimethod Assessment Fusion (VMAF) is proposed.

2.3.2 Adaptive Bitrate Streaming(ABR)

Adaptive Bitrate Streaming (ABR) is a crucial technology optimising video streaming experience. [136] Its core principle involves dynamically adjusting the quality of the video stream based on the user's current network environment through the selection algorithms of different clients, thereby reducing buffering time and enhancing the viewing experience for users.

Initially, video content is encoded at various bitrates on the server side, with each version having different resolutions and bitrates to cater to the network conditions of different users. [35] These versions are typically segmented into many fragments, each lasting a few seconds, in preparation for dynamic adaptation to the user's network environment. The player dynamically chooses the most suitable video stream bitrate based on the client's selection algorithm. When a decrease in network bandwidth is detected, the client lowers the video quality by selecting lower-quality segments for transmission to avoid buffering wait time. Conversely, when the network environment improves, the client selects higher-quality video segments to enhance the user experience. Importantly, some clients also adapt based on user Quality of Experience feedback, effectively saving bandwidth and improving the experience.

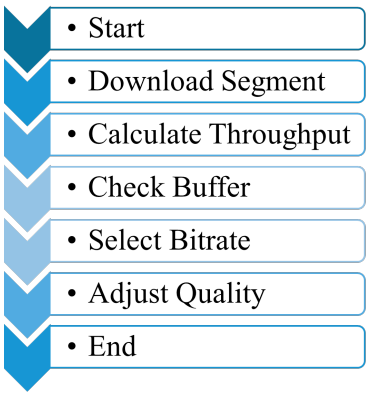


Fig. 2.4 Adaptive bitrate(ABR) Streaming Process

Currently, the core algorithms of ABR technology include the following, each implemented in various streaming video services. The Buffer-Based Algorithm monitors and maintains the data volume in the client's buffer to determine bitrate selection. When the buffer data volume is high, the algorithm prioritises a higher bitrate, suggesting that the network can support a higher data transmission rate. Conversely, when the buffer data volume is low, the algorithm immediately lowers the bitrate to prevent video playback interruption due to buffering. The Buffer-Based Algorithm is one of the simplest ABR algorithms, but it lacks responsiveness to fluctuations in network bandwidth and is not the optimal choice for video quality. [10] The Throughput-Based Algorithm measures the average rate of video segment downloads over a past duration. This

algorithm first monitors the time required to download different video segments to determine the average rate reflecting the current network conditions. Based on the average rate under the current network, it dynamically selects the most suitable video quality and bitrate parameters for transmission. This algorithm can accurately predict short-term changes in network conditions, thereby making more precise bitrate adjustments. The Hybrid Algorithm combines the features of the above two algorithms; it can predict the average rate of video segment downloads in the current network environment through the Throughput-Based Algorithm and also control the buffer size through the Buffer-Based Algorithm to prevent play interruptions due to insufficient buffering. This mixed algorithm effectively balances video quality and smoothness, thus widely used in many streaming video platforms. [58] However, the Rule-Based Algorithm is widely used on some older or simpler platforms. This highly programmed algorithm uses if-then rules to assess and select the appropriate bitrate. Establishing thresholds for buffering time, bitrate, and download speed adjusts the quality selection of video segments. This algorithm can be applied on most devices and is convenient for debugging, but it is not as flexible as other algorithms. Finally, with the advancement of machine learning technology, many mainstream streaming video platforms like Netflix have started using machine learning-based ABR algorithms. [7] Typically, by learning extensively from historical data about the relationship between user habits, video quality characteristics, and network conditions, those platforms make more effective predictions, thereby providing users with the best viewing experience.

From the characteristics of ABR technology discussed, it is evident that not all existing streaming video platforms consider the impact of user interactive behaviour. [34] This is because information transmission in traditional videos is unidirectional, and users do not need to provide feedback on the video content. However, cloud gaming services feature real-time interaction. Therefore, this research proposal aims to leverage machine learning ABR algorithms to establish the relationship between video quality in cloud gaming streams and user interaction data. The ultimate goal is to enhance the quality of cloud gaming services.

2.3.3 Recurrent Neural Networks (RNNs) in Adaptive Bitrate Streaming

Recurrent Neural Networks (RNNs) represent a category of artificial neural networks engineered for the identification of patterns in data sequences. They achieve this through the incorporation of loops within their architectural framework, allowing information retention over time. This "memory" capability permits RNNs to take into account antecedent inputs during the processing of new data, rendering them particularly efficacious for the analysis of time-series data and tasks wherein context and temporal dynamics hold significant importance.

Within the realm of Adaptive Bitrate (ABR) streaming, particularly pertaining to cloud gaming, Recurrent Neural Networks (RNNs) demonstrate significant utility due to their capacity to model and forecast temporal dependencies within network conditions and user interactions. Cloud gaming environments are marked by rapidly fluctuating network bandwidth and latency, coupled with dynamic user inputs such as keyboard and mouse actions. RNNs leverage historical data on network performance and user behaviour to predict future network states and resource

requirements. This predictive proficiency enables ABR algorithms to proactively adapt video bitrate and encoding parameters, thereby ensuring optimal streaming quality and mitigating issues such as buffering and latency. [99] By forecasting alterations in network conditions and user activity, RNN-based ABR systems can consistently uphold high video quality, thus offering a more seamless and responsive gaming experience.

Numerous research efforts have demonstrated the effective use of Recurrent Neural Networks (RNNs) in enhancing Adaptive Bitrate (ABR) algorithms for video streaming applications by leveraging their ability to model temporal dependencies and predict future network conditions, which is critical for optimising bitrate decisions and improving streaming quality; for example, Hongzi Mao introduced Pensieve, a system that utilises deep reinforcement learning with RNNs to automatically learn optimal ABR algorithms by training a neural network to make bitrate decisions based on observations of past network conditions, resulting in improved Quality of Experience (QoE) metrics compared to traditional rule-based methods; [73] similarly, research by Souane Naima demonstrated that integrating RNNs enhances the accuracy of bandwidth prediction and improves overall streaming performance; Huaizheng Zhang proposed DeepQoE, utilising RNNs to predict QoE in real-time for adaptive video streaming by capturing temporal variations in streaming data and network conditions, thereby guiding ABR algorithms to make more informed bitrate adaptation decisions; [84] these studies underscore the significant role of RNNs in advancing ABR strategies by effectively modelling the temporal dynamics of network conditions and user interactions, ultimately enhancing streaming quality and user experience in cloud gaming and other real-time video applications. [156]

In this thesis, we used NARX as our main research model. Comparing Non-linear AutoRegressive models with eXogenous inputs (NARX) to traditional Recurrent Neural Networks (RNNs) in Adaptive Bitrate (ABR) streaming reveals several advantages that make NARX models particularly suitable for cloud gaming applications; while both models handle sequential data, NARX models explicitly incorporate past output values and exogenous inputs—such as network conditions and user interactions—directly into their predictions, allowing for a more comprehensive and accurate modelling of the factors affecting video quality; this explicit inclusion enables NARX models to learn the influence of external factors on video quality more effectively than traditional RNNs, which may not distinguish exogenous inputs as distinctly; [15] moreover, NARX models capture long-term dependencies through delayed inputs and outputs without the complexities associated with training deep RNNs, such as vanishing gradients; [40] their ability to model complex non-linear relationships and avoid training difficulties makes them advantageous for ABR algorithms, where accurately predicting network fluctuations and user behaviour is critical for optimising video quality and reducing latency; [61] therefore, using NARX models in ABR streaming enhances video quality predictions, leading to better bitrate adaptation decisions, optimised resource utilisation, and an improved user experience in cloud gaming environments.

2.4 Video Quality Evaluation

To significantly enhance the quality of cloud gaming services, reduce the cost of such services, and most importantly, to elevate the Quality of Experience for users, the establishment of suitable evaluation metrics is not just crucial, but paramount. [62] The very essence of current cloud gaming services is rooted in streaming media video services; hence, the quality evaluation metrics for streaming media video have rightfully become a focal point of interest. [110]

The assessment of streaming media video quality encompasses a variety of metrics and methods. It involves numerous objective evaluation methods and parameters and should also consider users' subjective experience index. Subjectively, the number of buffering events is the most critical factor in the subjective experience index. [59] Frequent video buffering significantly degrades the user experience, affecting the subjective experience index. Additionally, playback smoothness is a key measure of subjective experience, where issues such as stuttering and latency directly impact the viewing experience. Similarly, playback start delay significantly affects user satisfaction with streaming video playback. Regarding content, viewing depth and user drop rate also impact the quality rating of streaming media video services. [67] However, these factors do not apply to cloud gaming services. On the objective front, metrics such as video quality switching frequency, average video quality, bandwidth usage efficiency, repeat buffering times, and error rate form the main content of the evaluation system. Video quality switching frequency can effectively observe the rationality of streaming video service algorithms. Average video quality can be deduced from average bitrate and resolution, directly affecting the viewing experience. [141] Lastly, bandwidth usage efficiency measures the consumption of network resources, thereby assessing the quality of cloud-streaming video services and potential resource wastage.

From the above assessment, it becomes evident that streaming video quality evaluation metrics encompass both strong objective content and substantial subjective content. The final evaluation method requires a multi-faceted and multi-methodological analysis, presenting a complex yet exciting challenge for our industry.

First, subjective evaluation methods rely on the direct feelings and ratings of viewers, typically obtained through user testing with video samples. Standard subjective evaluation systems include the Mean Opinion Score (MOS), Double-Stimulus Continuous Quality Scale (DSCQS), and Single-Stimulus (SS) scoring.

Mean Opinion Score (MOS): The MOS is a widely used subjective method for assessing video quality. In this approach, users watch multiple streaming video samples in a test environment and rate the quality of each sample based on their subjective feelings. [36] Service providers usually design these ratings in a numerical grade format for users to select from, culminating in an aggregated average score representing the MOS for video quality. [132] This method can intuitively represent user preferences and perceptions and is quick and convenient. However, due to variations in user preferences, a large sample size is necessary to ensure accuracy, and ratings can be influenced by video content and other factors. [126]

Double-Stimulus Continuous Quality Scale (DSCQS): In this method, users watch two test samples—a processed streaming video and the original video file. After making comparisons, users rate the two samples, effectively reducing the subjective impact caused by video content. However, getting a large number of users to watch original video files is challenging.

Single-Stimulus (SS): In this method, users watch only one streaming video sample instead of comparing two or more samples. Thus, their ratings are also based on a standardised grading system. This method is simple, quick, and suitable for a wide range of assessment samples, but it is the least reliable due to the subjective nature of individual user perceptions.

From these methods, it is clear that subjective evaluation systems, often influenced by user sensations, yield only a few regular results, making them less helpful for predicting video quality in cloud gaming. Objective video quality evaluation models, which rely on video parameter information, graphical structure, and compression rate, reflect regular changes in video quality. [82] Typically, objective video quality models are categorised into three types: Full-Reference (FR), No-Reference (NR), and Reduced-Reference (RR).

2.4.1 No-Reference (NR) Model

NR models do not depend on original or unprocessed reference videos and are primarily used in scenarios where original videos are unavailable. BRISQUE (Blind/Referenceless Image Spatial Quality Evaluator) is a frequently used NR model that employs Natural Scene Statistics (NSS) to assess image quality.

Initially, images are segmented into many small blocks, and each block is locally normalised to zero mean and unit variance to eliminate the effects of lighting and contrast. [79] The model then extracts features from the normalised image blocks, involving statistics such as pixel intensity, skewness, and kurtosis of pixel distribution. [117] These statistical values reflect the naturalness and distortion of the image. These data are then trained using machine learning models like Support Vector Machines, yielding a good prediction of the current video quality score. [152] Video-BLIINDS (Blind/Referenceless Image and Video Integrity Evaluation using Natural Scene Statistics) is also an NR video quality evaluation model based on NSS theory, commonly used to predict the visual quality of videos degraded by compression, transmission errors, frame rate changes, etc. [66] The model initially extracts natural scene statistical features from video frames, such as pixel intensity distribution in local areas, and analyses motion consistency and temporal stability between frames, thereby assessing jitters, blurs, or other temporal distortions in the video.

These feature values are input into machine learning models like Support Vector Machines for training, eventually determining the current video visual quality score. [115] [128] Video-BLIINDS is typically suitable for scenarios requiring automatic video quality monitoring, such as Content Delivery Networks (CDNs), which were mentioned in the previous section. However, its accuracy and robustness may be limited for complex or uncommon distortion types.

2.4.2 Full-Reference (FR) Model

FR models necessitate access to both the original undistorted video and the distorted version to evaluate quality by direct comparison. These models are prevalent when the reference video is available, enabling precise quantification of degradation introduced by processing, compression, or transmission. [139] FR models operate under the assumption that any deviation from the reference video signifies a loss in quality, and thus, they aim to measure these discrepancies accurately.

One of the most fundamental FR metrics is the Peak Signal-to-Noise Ratio (PSNR), which computes the logarithmic ratio between the maximum possible pixel value and the mean squared error between the reference and distorted images. [94] Although PSNR is simple and computationally efficient, it often correlates poorly with perceived visual quality because it does not account for the characteristics of the human visual system (HVS). [54] PSNR treats all pixel differences equally, disregarding the fact that humans are more sensitive to certain types of distortions than others.

To address the limitations of PSNR, the Structural Similarity Index (SSIM) was introduced as an FR model that aligns more closely with human perception by considering luminance, contrast, and structural information. [137] SSIM operates by sliding a window over the images and computing local statistics, providing a measure of structural distortion that correlates better with subjective visual assessments. [83] It assumes that the HVS is highly adapted to extract structural information from visual scenes, making it more sensitive to structural changes than to absolute luminance errors.

Further advancements in FR models led to the development of the Video Quality Model (VQM), which incorporates spatio-temporal features and models the HVS to predict perceived video quality more accurately. [92] VQM analyses changes in features such as edges, motion, and colour to assess the impact of distortions on visual perception. It quantifies video impairments by comparing features extracted from the reference and distorted videos, considering factors like blurring, noise, and jerkiness.

Another notable FR metric is the Video Multimethod Assessment Fusion (VMAF), developed by Netflix. VMAF combines multiple quality metrics using machine learning techniques to predict video quality as perceived by end-users. [16] It utilises features from several basic quality metrics, such as Detail Loss Metric (DLM) and Visual Information Fidelity (VIF), and employs a Support Vector Machine (SVM) regressor to model the complex relationship between objective features and subjective quality scores. [4] This approach enhances prediction accuracy by capturing the nuances of human visual perception and weighting different aspects of quality accordingly.

FR models are essential in applications where the original content is accessible, such as codec optimisation, video processing algorithm development, and benchmarking of compression techniques. [109] They provide a reliable assessment of video quality degradation introduced by various processing steps. For instance, during codec development, FR models help engineers

understand how different encoding parameters affect the perceived quality, enabling them to make informed decisions to improve performance. [74]

However, the reliance on reference videos limits the applicability of FR models in scenarios where the original content is unavailable, such as live streaming monitoring or user-generated content evaluation. [142] In such cases, it is impractical or impossible to obtain the reference video for comparison. Additionally, some FR metrics can be computationally intensive, making them less suitable for real-time applications where processing speed is critical. The need for significant computational resources can hinder their deployment in systems with limited hardware capabilities or in large-scale monitoring scenarios.

Moreover, while FR models like SSIM and VMAF offer improved correlation with human perception over traditional metrics like PSNR, they may still face challenges in accurately predicting quality for certain types of content or distortions. [6] For example, they might not fully capture the perceptual impact of complex temporal artefacts or fail to account for viewer attention mechanisms that influence perceived quality. [108] Ongoing research seeks to enhance FR models by integrating more sophisticated models of the HVS and incorporating factors such as visual saliency and temporal masking effects.

In conclusion, FR models play a critical role in scenarios where precise measurement of video quality degradation is required and the reference content is available. They offer valuable insights into the performance of video processing systems and contribute to the advancement of video quality enhancement techniques. However, their limitations necessitate the use of other models, such as No-Reference (NR) or Reduced-Reference (RR) models, in situations where the reference video is inaccessible or when computational efficiency is paramount.

2.4.3 Reduced-Reference (RR) Model

RR models occupy an intermediate position between Full-Reference (FR) and No-Reference (NR) models by requiring only partial information from the original, undistorted video to assess quality. These models are particularly advantageous in scenarios where transmitting the entire reference video is impractical due to bandwidth constraints, yet some side information can be shared between the sender and receiver. RR models function by extracting a compact set of features or statistics from the reference video, which are then transmitted alongside or embedded within the distorted video stream. At the receiver's end, these features are compared with those extracted from the received video to estimate the quality degradation. [17]

One prominent RR metric is the Reduced-Reference Structural Similarity Index (RR-SSIM), which extends the concept of SSIM by utilising a subset of statistical features representing structural information. [102] Instead of requiring the full reference image, RR-SSIM computes features such as mean luminance, contrast, and correlation coefficients over specific sub-bands or regions. These features are quantised and transmitted to the receiver, enabling a quality assessment that closely approximates the FR-SSIM while significantly reducing the amount of side information needed. [138]

Another notable RR model is the Generalised Reduced Reference (GRR) model, which is based on Natural Scene Statistics (NSS).[120] GRR models assume that natural images and videos exhibit certain statistical regularities, and distortions introduce deviations from these patterns. By transmitting key statistical parameters extracted from the reference video, the GRR model allows the receiver to detect and quantify deviations in the received video, thereby estimating the quality loss.

RR models often employ wavelet-based techniques to extract features that capture both spatial and temporal information. [45] For instance, features might be derived from the coefficients of a wavelet transform applied to the reference video frames. These coefficients represent different frequency components, and changes in their distributions can indicate various types of distortions, such as blurring or blocking artefacts [70] By focusing on salient features that are most sensitive to human perception, RR models can achieve a good balance between assessment accuracy and the overhead of side information transmission.

In practical applications, RR models are valuable for monitoring video quality in networked environments where bandwidth is constrained, but some control over the content delivery system is possible.[119] For example, in satellite or cable television broadcasting, the broadcaster can embed RR features within the broadcast stream. [142] The end-user devices can then use these features to assess the quality of the received signal and report back any significant degradation. This feedback can be instrumental in maintaining Quality of Service (QoS) by allowing network operators to detect and address transmission issues promptly.

RR models are also used in scalable video coding, where different layers of video quality are transmitted, and the receiver may only receive a subset of these layers due to bandwidth variations. [106] By transmitting RR features corresponding to different layers, the receiver can estimate the quality of the reconstructed video even when some layers are missing, facilitating adaptive streaming strategies that optimise user experience under varying network conditions.

Despite their advantages, RR models face challenges related to the selection and transmission of side information. The amount of side information must be carefully managed to avoid negating the bandwidth savings achieved by not transmitting the full reference video. Moreover, the features selected must be robust to typical transmission errors and sufficiently informative to enable accurate quality assessment. Designing RR models thus involves a trade-off between the granularity of quality estimation and the overhead of feature transmission.

Another limitation is that RR models may not capture all types of distortions effectively, especially those that affect higher-order statistics or are content-dependent. [11] For instance, certain temporal artefacts that arise in video compression might not be fully characterised by the transmitted features, leading to less accurate quality predictions. Additionally, the performance of RR models can be influenced by the choice of quantisation and encoding schemes used for the side information, which must balance precision against bandwidth constraints.

Research continues to enhance RR models by exploring advanced feature extraction methods and machine learning techniques. [78] For example, incorporating perceptual models that account for the Human Visual System (HVS) can improve the correlation between RR model

predictions and subjective quality assessments. Machine learning algorithms can be trained to map extracted features to quality scores more accurately, potentially adapting to different types of content and distortion patterns.

In conclusion, RR models offer a practical solution for video quality assessment in scenarios where full reference data is unavailable but limited side information can be utilised. They provide a compromise between the comprehensive analysis of FR models and the flexibility of NR models, making them suitable for a variety of applications in networked video services and broadcast systems. Ongoing advancements aim to improve their accuracy, efficiency, and applicability across diverse contexts, addressing the inherent challenges associated with feature selection and transmission overhead.

2.4.4 Comparison of NR, RR, and FR Models

The selection among No-Reference (NR), Reduced-Reference (RR), and Full-Reference (FR) models for video quality assessment is contingent upon the availability of the original content and the specific demands of the application domain. NR models are advantageous in situations where the reference video is inaccessible, as they estimate video quality solely based on the distorted content. [81] While this makes NR models flexible and widely applicable, they often suffer from lower accuracy due to the absence of a baseline for comparison, leading to potential inaccuracies, especially with complex or unfamiliar distortion types. [80]

RR models bridge the gap between NR and FR models by utilising partial information from the reference video. They extract a concise set of features or statistics from the original content, which are then transmitted to the receiver for comparison with the distorted video. [138] RR models offer improved accuracy over NR models while requiring less bandwidth than FR models. However, they still necessitate the transmission of side information, which may not be feasible in all scenarios, and selecting the most representative features without imposing significant overhead remains a challenge.

FR models provide the most precise and exhaustive assessment of video quality by directly comparing the distorted video with the full reference video. [137] The availability of the complete original content allows FR models to detect even subtle differences and accurately quantify degradation caused by compression, transmission errors, or processing artefacts. [108] This high level of accuracy is particularly advantageous in applications where exact quality measurement is critical.

In the context of cloud gaming, the benefits of FR models are especially pronounced. Cloud gaming involves rendering game content on remote servers and streaming the resulting video to players in real-time. [24] Service providers have access to both the original rendered frames and the transmitted video stream, making FR models highly suitable for assessing the quality of the streamed video. [46] Accurate quality assessment is essential in cloud gaming because any degradation in video quality can significantly impact the user experience, affecting gameplay responsiveness and visual fidelity. [56]

FR models enable cloud gaming platforms to monitor and optimise video streaming quality effectively. By detecting discrepancies between the original and streamed videos, service providers can identify issues such as compression artefacts, latency-induced distortions, or network-induced losses. [130] This information can be utilised to adjust encoding parameters, implement error correction strategies, or adapt streaming bitrates in real-time to maintain optimal quality.

Moreover, the precise measurements provided by FR models facilitate the fine-tuning of video codecs and streaming protocols specifically for cloud gaming applications. Given the interactive nature of gaming, where rapid visual changes and high frame rates are common, the ability of FR models to capture detailed temporal and spatial distortions is crucial. [30] This level of analysis helps ensure that the visual quality meets the stringent requirements of gamers, who often have high expectations for performance and visual clarity.

In conclusion, while NR and RR models offer benefits in terms of flexibility and reduced data requirements, FR models provide the most accurate and detailed assessment of video quality. Their advantages make them particularly suitable for cloud gaming video streamed quality analysis, where access to the original content is available, and precise quality measurement is necessary to deliver a high-quality user experience. The utilisation of FR models in cloud gaming ensures that service providers can maintain the visual fidelity and responsiveness that are critical for player satisfaction.

2.4.5 Peak Signal-to-Noise Ratio (PSNR)

The Peak Signal-to-Noise Ratio, commonly known as PSNR, stands as a venerable metric in digital image and video quality assessment. At its essence, PSNR quantifies the difference between a reference image or video frame (typically the original, uncompressed version) and a distorted version, often resulting from compression or transmission errors. [107]

Mathematically, PSNR is defined in terms of the Mean Squared Error (MSE) between the reference and the distorted image. MSE represents the average squared difference between corresponding pixels in the two images. [90] The formula for PSNR, given in decibels (dB), is:

$$\text{PSNR} = 10 \times \log_{10}\left(\frac{\text{MAX}^2}{\text{MSE}}\right) \tag{2.1}$$

Where:

- MAX is the maximum possible pixel value of the image. For an 8-bit greyscale image, this value is 255.
- MSE is the mean squared error between the reference and distorted images.

The logarithmic scale in the formula ensures that the PSNR value captures a wide range of error magnitudes. [71] Higher PSNR values indicate better quality, as the noise or error is lower than the peak signal.

While PSNR is revered for its simplicity and ease of computation, it is not without nuances. One of the primary criticisms of PSNR is its occasional misalignment with human visual perception. [55] For instance, two images with similar PSNR values might be perceived differently in terms of quality by a human observer. [26] This discrepancy arises because PSNR is a purely mathematical metric without considerations related to human visual system characteristics.

PSNR has been a staple for codec evaluation and development in video compression. Engineers and researchers often plot rate-distortion curves, where the bit rate (indicative of compression) is plotted against PSNR values. [111] Such curves provide insights into the efficiency of compression algorithms, revealing the trade-off between bit rate and perceived quality.

2.4.6 Structural Similarity Index (SSIM)

The Structural Similarity Index, or SSIM, emerged as a response to the limitations of traditional error-based metrics like PSNR. [98] Recognising that human perception of image quality is inherently linked to structural information, SSIM was designed to provide a more perceptually relevant measure of image similarity. [121] It captures luminance, contrast, and structure, three fundamental components that significantly influence our perception of visual quality. [148] The Structural Similarity Index (SSIM) for two windows, x and y, of an image is defined as:

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{2.2}$$

Where:

- μ_x and μ_y are the average of x and y respectively.
- σ_x^2 and σ_y^2 are the variances of x and y respectively.
- σ_{xy} is the covariance of x and y.
- C_1 and C_2 are constants to avoid instability when the denominators are close to zero.

The brilliance of SSIM lies in its ability to separately assess the luminance, contrast, and structural similarities and then combine them into a comprehensive measure. [93] [160] This multi-faceted approach ensures that even if two images are pixel-wise different, they might still be deemed similar if they share structural patterns, luminance, and contrast characteristics that align with human visual perception. [158]

In practical applications, SSIM has been found to be widespread in image and video compression evaluation, watermarking, and transmission error analysis. [28] Its perceptual relevance makes it particularly valuable for assessing the performance of codecs and compression algorithms, especially when the goal is to optimise for human viewership. [101]

However, like all metrics, SSIM has its challenges. While it offers a more perceptually aligned measure than PSNR, it is computationally more intensive. [77] Furthermore, while SSIM

captures many aspects of human visual perception, it does not encapsulate all nuances, leading researchers to develop variants and extensions to further align with human perception.

2.4.7 Video Quality Metric (VQM)

The Video Quality Metric, often abbreviated as VQM, is a testament to the continuous pursuit of more refined and perceptually relevant video quality measures. Born out of the need to capture a broader spectrum of distortions in video sequences, VQM offers a comprehensive assessment beyond traditional error-based metrics. VQM is distinct in its approach. [60] Instead of solely focusing on pixel-wise differences, it considers spatial, temporal, and chromatic distortions. This multi-dimensional assessment ensures that VQM captures various artefacts, from blurring and noise to ghosting and colour inaccuracies.

The computation of VQM involves several stages. Initially, both the reference and test videos undergo a series of pre-processing steps, including spatial and temporal alignment. This ensures that the videos are perfectly synchronised for subsequent analysis. Following this, perceptual models are applied to extract features from the videos. These features capture various aspects of video quality, including sharpness, motion, and colour fidelity. Once the features are extracted, they are combined using a calibration model to produce the final VQM score. This score typically ranges between 0 and 1, with 0 indicating no perceptible difference between the reference and test videos and 1 indicating maximal perceptual difference.

One of the standout attributes of VQM is its ability to operate in a "no-reference" mode. While many quality metrics require a pristine reference video for comparison, VQM can provide quality assessments even when a reference is unavailable. This capability is precious in real-world scenarios where the original, uncompressed video might not be accessible. In practical applications, VQM has found favour in a myriad of domains. Broadcasters use it to assess the quality of transmitted video content, ensuring viewers receive a high-quality viewing experience. Streaming platforms employ VQM to optimise compression settings, balancing bandwidth constraints with perceptual quality. Additionally, researchers leverage VQM to evaluate the performance of new video codecs and transmission protocols.

However, as with all metrics, VQM has its challenges. Its computation can be more intensive than straightforward metrics, making real-time assessments challenging in specific scenarios. [140] Moreover, while VQM captures a broad spectrum of distortions, no single metric can encapsulate all nuances of human visual perception.

2.5 Video Multimethod Assessment Fusion (VMAF)

The Video Multimethod Assessment Fusion, VMAF, emerged as a beacon in video quality metrics. Developed by Netflix, VMAF was conceived to address the evolving challenges of delivering high-quality streaming content to a diverse global audience with varying network conditions. VMAF distinguishes itself by fusing multiple elementary quality metrics into a singular score that aims to align closely with human visual perception. Instead of relying on a

single method to gauge quality, VMAF taps into the strengths of multiple algorithms, thereby offering a more holistic and robust assessment of video quality. [127] The computation of VMAF involves several intricate steps. Initially, both the reference and test videos are subjected to a series of quality metrics, each capturing different aspects of video quality. [100] These can include metrics that assess spatial distortions, temporal inconsistencies, and even ringing artefacts. Once these elementary scores are computed, they are combined using a machine-learning model trained on a vast dataset of human-rated video samples. This fusion of metrics, guided by human-rated scores, ensures that VMAF captures a wide array of perceptual artefacts. Moreover, the machine-learning backbone allows VMAF to be adaptive. As more human-rated data becomes available, the model can be refined and updated, ensuring that VMAF remains attuned to evolving perceptions of video quality.

In practical applications, VMAF has become a cornerstone for streaming platforms, especially Netflix. Given its genesis at Netflix, VMAF was tailored to assess the quality of compressed video content, making it particularly relevant for streaming scenarios. Engineers leverage VMAF scores to fine-tune encoding settings, ensuring viewers receive optimal video quality for their specific bandwidth conditions. [131] However, VMAF, like all metrics, is not an absolute measure. While it offers a comprehensive assessment of video quality, it is essential to interpret its scores in context. [41] A higher VMAF score indicates that the perceptual quality will be better, but the difference between scores is not always linearly related to perceptual differences.

Moreover, while VMAF captures many nuances of human visual perception, certain artefacts or scenarios require complementary metrics for a complete assessment. VMAF, with its unique fusion of multiple quality metrics and machine learning, offers many advantages that make it particularly appealing for video quality assessment. One of the standout attributes of VMAF is its alignment with human visual perception. By training the model on human-rated video samples, VMAF ensures that its scores resonate with how real viewers perceive video quality. [151] This perceptual relevance is paramount, especially when the end goal is to optimise content for human consumption. Furthermore, VMAF's machine-learning backbone allows it to be continually refined. As more human-rated data becomes available, the model can be updated, ensuring that VMAF remains attuned to evolving perceptions of video quality. This adaptability means that VMAF is not a static metric but one that can evolve with advancements in video technology and shifts in viewer preferences.

Another significant advantage is VMAF's versatility. While it was initially developed with streaming content in mind, its applicability extends to various video applications, from broadcasting to video conferencing. Its ability to provide meaningful scores across different video types and content makes it a universal tool for video quality assessment. [33] Choosing VMAF for experiments often stems from its robustness and perceptual relevance. In experimental contexts, where the objective is to draw meaningful conclusions about video quality, VMAF provides a reliable and perceptually aligned measure. Its scores offer insights that are both quantitatively rigorous and qualitatively meaningful, bridging the gap between numbers and human experience. Moreover, backing a significant streaming platform like Netflix adds to VMAF's credibility.

Since Netflix serves content to millions of viewers worldwide, its investment in developing and refining VMAF underscores its importance and reliability.

In the rapidly evolving digital media landscape, streaming video quality is a pivotal cornerstone. As cloud gaming gains traction, offering immersive experiences to players across the globe, the importance of delivering pristine video quality becomes paramount. Cloud gaming, inherently reliant on streamed video, demands high-quality content and consistent and lag-free delivery. [153] Any compromise in video quality can significantly impact the gaming experience, turning potential immersion into frustration. [159] Video quality metrics are indispensable to ensure optimal video quality. These metrics, from traditional PSNR to more advanced measures like VMAF, provide objective video quality assessments. They serve as compasses, guiding the continuous optimisation of video streams and ensuring that gamers receive the best possible visual experience.

Among the plethora of metrics available, VMAF emerges as a particularly compelling choice. Developed by Netflix, a leader in video streaming, VMAF offers a holistic assessment of video quality, fusing multiple elementary metrics through a machine-learning model trained on human-rated samples. [69] Its scores resonate closely with human perception, making it especially relevant for applications like cloud gaming, where the end viewer's experience is paramount. Given its perceptual relevance, adaptability, and robustness, we have chosen VMAF as our metric of choice. Its advantages position it as an invaluable tool in our quest to understand and optimise video quality for cloud gaming, ensuring that players receive an unparalleled gaming experience.

2.6 User Activity

2.6.1 Mouse

Mouse input in video games is as fundamental as keyboard interaction, providing precision and enhancing player control. The mouse facilitates accurate aiming in first-person shooter (FPS) games, efficient unit selection in real-time strategy (RTS) games, and smooth camera movement in 3D games, effectively converting physical movements into precise in-game actions. [143] This direct mapping enables intuitive interaction, making gameplay more engaging and immersive for players.

The versatility of mouse input extends beyond basic pointing and clicking. The mouse wheel provides additional functionalities, such as zooming or scrolling, while extra buttons can be customised for frequently used actions, such as reloading or activating abilities. The ability to customise mouse sensitivity, acceleration, and button assignments allows players to tailor the control settings to suit their preferences, thereby optimising the gaming experience and enhancing responsiveness.

Real-time responsiveness is critical for mouse input. Event-driven input methods are used to ensure that in-game reactions to player actions occur without delay, which is essential for fast-paced gaming genres. This responsiveness is instrumental in maintaining player control over their in-game avatar and environment, contributing to a more satisfying gameplay experience.

In addition to precision, mouse input provides valuable visual feedback to players. The cursor may change shape or colour based on context, offering visual cues that assist players in effectively interacting with the game environment. Such feedback helps players execute tasks accurately and maintain a strong connection with the virtual world. Furthermore, mouse input detection and processing involve sophisticated algorithms that guarantee smooth cursor movement, collision detection, and responsiveness, enhancing the overall quality of interaction.

The mouse is particularly valuable for navigation and camera control in gaming. In many 3D and strategy games, the mouse allows players to adjust their perspective by panning or tilting, enabling them to maintain an acute awareness of their surroundings. [39] These functions are essential for effective gameplay, as they allow players to navigate complex environments with efficiency and precision.

Customisation features also play a significant role in enhancing accessibility. Players can adjust mouse sensitivity, reassign buttons, and modify input behaviour according to their needs, ensuring that games remain inclusive and enjoyable for a wide range of players. [51] This adaptability accommodates the diverse abilities and preferences of the gaming community, ensuring that individuals with different levels of skill and physical capability can engage fully with the game.

In summary, mouse input serves as a crucial interface in gaming, offering accuracy, adaptability, and responsiveness. By establishing a direct connection between players and the virtual environment, mouse input enhances the immersive quality of video games, rendering them more interactive and accessible. The combination of customisation and precision enriches the overall player experience, making virtual worlds feel more dynamic, engaging, and responsive to player actions.

2.6.2 Keyboard

Keyboard input is fundamental to player engagement in video games, enabling players to execute a wide range of actions within the virtual environment. The keyboard provides extensive control and functionality, supporting both basic operations—such as character movement—and complex action sequences, which are essential for strategic manoeuvres or solving intricate puzzles.

Keyboard interaction adds depth and immediacy to gameplay. Each key or key combination is mapped to a specific in-game action, ensuring responsive interaction that closely aligns with the player's intentions. Context-sensitive input further enriches gameplay by allowing a single key to serve multiple functions based on the situation, such as toggling between walking and sprinting. [38] This versatility supports more complex game mechanics and allows for deeper narrative integration, thereby enhancing player immersion.

Modifier keys expand the functional capabilities of the keyboard by altering the behaviour of other keys. This feature is crucial for games that require numerous inputs, such as strategy or simulation games, as it enables quick access to complex commands. Additionally, input sequences and timing-sensitive actions introduce a rhythmic element to the gameplay, challenging

players' reflexes and adding a satisfying layer of difficulty, particularly in genres that demand precision.

Game developers employ techniques such as polling and event-driven input to capture keyboard interactions effectively. Event-driven input is preferred for its responsiveness, as it allows the game to react promptly to player actions. Techniques such as buffering and debouncing are also utilised to address the limitations of mechanical keyboards—buffering ensures that rapid input sequences are executed in the correct order, while debouncing prevents unintended repeated actions that may disrupt gameplay.

Customisation and remapping capabilities have become crucial elements in enhancing accessibility and personalisation within video games. By allowing players to customise control layouts, game developers cater to diverse needs, including those of players with disabilities. This flexibility ensures that players can interact with the game comfortably and effectively, reflecting the industry's commitment to inclusivity and accessibility. [125]

In conclusion, keyboard input in video games represents the intersection of technical precision and the creative challenges inherent in game design. By leveraging responsive interactions, context-sensitive functionality, and customisation options, developers create immersive and accessible experiences that resonate deeply with players. These interactions allow players to shape outcomes and explore virtual worlds through actions performed using a standard keyboard, thereby contributing to engaging and meaningful gameplay experiences.

2.6.3 Controller

Game controller input is not merely a component of console gaming but forms the foundation of the gaming experience, with increasing relevance in PC gaming as well. Game controllers are not simply collections of buttons, triggers, and joysticks; they are intricately designed tools that provide an intuitive and precise interface for interacting with games, thereby enhancing player immersion. [96] Controllers accommodate a wide range of actions, from basic movements to complex sequences, which makes them versatile tools suitable for a variety of game genres.

The principles of game controller input are centred on ergonomic design, with the goal of optimising both comfort and efficiency. Each button, trigger, joystick, and directional pad (D-pad) is strategically placed to allow quick and comfortable access. This layout facilitates intuitive interactions, such as using joysticks for character movement and buttons for executing specific actions. Tactile feedback from button presses, combined with the smooth analogue input from joysticks, contributes to enhanced player engagement and allows for precise manoeuvres.

Analogue input, such as that provided by joysticks and triggers, is a significant advantage of game controllers. Unlike the binary nature of keyboard input, analogue input detects a range of motion or pressure, enabling finer control over in-game actions, such as adjusting vehicle speed or precisely aiming at targets. This nuanced input capability results in smoother, more realistic gameplay, enhancing the immersive quality of the gaming experience.

Controllers can be connected through direct or wireless connections, ensuring low latency and reliable input detection. [42] Modern gaming platforms and engines feature comprehensive

application programming interfaces (APIs) that simplify the integration of controller input, enabling developers to assign in-game actions to specific controls with ease. These APIs also support haptic feedback, such as vibration, which enhances player immersion by providing tactile cues that align with in-game events, such as explosions or collisions.

Advanced features, such as motion sensors and touchpads, further extend the interactivity offered by game controllers. Motion controls allow players to manipulate game elements by physically moving the controller, creating a more immersive experience—such as steering a car in a racing game. Touchpads add precision, combining the functionality of traditional buttons with mouse-like pointing capabilities, allowing for innovative gameplay mechanics and enhanced navigation methods.

Customisation and accessibility are fundamental considerations in both controller design and game integration. Many games and platforms offer players the ability to remap buttons, allowing them to tailor controls to their preferences. [86] This customisation is especially beneficial for enhancing accessibility for players with diverse abilities. Sensitivity adjustments for analogue controls, as well as optional vibration feedback, help cater to individual preferences, ensuring that players with varying physical capabilities can enjoy a comfortable gaming experience.

Game controllers are not merely devices for interaction; they serve as tools for inclusivity, enabling diverse players to fully engage with games. The combination of ergonomic design, tactile feedback, and analogue control empowers developers to create experiences that are both immersive and accessible. As gaming technology continues to evolve, game controllers also continue to adapt to meet the changing needs of player interaction. This adaptability ensures that controllers remain integral to the gaming experience, bridging the digital and physical worlds while inspiring excitement for future innovations in gaming technology.

2.7 Game Genres

2.7.1 First Person Shooter (FPS)

First-Person Shooter (FPS) games are a genre of video games in which players engage in combat from a first-person perspective. In FPS games, players assume the role of the protagonist and utilise various firearms to combat enemies. The core mechanics of FPS games centre around rapid combat, precise shooting, and tactical movement, necessitating quick reflexes, accurate aiming, and strategic thinking to succeed.

FPS games are renowned for their immersive perspective, diverse weapon options, challenging enemy artificial intelligence (AI), sophisticated multiplayer modes, and intricately designed levels. Many FPS games also incorporate narratives and character progression, adding depth to the gameplay experience. These features contribute to an engaging experience that tests both reflexes and decision-making abilities, appealing to a wide audience. Understanding user interactions in FPS games involves examining how players use input devices—such as keyboards, mice, or controllers—to interact with the game environment. These interactions extend beyond basic navigation and item selection, playing a crucial role in core mechanics such as aiming,

shooting, and tactical manoeuvring. The relationship between input devices and game content reveals important aspects of FPS game design, particularly regarding player immersion and engagement.

Keyboards are primarily employed for movement, executing actions, and navigating menus in FPS games. The standard WASD key layout is used for character movement, allowing players to move forward, backwards, left, and right with precision. Additional keys facilitate actions such as jumping, crouching, reloading, and switching weapons. The haptic feedback and layout of keys enable players to perform complex in-game actions efficiently. This level of control allows designers to create intricate levels that require skilful navigation and strategic movement. [32] For instance, multiplayer maps may include narrow corridors, open areas, and verticality, encouraging players to utilise movement strategically—whether through stealthy approaches or rapid escapes.

The mouse is the most critical input device for FPS games due to its role in aiming and camera control. The high sensitivity and precision provided by the mouse are crucial for achieving accurate targeting, which is essential in games where shooting precision determines success. Mouse buttons are used for primary actions, such as firing, as well as secondary actions, like using weapon scopes. The precision afforded by the mouse significantly influences the design of enemy behaviours and shooting mechanics, allowing developers to create enemies with specific vulnerabilities that reward accurate targeting. Additionally, the design of weapons, each with distinct handling characteristics, is shaped by mouse precision. The agility provided by the mouse supports the fast-paced gameplay that defines FPS games, requiring quick reflexes and precise aiming to achieve optimal performance.

Game controllers offer a different experience in FPS games by combining joystick movement with buttons for various actions. Typically, the left joystick controls character movement, while the right joystick is used for aiming and camera control. Triggers and bumpers facilitate shooting and other actions, such as throwing grenades or performing melee attacks. Due to the lower precision of joysticks compared to mice, game controllers often necessitate the implementation of aim-assist features, which help compensate for the inherent limitations of analogue input. This affects both the pacing and difficulty of the game, as developers must balance the experience to ensure fairness and enjoyment for players using either controllers or mice. Additionally, the ergonomic design of game controllers allows players to comfortably engage in extended gaming sessions, influencing the overall pacing and structure of the game.

The integration of keyboards, mice, and controllers in FPS games requires a careful balance of game mechanics and content design. Developers must consider the advantages and limitations of each input method to create an accessible and satisfying experience. This involves refining control schemes, offering customisation options, and incorporating features such as aim assist and sensitivity adjustments. The attributes of these input devices—including latency, precision, and ergonomic design—play a significant role in determining FPS game content. High-stakes competitive FPS games rely on low input latency and high precision to ensure fair, skill-based

gameplay. Conversely, more casual or narrative-driven FPS games prioritise accessibility and immersion to appeal to a broader audience.

In conclusion, user interaction through keyboards, mice, or controllers significantly influences FPS game design and content. [29] Developers must navigate the complexities of each input method to create game worlds, mechanics, and narratives that cater to the diverse skills and preferences of FPS players, ensuring both accessibility and engagement. Achieving this balance requires a harmonious integration of technical precision and creative design, resulting in an immersive FPS experience that captivates players and delivers the excitement of fast-paced combat.

2.7.2 Real Time Strategy (RTS)

Real-time strategy (RTS) games are a genre of video games in which players gather resources, construct bases, create units, and control armies in real-time. Unlike turn-based strategy games, RTS games require players to make decisions and act swiftly, responding to changing conditions on the battlefield. Players typically view the game from a top-down perspective, using keyboards and mice as primary input devices to manage gameplay.

RTS games are characterised by their focus on resource management, base construction, unit production, and real-time combat. Players are tasked with gathering resources, such as minerals and oil, to support troop production and facilitate technological advancements while simultaneously constructing bases for production and upgrades. The core gameplay involves managing various military units with unique roles, requiring tactical adaptations to dynamic scenarios. RTS games frequently feature multiple game modes, such as campaigns and skirmishes, and offer extensive content, including different factions, technology trees, and diverse maps to extend playability.

Analysing user interaction in RTS games provides insight into how players use input devices—such as keyboards, mice, or controllers—to manage resources and control units and engage in strategic planning within expansive game environments. RTS gameplay demands strategic decision-making, rapid responses, and effective multitasking. The attributes of input devices and their correlation with game content highlight the balance between game design and player engagement, ultimately contributing to a cohesive RTS experience.

Keyboards are fundamental in RTS games, serving as the primary input device for executing commands, managing shortcuts, and navigating menus efficiently. Players utilise keyboards to issue group commands, trigger abilities, and establish control groups by assigning units to specific numbered keys. This enables the rapid selection of multiple units, enhancing the player's ability to execute complex strategies with both speed and precision. Effective keyboard use is essential for managing large armies and adapting to the fluid nature of RTS battles.

The mouse also plays a critical role in RTS games, allowing players to select units or structures, navigate the game map, and issue commands for movement or attacks. The precision provided by the mouse is crucial for controlling unit placement and executing strategic manoeuvres. Common mouse interactions include dragging to select multiple units, right-clicking to set

movement destinations or attack targets, and interacting with the user interface to build structures or initiate unit production. Ensuring smooth mouse input is vital for maintaining responsiveness in high-pressure situations, such as large-scale battles or resource management tasks.

Game controllers, though less common in RTS games due to the precision required, are gaining popularity as console-based RTS games become more widespread. Controllers are employed to navigate the battlefield, select units, and trigger abilities through button combinations and joystick movements. To accommodate the limited number of buttons available on controllers, designers often develop innovative control schemes to adapt RTS mechanics, ensuring that players can effectively manage their armies and execute complex commands without sacrificing gameplay quality.

Integrating input from keyboards, mice, and controllers in RTS games presents several challenges. Developers must design intuitive control schemes that account for the inherent complexity of the RTS genre while ensuring accessibility. Effective management of multiple units and actions on expansive maps requires an efficient and accessible interface. Customisable hotkeys, intelligent user interface (UI) design, and features such as group selection and command queuing are crucial for enhancing the strategic gameplay experience, allowing players to control large armies and coordinate complex strategies seamlessly.

The choice of input device significantly impacts RTS game design and content. Understanding the strengths and weaknesses of each input method allows developers to leverage their unique advantages to create engaging and immersive RTS experiences. Keyboards and mice are typically better suited for RTS games that require precise unit management and rapid command execution, while game controllers are more appropriate for those that prioritise accessibility and broader appeal. Effective design must balance technological capabilities with creative gameplay elements, ensuring an enjoyable experience for all players.

Player interaction through keyboards, mice, or controllers has a profound influence on RTS game design. By understanding and leveraging the strengths of each input device, developers can craft immersive and challenging RTS games that require players to strategise, efficiently manage resources, and execute complex tactical plans. [155] Striking a balance between technical innovation and creative game design is essential for developing compelling RTS games that cater to a diverse audience, delivering a strategic experience that is both accessible and engaging. Successful RTS games must seamlessly integrate complex control systems, rich content, and adaptive gameplay, providing players with a dynamic and rewarding experience that holds their attention and challenges their strategic thinking.

2.7.3 Role Playing Games (RPG)

Role-playing games (RPGs) represent a genre of video games in which players assume character roles, embark on quests, and immerse themselves in complex storylines within a virtual world. These games often feature intricate narratives, character progression, and gameplay systems that provide immersive, interactive adventures. [18] RPGs are distinguished by their

emphasis on storytelling, combat mechanics, and character development, where players enhance their characters by gaining experience, acquiring new skills, and upgrading equipment.

In RPGs, players either create or choose characters, navigating the virtual environment from the perspective of their avatars. Story-driven quests are accomplished through exploration, dialogue, and decision-making. [97] Combat systems can be either turn-based or real-time, requiring strategic thinking and skilful execution to overcome opponents. Character development is fundamental to RPGs, allowing players to improve skills, gain experience, and unlock abilities as they progress.

Understanding player interaction in RPGs provides insight into how players use input devices—such as keyboards, mice, or game controllers—to navigate, engage in combat, and interact with the game environment. RPG gameplay demands strategic decision-making, exploration, and adaptive problem-solving. The interaction between input devices and game content highlights how game design influences player immersion, resulting in an engaging RPG experience.

Keyboards serve as a primary input mechanism for character control and interaction within the game environment. The arrow keys or WASD layout is typically used for character movement, while other keys facilitate actions such as interacting with non-player characters (NPCs), accessing menus, and using skills. The arrangement and functionality of the keyboard significantly impact gameplay quality, as effective control of characters requires intuitive and accessible shortcut configurations. Designers must balance movement speed, action responsiveness, and difficulty progression to ensure a smooth and satisfying experience for players.

The mouse also plays a significant role, particularly in click-based RPGs. Players utilise the mouse to select targets, interact with the game world, and manage items and skills. The precision of the mouse enables intuitive control and facilitates the swift execution of actions. During combat, the mouse is used for targeting, skill activation, and inventory management. This precision necessitates careful design of combat mechanics and enemy behaviours to ensure balanced difficulty and responsive interaction. [88] Designers must ensure that mouse-based interactions are fluid and allow for natural and engaging gameplay.

Game controllers are popular for console-based RPGs, providing an ergonomic and user-friendly interface that is particularly suitable for prolonged gameplay sessions. Players use analogue sticks for movement, while buttons are employed for actions such as attacking, interacting with NPCs, and navigating menus. Controllers have a significant influence on RPG design, requiring optimised character control, camera angles, and interface navigation to ensure an intuitive and comfortable experience. Designers must adjust game difficulty and pacing to align with the attributes of controllers, providing a seamless experience compared to the more precision-focused control offered by keyboards and mice.

The integration of keyboards, mice, and controllers in RPGs requires careful planning by developers to optimise both game design and technical performance. Interfaces, control schemes, and difficulty progression must be tailored to the characteristics of the input devices to ensure a positive user experience. Key input device attributes—including response time, sensitivity, and

accuracy—must align with gameplay requirements, providing players with effective control and appropriate customisation options.

The choice of input device fundamentally influences how players interact with RPGs, impacting both the design and content of these games. Keyboards and mice are ideal for RPGs requiring high levels of precision, whereas controllers provide comfort and immersion, particularly during extended play. Developers must consider the strengths and limitations of each input device when designing elements such as pacing, difficulty, and task execution to ensure that the experience is enjoyable and accessible to all players.

Player interaction through keyboards, mice, or controllers significantly affects RPG game design. Understanding and leveraging the unique strengths of each input device allows developers to create immersive RPG experiences that engage players in exploration, combat, and rich storytelling. Ultimately, successful RPGs are those that encourage players to fully immerse themselves in the virtual world, allowing them to experience a dynamic journey through complex narratives and character development. By tailoring gameplay elements to the characteristics of different input devices, developers can create captivating RPG experiences that resonate with a diverse player base.

2.7.4 Action Games

Action games represent a genre of fast-paced video games in which players navigate characters through precise movements and rapid reactions to overcome various obstacles and adversaries. These games feature intense gameplay, sophisticated visual effects, and intricate control schemes, testing players' reaction times, hand-eye coordination, and strategic thinking. [43] The primary objectives typically include progressing through multiple levels, defeating adversaries, and completing missions to achieve high scores or unlock additional content.

Action games are characterised by their high-speed gameplay, which demands precise control, challenging boss battles, and diverse content. [95] Players must skilfully manage character movements, jumps, attacks, and dodges to tackle a variety of foes and obstacles. Levels are often comprised of unique environments, enemies, and challenges that require players to employ different strategies in order to advance. Boss battles are a prominent aspect of action games, necessitating players to understand attack patterns, carefully time their actions, and utilise special abilities to emerge victorious. Action games also offer various game modes and rewards that contribute to enhanced player enjoyment and replayability.

Understanding player engagement in action games requires examining how players utilise input devices such as keyboards, mice, or game controllers to navigate characters, perform actions, and overcome obstacles. These games demand fast reflexes, precise control, and quick decision-making. The connection between input devices and game content highlights how control design affects player engagement, leading to a seamless and immersive action gameplay experience.

On PC platforms, keyboards serve as the primary input device for executing actions in action games. Arrow keys or the WASD layout are commonly used to control character movement,

while additional keys facilitate actions such as jumping, attacking, or dodging. The flexibility of the keyboard significantly impacts the accuracy and speed of player movements. Game designers must carefully plan features to ensure that players can perform actions efficiently while minimising issues such as key conflicts or finger fatigue. Level design must consider keyboard capabilities, balancing difficulty and pacing to create an enjoyable yet challenging experience.

Mice also play a critical role in specific action game genres, such as third-person shooters and action role-playing games (RPGs). Players utilise the mouse to control the camera, aim, shoot, and interact with the game environment. The precision of the mouse allows for accurate targeting and swift execution of actions, and designers must ensure that camera and aiming controls are natural, optimising sensitivity for a seamless experience. Interactive elements of the game must be designed to accommodate mouse input, allowing players to engage intuitively and accurately.

Game controllers are the primary input device for action games on console platforms. Controllers provide an ergonomic means of gameplay, particularly during prolonged gaming sessions, and are considered more user-friendly compared to keyboards and mice. Players use analogue sticks to control character movement and camera angles, while buttons are assigned to actions such as jumping, attacking, and dodging. Vibration feedback and triggers are employed to enhance immersion, adding a layer of realism to the gameplay. Controller design influences game content, affecting control schemes, level design, and difficulty balance. Designers must optimise controls and camera angles to ensure an intuitive experience, while level design must take into account controller capabilities, ensuring that jumps, attacks and other interactions are challenging but not overly complex.

Integrating keyboards, mice, and controllers in action games requires meticulous planning and optimisation by developers to achieve seamless gameplay. Game mechanics, level design, and difficulty must be tailored to exploit the strengths of each input device. Parameters such as response time, sensitivity, and ergonomics must align with gameplay requirements to provide effective control and customisation options for players.

Different input devices are suited to different types of action games. Keyboards and mice are ideal for games that require precise aiming and rapid input, whereas controllers are more appropriate for action games that emphasise movement and rhythm. Developers must adjust level design, enemy behaviour, and feedback systems according to input device features to create an enjoyable and accessible experience for all players.

Player interaction through keyboards, mice, or controllers significantly influences the design of action games. Understanding the strengths and limitations of each input device allows developers to create compelling experiences that showcase players' skills and reflexes. Successful action games require a careful balance of responsive controls, well-designed levels, and audiovisual feedback to deliver enjoyable and challenging experiences. By mastering the complexities of player expectations and input device capabilities, developers can create action games that engage and captivate players, leaving a lasting impression.

2.7.5 Music Games

Music games are a genre of interactive entertainment where players synchronise movements with visual cues and music, aiming to execute notes or actions accurately in rhythm with the soundtrack. The objective is to synchronise with the background music, hit notes precisely, and accumulate scores. Music games often feature popular, classical, or original songs with various difficulty levels to cater to players of different expertise. Music games are characterised by rhythm, coordination, reaction speed, and melodic perception. Players must respond to visual cues by hitting keys or performing actions in time with the music. Games often use scrolling notes or visual prompts to convey timing, requiring players to synchronise movements with the rhythm. [22] Scoring systems assess player precision, motivating them to improve and tackle higher difficulties. Many music games also incorporate multiplayer modes for social play, enhancing player engagement.

Examining user interaction in music games offers insights into how players use keyboards, mice, controllers, or specialised peripherals like dance mats or electronic drums to match music rhythms and visual cues. Music games require strong rhythmic perception, hand-eye coordination, and fast response times. The characteristics of input devices and their relationship with game content demonstrate a close connection between game design and player engagement, resulting in immersive music game experiences.

The keyboard is a common input device in music games, requiring players to press keys in time with on-screen notes. Typical setups involve directional or letter keys (e.g., WASD or HJKL), which correspond to specific notes or actions. The haptic feedback and key arrangement enable precise and quick execution. The design of rhythm and difficulty in music games often hinges on keyboard input, allowing developers to create challenging sequences aligned with song melodies. Difficulty is varied by increasing note density, complexity, and speed, challenging players' reaction times and dexterity.

Game controllers, especially those designed specifically for music games, play a vital role. Controllers often mimic musical instruments like guitars, drums, or pianos. Players must press corresponding buttons or perform actions (e.g., strumming or drumming) in response to visual cues. Specialised controllers significantly impact gameplay, enabling developers to design unique mechanics that offer immersive experiences. For instance, guitar controllers feature coloured buttons and a strum bar, requiring players to synchronise presses with the music. Drum controllers use pads and pedals for accurate response to visual cues, enhancing player immersion.

Specialised peripherals like dance mats, electronic drum kits, and DJ controllers are also common in music games. These devices are specifically designed to enhance immersion. Dance mats allow players to perform challenges using their feet, while electronic drum kits provide realistic drumming experiences. The use of specialised peripherals influences game content, allowing developers to create levels that fully utilise these devices for an authentic experience. Dance games, for example, can use varied step sequences to emulate different dance styles, adding to player immersion. Such peripherals also support highly interactive multiplayer modes, like dance battles or cooperative band play.

Incorporating keyboards, controllers, and specialised peripherals into music games requires careful planning by developers to align game design with the capabilities of each input method. The game interface, level design, and scoring system must correspond to the attributes of the input devices and player expectations. Developers need to ensure that response time, sensitivity, and accuracy meet game requirements, providing a seamless experience. When designing music games, developers must analyse the strengths and limitations of different input devices. Keyboards are well-suited for games requiring rapid, precise input, while dance mats are ideal for games focused on physical movement and rhythm. Developers need to optimise game difficulty, rhythm patterns, and visual feedback according to input characteristics to ensure a rewarding experience for players. Ultimately, player interaction in music games significantly influences game design and content. Developers can create engaging and challenging experiences by leveraging the advantages of keyboards, controllers, and specialised peripherals. This allows players to fully immerse themselves in rhythm and music. Successful music games strike a balance between technological innovation, game design, and player engagement, offering an exciting and interactive music experience. To create music games that leave a lasting impact, developers must deeply understand music game dynamics, player expectations, and the specific strengths of different input devices.

2.7.6 Card Games

Card games are a genre focused on collecting and strategically using cards to engage in combat or achieve objectives. Players gather, combine, and deploy cards with diverse functionalities to compete against opponents. [144] Card games often feature a variety of cards, deck-building strategies, and turn-based combat. They assess players' ability to think strategically, manage resources, and adapt to changing circumstances, with goals centred on overpowering opponents, enhancing cards, and employing tactical manoeuvres to succeed.

Card games include features like card collection, deck building, turn-based combat, and strategic decision-making. Players acquire new cards to strengthen their collection and build effective decks based on card traits, costs, and effects. Combat is typically turn-based, requiring players to manage cards, allocate resources, and make tactical choices to defeat opponents. Strategic decision-making is a key aspect, as players must adapt to their opponent's strategy and manage resources wisely while leveraging card synergies to gain an advantage. Games also offer various card types, rarity levels, and acquisition methods to cater to players' progression and collection needs.

Examining user interaction in card games offers insights into how players use input devices like mice or touchscreens to select, use, and control cards. Card games require advanced strategic thinking, decision-making, and resource management. The features of input devices and their relationship with game content show a strong connection between game design and player engagement, resulting in immersive card game experiences.

The mouse is the primary input device for executing card actions and navigating interfaces in card games on PCs and mobile platforms. Players use the mouse to select cards, drag them, assign

targets, and confirm actions. The mouse's precision allows for fast, accurate card manipulation. The relationship between mouse input and card game content is evident in interface design and interaction processes. Game designers must arrange cards, resources, and target information in a way that allows easy navigation. The visual feedback, animations, and prompts must be designed to align with mouse operations, ensuring a seamless experience for players.

Touchscreens are a key interaction method for mobile card games, allowing players to tap, drag, and release cards directly on the screen. Touchscreens offer an intuitive way to perform actions, making strategic decisions and card manipulation straightforward. The influence of touchscreen input on card game content is seen in interface layout, card size, and interaction methods. Designers must optimise the spacing and arrangement of cards to ensure precision, while interaction design must consider touch gestures and habits, offering suitable clickable regions and gesture support. Visual feedback and animations should be tailored to match the responsiveness of touch interaction, ensuring an intuitive experience.

While game controllers are less common in card games, some console-based card games do support them. Players use analogue sticks or directional keys to select cards and buttons to confirm actions or switch interfaces. Controllers provide an ergonomic and effective way to play, especially during extended sessions. The design of game content must accommodate controller input, ensuring that card selection and gameplay flow naturally with the analogue stick or directional keys. Prompts and feedback must align with button layouts for clarity while pacing and animations must account for controller capabilities to maintain a smooth experience.

The integration of mice, touchscreens, and controllers into card games requires careful optimisation by developers to ensure seamless gameplay. The game's interface, card layout, and interaction methods must align with input device characteristics, providing a user-friendly experience. Response speed, sensitivity, and accuracy of each input device should meet game requirements, supporting suitable control options and adaptable features.

In designing card games, developers must analyse the strengths and weaknesses of each input method and how they apply to different scenarios. Mice and touchscreens are ideal for games requiring rapid, precise selection, while controllers may be better suited for games focused on strategic depth and longer play sessions. Game elements such as card effects, animations, and interactive feedback should be adjusted based on input device features to ensure an enjoyable and strategic experience for players.

Ultimately, player interaction in card games significantly influences game design and content. Developers can create strategic and immersive card experiences by leveraging the unique strengths of mice, touchscreens, and controllers. This enables players to fully exercise their decision-making skills and strategic thinking. To develop successful card games, developers must balance card design, pacing, and ease of operation, ensuring a tactically satisfying experience. A deep understanding of card game mechanics, player expectations, and input device attributes is crucial for creating card games that captivate players and leave a lasting impact.

2.8 Summary

Through the above content, we can understand that in today's cloud gaming services, reducing service provider costs and providing a better quality experience are urgent problems that need to be solved. The core content of cloud gaming services is how to optimise streaming video quality while dynamically adjusting video streaming parameters based on video quality. Through a large amount of literature analysis on related video quality parameters and an introduction to user interaction methods for different game types, we found that in current research, there is no video quality parameter model based on user operations. For cloud gaming services, especially the Quality of Experience of cloud gaming, user operations often have a critical impact on game content and the video quality parameters that change based on the content. Therefore, in the following chapters, we collect and analyse UAI and VMAF through methods such as programs and model UAI and VMAF through different methods.

Chapter 3

Methodology

3.1 Overview

In the previous chapter, we conducted a detailed analysis of the development and current issues faced by cloud gaming services, as well as the core streaming video technology and the video quality parameters required to improve the Quality of Experience. By analysing the different characteristics of user input in various game types, we determine the main research direction of this study, which is to model UAI and VMAF and predict VMAF based on UAI using the model. Therefore, in this chapter, we first create a program to collect UAI and VMAF data and collect a large amount of data based on corresponding parameters and different types of games. At the same time, according to the characteristics of the UAI and VMAF data, we propose two different modelling schemes, namely time series analysis and the NARX model. In order to better analyse the modelling results, we also provide a detailed introduction and comparison of the two different methods in this chapter. Moreover, we offer a detailed explanation of the two methods based on the characteristics of cloud gaming streaming video.

3.2 User Activity Index

As mentioned previously, analysing user behaviour data in cloud gaming services is a crucial issue. In Chapter 2, we elaborated on the distinctive characteristics of various game contents and the operational principles and data collection methods of multiple gaming controllers. Integrating these factors, we developed a metric for measuring user behaviour in cloud gaming services, known as the User Activity Index (UAI). The specific methods of data collection comprise the following components, as shown in the flowchart below Fig 3.1:

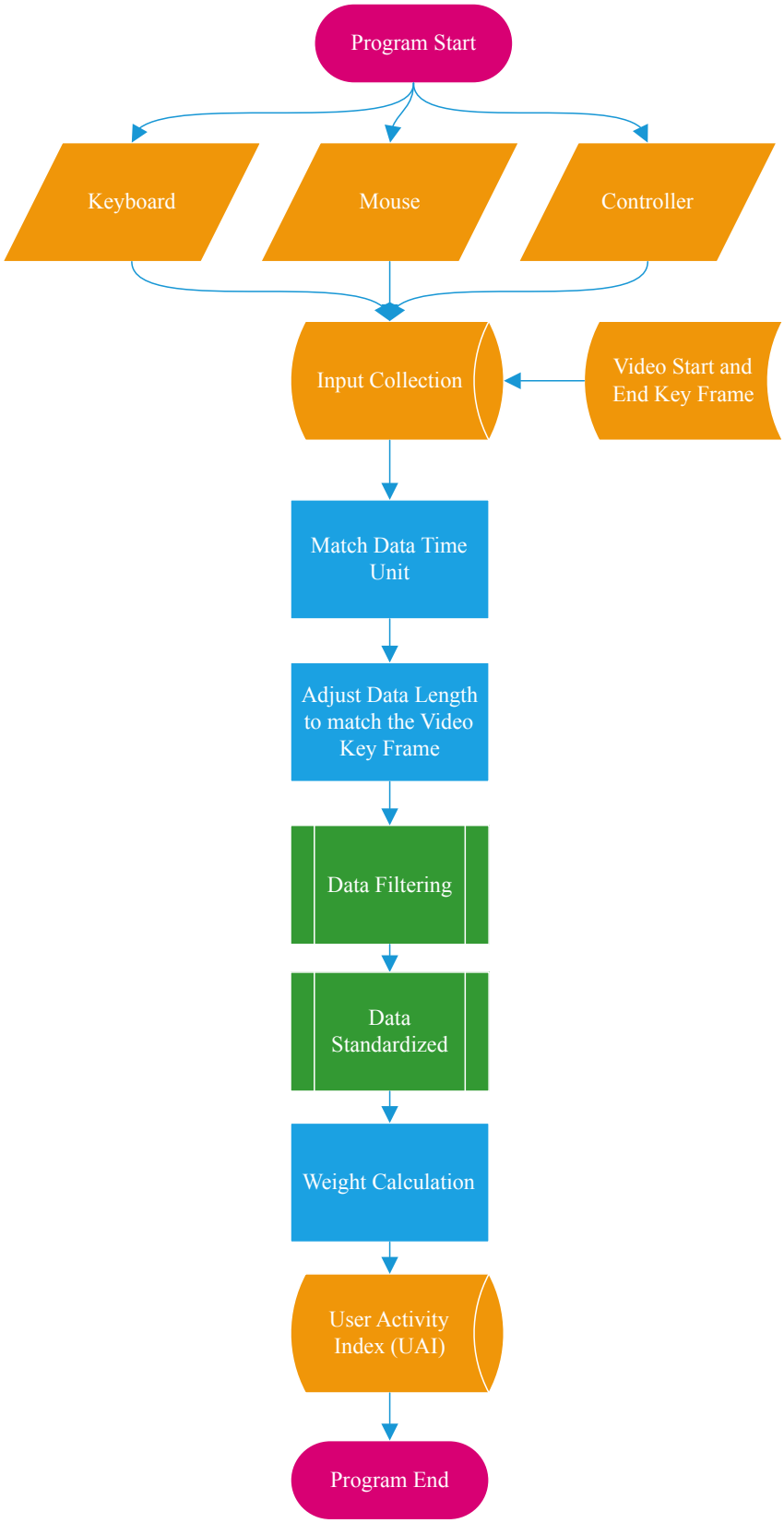


Fig. 3.1 User Activity Index Program Workflow

3.2.1 Keyboard Collection

For different types of games, the actions inputted through the keyboard are often highly consistent. A focal point in analysing user behaviour data is minimising the latency during data

collection. In our experiments, user inputs are captured directly via API interfaces provided by the Windows platform. By setting the operating system's task priority to prioritise the data collection program above the simulated game environment, we ensure a more stable recording of all keyboard actions and effectively ascertain a consistent delay across the collected data. Various methods have been employed to further mitigate the influence of external factors on the experiments.

3.2.2 Mouse Collection

In PC gaming platforms, mouse operation data is among the most complex. Different game genres exhibit vast disparities in content, leading to significant fluctuations in video complexity and mouse operation data. For instance, in card games, rapid mouse movements are generally unnecessary, and mouse data tends to show long-term stable patterns as these games are almost entirely reliant on mouse operations. Conversely, in Real-Time Strategy (RTS) games, where users must quickly switch displayed content, frequent and rapid mouse movements are necessary, showing a cyclic pattern of quick changes. For various situations, we weigh mouse operation data according to different game content variations, unifying the trends in cloud game streaming video content reflected by mouse operations. We divide the game screen into many areas of the same size based on resolution and take the maximum mouse travel distance under the current screen resolution. Subsequently, the distance between mouse coordinates across two frames is calculated to determine the mouse movement distance. Depending on the game and screen resolution, we standardise these distances on a scale from 1 to 10, where the farthest movement scores a ten and minimal movements score a 1. Furthermore, the data from the previous step is further weighted based on different game contents; for instance, in RTS games where video content and mouse operations change rapidly, we amplify the impact of mouse data in RTS games by a factor of 2. In card games, where mouse operations are more concentrated and regular with additional click data, we focus on calculating mouse click data and reduce the weighting of mouse movement data by a factor of 0.2 to balance the relationship between game video content and mouse data.

3.2.3 Controller Collection

In console platforms and native console games, collecting user data from controllers is paramount. As the primary devices designed for game interaction, controller data often better reflects changes in game operations. Similar to keyboard data, we prioritise the collection of controller button data using system-level APIs to effectively reduce latency. We weigh joystick data similarly to keyboard data. However, unlike keyboard data, joystick data is not calculated by movement distance but by the angle of joystick deflection. We record the maximum deflection angle of the joystick in different games and divide the joystick deflection angle in the current game by this maximum, converting this data into a value between 0 and 1 to reflect the variations in user behaviour data when using controllers.

By integrating the data collected from these diverse inputs, we standardise the quantification of data gathered from different control devices across various games. Additionally, the number of actions per frame varies significantly across different games, such as in action games where players constantly need to perform rapid actions, unlike in card games. To better reflect the discrepancies brought by different game contents, we define the basic time unit as one second and, based on different game frame rates, we calculate the total value of the three types of operation data within one second, thus illustrating the trend of game user behaviour data over time. Ultimately, by combining these various weighting and statistical methods, we can derive new user behaviour operation data, which is defined as UAI, which we will further analyse in relation to VMAF in the following sections.

3.3 VMAF Capture

Just like with user behaviour data, the collection of VMAF must also be adapted to align with the specific requirements of different game content. Given that Netflix developed the method for calculating VMAF, we are unable to modify the core calculation algorithm itself. Therefore, our focus shifts towards optimising the methods of VMAF data collection and the capture of original video content to accurately reflect the impact of game content on VMAF values. The VMAF capture program workflows as shown in Fig 3.2.

Firstly, akin to our approach with user behaviour data, we systematically gather VMAF data for each video frame according to the frame rate of the game content. This data is then aggregated and averaged to compute the average VMAF per second for the game video content. This methodology is particularly effective in illustrating the dynamic trend of quality changes associated with current game content. For genres such as action games, which typically experience more drastic visual changes, we refine our approach by subdividing the data according to the specific frame rate, allowing us to capture and reflect the more pronounced fluctuations in video quality.

Furthermore, an additional critical factor influencing VMAF variation is the methodology employed in original video capture. To replicate the pristine video content typical of cloud gaming services, we utilise the UT Video Codec format to record the original footage. The UT Video Codec is a lossless video encoding format, extensively utilised in professional video editing and post-production, especially beneficial in scenarios that demand frequent encoding and decoding without any loss in video quality. This codec maintains the integrity of the video quality and is ideally suited for simulating the original data content from cloud gaming servers. Conversely, for the compressed video, we employ the H.265 format, which is the current standard for high-efficiency video compression used by major media platforms today. This format is particularly adept at handling high-resolution video data and is crucial for providing high-quality video while significantly minimising the need for data transmission and storage. Depending on the type of game content, we adjust the bit rate at which the original videos are compressed to

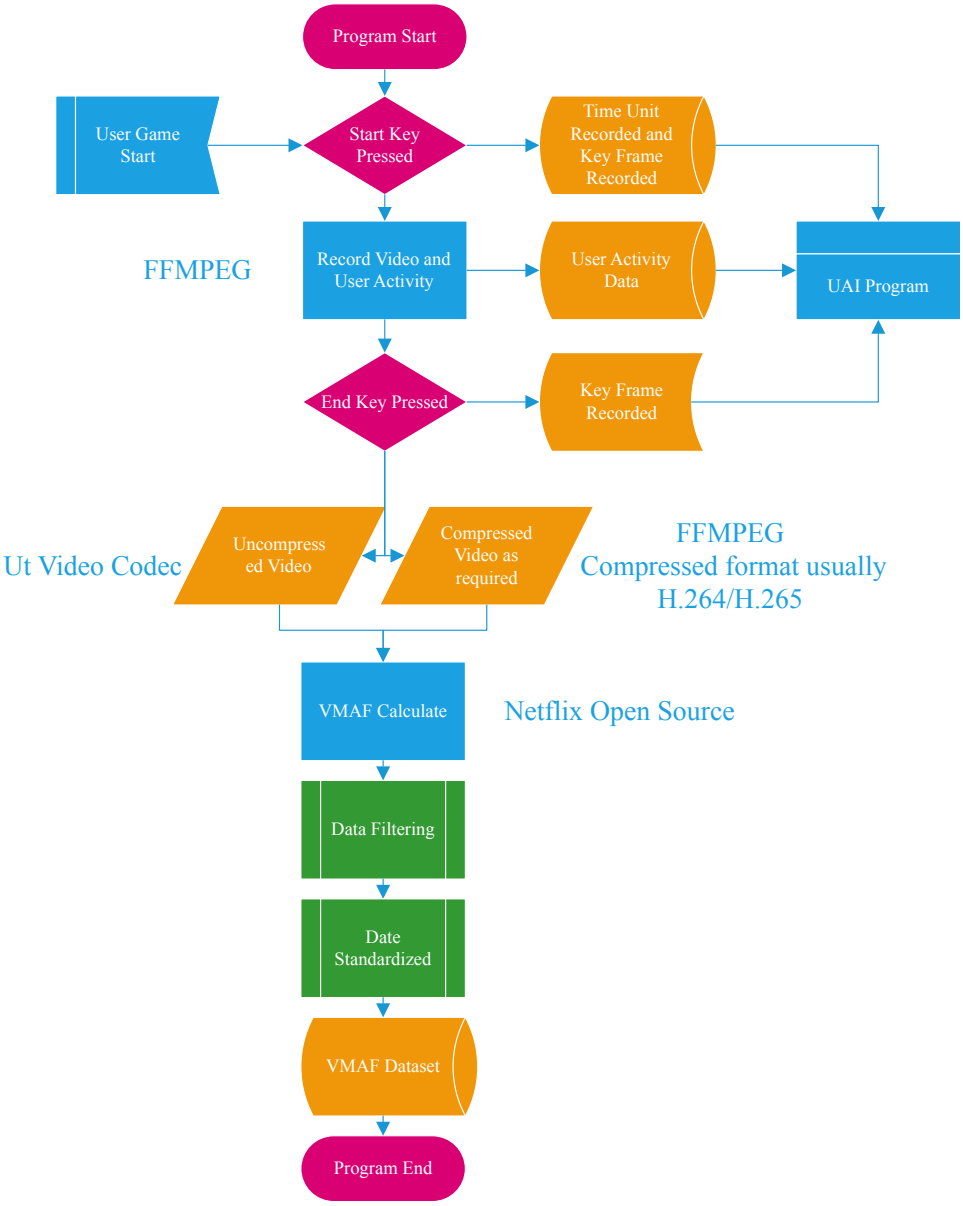


Fig. 3.2 Video Record and VMAF Calculate Program Workflow

further simulate the variations in video quality that users experience under different network conditions.

In summary, by applying distinct treatments to the VMAF sample videos and compressed videos and by integrating these with the timing characteristics of the game user behaviour data, we have refined our processing of the VMAF data. This enhancement significantly bolsters the accuracy of the experiments related to this study, ensuring a more reliable reflection of how game content influences video quality assessments.

3.4 Data Normalisation

The raw data obtained from the keyboard and VMAF must first be normalised. Normalisation is a linear transformation that does not change the numerical ordering of the original data and is able to maintain the original information of the data. In model training, gradient descent is the

most commonly used solution algorithm, and if it is not normalised, the numerical range of the original data may vary greatly, and the contour lines formed in iterations will be very sharp or partially elliptical, resulting in an increase in the number of iterations and a slower convergence speed. In addition, if the numerical range of the data is very different, it may also cause the truncation error of floating-point calculation, or the cumulative error may increase, which will affect the relationship analysis between the data.

The primary normalisation methods are below:

Min-Max Normalisation

$$x'_{ik} = \frac{x_{ik} - min\{x_{ik}\}}{max\{x_{ik}\} - min\{x_{ik}\}}, \quad i,k = 1,2,... \tag{3.1}$$

Where:

- x_{ik}: Represents the original data value for an element with indices i and k.
- $\min\{x_{ik}\}$: Denotes the minimum value of x_{ik} over all i and k, or within a specific subset.
- $\max\{x_{ik}\}$: Denotes the maximum value of x_{ik} over all i and k, or within a specific subset.
- x'_{ik}: Represents the normalised value of x_{ik} after applying the transformation.
- $i,k = 1,2,\ldots$: Indicates the indices i and k can take on values $1,2,\ldots$, depending on the dimensions or scope of the data.

The formula adjusts x_{ik} by first subtracting the minimum value ($\min\{x_{ik}\}$), ensuring the minimum value of the transformed data becomes 0. Then it divides the result by the range ($\max\{x_{ik}\} - \min\{x_{ik}\}$), ensuring the maximum value of the transformed data becomes 1.

The Min-Max normalisation method linearly maps the raw data to the interval [0, 1], achieving data dimensionless and unifying different data ranges to the same scale. The principle of this technique is to normalise the data by scaling the minimum and maximum values of the original dataset. Consequently, when the raw data contains abnormal extreme values (maximum or minimum), the Min-Max normalisation method is relatively sensitive and impacts the normalisation effect. One compensating measure is to perform outlier detection and removal on the original data before normalisation, eliminating the individual abnormal extreme values to ensure the efficacy of the normalisation process.

Standardization Normalisation

$$x'_{ik} = \frac{x_{ik} - \bar{x_k}}{s_k} \tag{3.2}$$

$$where, \quad \bar{x_k} = \frac{1}{n}\sum_{i=1}^{n} x_{ik}, \quad s_k = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ik} - \bar{x_k})^2}, \quad i,k = 1,2,... \tag{3.3}$$

Where:

- x_{ik}: Represents the original data value for an element with indices i and k.
- x'_{ik}: Represents the standardised value of x_{ik} after applying the z-score normalisation.
- $\bar{x_k}$: Represents the mean of all values x_{ik} across the i-dimension for a fixed k:

$$\bar{x_k} = \frac{1}{n}\sum_{i=1}^{n} x_{ik}$$

  where n is the total number of elements in the i-dimension.

- s_k: Represents the standard deviation of all values x_{ik} across the i-dimension for a fixed k:

$$s_k = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ik} - \bar{x_k})^2}$$

- $i, k = 1, 2, \ldots$: Indicates the indices i and k can take on values $1, 2, \ldots$, depending on the dimensions or scope of the data.

Each x_{ik} is transformed by subtracting the mean ($\bar{x_k}$) to centre the data around 0, and then dividing by the standard deviation (s_k) to scale the data to have unit variance.

This method scales the raw data using the mean and standard deviation, transforming the original data into a normal distribution with a mean of 0 and a standard deviation of 1. This approach eliminates the influence of dimensions and magnitude, making the data more closely conform to a normal distribution, which is beneficial for normalising raw data that exhibits a Gaussian distribution. However, this method evidently has shortcomings for raw data that do not inherently possess a normal distribution. Moreover, the computational complexity of this method is relatively high.

Max Absolute Normalisation

$$x'_i = \frac{x_i}{\max(|x_i|)} \tag{3.4}$$

Where:

- x_i: Represents the original data value for an element with index i.
- x'_i: Represents the normalised value of x_i after applying max-absolute normalisation.
- $\max(|x_i|)$: Denotes the maximum absolute value of x_i across all elements:

$$\max(|x_i|) = \max\{|x_1|, |x_2|, \ldots, |x_n|\}$$

  where n is the total number of elements in the dataset.

- $i = 1, 2, \ldots, n$: Indicates the index i can take on values $1, 2, \ldots, n$, depending on the size of the dataset.

Each x_i is divided by the maximum absolute value $\max(|x_i|)$, ensuring all normalized values x_i' fall within the range $[-1, 1]$.

This method is to scale the original data by a particular ratio to reduce the difference of the original data and make the original data dimensionless. Its disadvantage is that it is sensitive to the extreme values of the original data.

Logarithm Normalisation

$$x_i' = log(x_i) \tag{3.5}$$

Where:

- x_i: Represents the original data value for an element with index i.
- x_i': Represents the transformed value of x_i after applying the logarithmic transformation.
- $\log(x_i)$: Refers to the natural logarithm (logarithm to the base e) of x_i. This can be generalised to other bases if needed:

  $$\log_b(x_i) = \frac{\ln(x_i)}{\ln(b)}$$

  where ln is the natural logarithm, and b is the base of the logarithm.
- $x_i > 0$: The logarithmic transformation is only defined for positive values of x_i. If the dataset contains non-positive values, a shift (e.g., $x_i' = \log(x_i + c)$, where $c > 0$) may be applied to ensure all values are positive.
- $i = 1, 2, \ldots, n$: Indicates the index i can take on values $1, 2, \ldots, n$, depending on the size of the dataset.

The logarithmic function compresses large values of x_i while expanding smaller values, making it useful for handling data with a wide range of exponential growth.

This method only achieves scaling of the raw data and cannot realise the dimensionless transformation of the original data; hence, it is generally not used independently. When the raw data is relatively concentrated in a dense region, and there are significant differences in values, this method is employed for scaling. Subsequently, other methods are utilised for normalisation.

3.5 Time Series Models

In a cloud gaming session, as the game content continuously changes, the user needs to constantly input control commands (keyboard, mouse, controller, etc.) to achieve specific

objectives within the game. This process ultimately generates a User Activity Index (UAI) sequence, which represents a series of control inputs. Under the influence of this input sequence, the game's scenes and logic change accordingly, resulting in the continuous variation of the cloud gaming streaming video content received by the user's terminal. Due to network conditions, dynamic video compression technology is employed to compress the streaming video received by the user's terminal in real-time. By simulating the cloud gaming service environment, we compare the original cloud gaming video data with the video data obtained by the user's terminal in real-time, thereby producing another output sequence called Video Multimethod Assessment Fusion (VMAF). Due to cloud gaming services' interactivity and temporal correlation, these two sequences occur on the same timeline and can be considered as two stationary stochastic processes. In this section, we use Time Series Models to model the relationship between the UAI and VMAF.

3.5.1 Stationary Stochastic Process

A stationary stochastic process is called time series if its parameters are discrete and represent time. [133] For example, a series

$$\{Y_t, t = ..., -2, -1, 0, 1, 2, ..., Y_t \in R\} \tag{3.6}$$

if $(1)E(Y_t) = \mu(constent), t = 0, \pm1, \pm2, ...; (2)E(Y_tY_{t-k})$ is independent of t, $k = 0, \pm1, \pm2, ...$ then Y_t is called stationary time series. Here $E(\cdot)$ presents the expectation.
Let:

$$r_k = E[(Y_t - \mu)(Y_{t+k} - \mu)], k = 0, \pm1, \pm2, ... \tag{3.7}$$

$$\rho_k = E[\frac{Y_t - \mu}{\sigma} \cdot \frac{Y_{t+k} - \mu}{\sigma}], where \quad \sigma^2 = D(Y_t) \tag{3.8}$$

r_k is called the auto-covariance function, ρ_k is called the auto-correlation function. μ, r_k, ρ_k are the most important statistical parameters of a time series used to characterise its dynamic features.

The purpose of time series analysis is to use the statistical parameters of some samples that could be sampled to estimate the statistics of the observing time series. Furthermore, to establish the linear model, such as the Auto-regressive (AR) model, Moving Average (MA) model, Auto-regressive Moving Average (ARMA) model, etc., and then use these models for prediction of a dynamic system.

3.5.2 Linear Model

Due to the randomness, periodicity, and short-term trend stability of time series, linear model theory can be employed for analysis and modelling. [145] Based on the different variation patterns of time series, the commonly used linear models include the following:

Auto-regressive(AR) Model

The general form of the AR Model is below:

$$Y_t = a_0 + a_1 Y_{(t-1)} + a_2 Y_{(t-2)} +, ..., + a_p Y_{(t-p)} + \phi_t \tag{3.9}$$

where Y_t is the value of the time series at time t, $Y_{(p-t)}$ is the value at time $p-t$, $a_i \in R, i = 0, 1, ..., p$ are called parameters, p is the order of the model, and ϕ_t is the model noise. Generally, ϕ_t is Gaussian white noise with a mean of 0 and a constant variance.

From the above equation, it can be observed that the foundation of the AR model is the assumption that a linear relationship exists between past data and current or future data. Therefore, historical data can be used to predict future data. The advantage of the AR model is its ability to capture the autoregressive properties of time series effectively. In cloud gaming streaming video, the changes in VMAF at the user's gaming terminal exhibit autoregressive characteristics. Consequently, it is feasible to attempt to model the relationship between UAI and VMAF using the AR model.

Moving Average(MA) Model

A stationary time series Y_t can also be represented as a weighted sum of white noise ϕ_t at time t and its q previous values $\phi_t, \phi_{(t-1)}, \phi_{(t-2)}, ..., \phi_{(t-q)}$, given by

$$Y_t = \phi_t - c_1 \phi_{(t-1)} - c_2 \phi_{(t-2)}, ..., - c_q \phi_{(t-q)} \tag{3.10}$$

where $c_i \in R, i = 0, 1, ..., q$, c_i are the model parameters, q is the order of the model, and the other symbols are the same as before. This equation is called the Moving Average (MA) Model of the time series Y_t.

From the expression of the MA model, it can be observed that the characteristic of the MA model is its ability to smooth historical data, making it effective in eliminating random disturbances and noise. This property makes it suitable for processing time series data with large fluctuations. However, due to the need to calculate the moving average, especially when the sliding window is designed to be wide, the MA model suffers from a significant time lag, reducing its responsiveness to sensitive data. Regarding the VMAF sequence of the user-end video quality, the magnitude of the averaging effect of historical data varies greatly depending on different game scenarios. Therefore, the characteristics of the MA model can be utilised to model the relationship between UAI and VMAF.

Auto-regressive Moving Average(ARMA) Model

Combining the above two models, we obtain the AutoRegressive Moving Average (ARMA) model for the time series Y_t, given by

$$Y_t - a_1 Y_{(t-1)} - a_2 Y_{(t-2)} -, ..., - a_p Y_{(t-p)} = \phi_t - c_1 \phi_{(t-1)} - c_2 \phi_{(t-2)}, ..., - c_q \phi_{(t-q)} \tag{3.11}$$

where $p > 0, q > 0, a_p \neq 0, c_q \neq 0$, and (p, q) is called the order of the model, denoted as ARMA(p,q).

Evidently, the ARMA model combines the characteristics of both the AR and MA models, making it adaptable to a wider range of time series forecasting problems. However, due to the increased complexity of the model, the number of parameters grows rapidly, and parameter estimation becomes more complicated, which may lead to a decrease in the model's robustness.

In this study, we primarily focus on investigating the correlation between UAI and VMAF. Therefore, it is necessary to analyse the variation patterns of VMAF under the influence of user behaviour control. Evidently, using any of the aforementioned models in isolation is insufficient. We need to comprehensively consider the characteristics of these models and incorporate the features of the ARMAX model, which will be discussed in the next subsection, to accurately capture the correlation patterns between UAI and VMAF. To facilitate subsequent research, we perform linear transformations on the three models introduced above by introducing the delay operator, defined as B, it takes a transform as below:

$$BY_t = Y_{(t-1)} \tag{3.12}$$

further $\underbrace{BB...B}_{\text{k}} Y_t = Y_{(t-k)}$,that equals $B^k Y_t = Y_{(t-1)}$. meanwhile, $B^k \phi_t = \phi_{(t-k)}$. Let

$$\Phi(B) = 1 - a_1 B - a_2 B^2 -, ..., -a_p B^p \tag{3.13}$$

$$\Psi(B) = 1 - \phi_1 B - \phi_2 B^2 -, ..., -\phi_q B^q \tag{3.14}$$

then, obtain the compact form of $ARMA(p, q)$ like follows

$$\Phi(B) Y_t = \Psi(B) \phi_t, \quad t = 0, \pm 1, \pm 2, ... \tag{3.15}$$

Using the delay operator, $AR(p)$ and $MA(q)$ are expressed as

$$\Phi(B) Y_t = \phi_t \tag{3.16}$$

$$Y_t = \Psi(B) \phi_t \tag{3.17}$$

respectively.

Auto-regressive Moving Average(ARMA) Model

The three models mentioned above represent situations without external input sources. When an external input source is present, the model becomes a controlled autoregressive model, also known as the ARMAX (AutoRegressive Moving Average with eXogenous inputs) model. The

ARMAX model can be expressed as follows:

$$Y_t - a_1 Y_{(t-1)} - a_2 Y_{(t-2)} -, ..., -a_p Y_{(t-p)} = \tag{3.18}$$

$$b_0 + b_1 u_{(t-1)} + b_2 u_{(t-2)} +, ..., +b_q u_{(t-m)} + \varphi_t + \tag{3.19}$$

$$\phi_t - c_1 \phi_{(t-1)} - c_2 \phi_{(t-2)} -, ..., -c_q \phi_{(t-q)} \tag{3.20}$$

where u_t is the external input to the time series at time t, $u_{(t-m)}$ is the input at time $t-m$, $b_i \in R, i = 0, 1, ..., m$ are the model parameters, m is the input order of the model, and φ_t is the external source input noise. The other symbols are the same as before. Using the delay operator, it can be transformed as below:

$$\Phi(B) Y_t = \Theta(B) u_{t-m} + \Psi(B) \phi_{t-q}, \quad t = 0, \pm 1, \pm 2, ... \tag{3.21}$$

where,

$$\Theta(B) = 1 + u_1 B + u_2 B^2 +, ..., +u_m B^m \tag{3.22}$$

the others are the same as above.

According to the theory of stochastic processes, when sufficient samples of Y_t and u_t are obtained, the estimates of the parameters in the above models can be calculated using the sample data. Consequently, the models can be employed to forecast Y_t.

3.5.3 Time Series Model Properties and Model Order

To analyse the dynamic patterns of UAI and VMAF using time series models, it is essential to determine the order of the model. Since linear models possess the properties of tailing and truncation, this forms the theoretical foundation for order determination in linear models.

The Tailing and Truncation of Time Series Model

For a stationary stochastic series(3.6), consider a segment containing $k+1$ $(k \geq 1)$ values, denoted as:

$$Y_t, Y_{t+1}, ..., Y_{t+k} \tag{3.23}$$

Here, we use the combination of the first k terms to estimate Y_{t+k}, given by:

$$\sum_{j=1}^{k} \phi_{kj} Y_{t+k-j} = \phi_{k1} Y_{t+k-} + \phi_{k2} Y_{t+k-2} + ... + \phi_{kk} Y_t \tag{3.24}$$

To estimate Y_{t+k}. Here, $\phi_{k1}, \phi_{k2}, ..., \phi_{kk}$ are coefficients, and we use the least variance method to determine these coefficients. That is, we choose $\phi_{k1}, \phi_{k2}, ..., \phi_{kk}$ to minimise

$$\delta = E(Y_{t+k} - \sum_{j=1}^{k} \phi_{kj} Y_{t+k-j})^2 \tag{3.25}$$

to minimal. According to the theory of extrema for multivariable functions, we establish the following system of equations:

$$\begin{cases} \frac{\partial \delta}{\partial \phi_{k1}} = E[2(Y_{t+k} - \phi_{k1}Y_{t+k-1} - \phi_{k2}Y_{t+k-2} - ... - \phi_{kk}Y_t) \cdot (-Y_{t+k-1})] = 0 \\ \frac{\partial \delta}{\partial \phi_2} = E[2(Y_{t+k} - \phi_{k1}Y_{t+k-1} - \phi_{k2}Y_{t+k-2} - ... - \phi_{kk}Y_t) \cdot (-Y_{t+k-2})] = 0 \\ ..... \\ \frac{\partial \delta}{\partial \phi_{kk}} = E[2(Y_{t+k} - \phi_{k1}Y_{t+k-1} - \phi_{k2}Y_{t+k-2} - ... - \phi_{kk}Y_t) \cdot (-Y_t)] = 0 \end{cases}$$

Simplifying and applying (3.8), we can get:

$$\begin{bmatrix} 1 & \rho_1 & \rho_2 \cdots & \rho_{k-1} \\ \rho_1 & 1 & \dots & \rho_{k-2} \\ \rho_2 & \vdots & \ddots & \vdots \\ \vdots & \vdots & \dots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \dots & 1 \end{bmatrix} \cdot \begin{bmatrix} \phi_{k1} \\ \phi_{k2} \\ \vdots \\ \vdots \\ \phi_{kk} \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \vdots \\ \rho_k \end{bmatrix} \tag{3.26}$$

Solving this equation yields $\phi_{k1}, \phi_{k2}, ..., \phi_{kk}$. where let $\phi_{00} = 1$, and $k \geq 1$. ϕ_{kk} is called the partial auto-correlation function. The significance of the partial auto-correlation function is that it describes the degree of relationship between Y_t and Y_{t+k} in any segment of length $k+1$, i.e., $Y_t, Y_{t+1}, ..., Y_{t+k-1}, Y_{t+k}$, of a stationary stochastic series, given that $Y_{t+1}, ..., Y_{t+k-1}$ remain constant. In other words, ϕ_{kk} reflects the relationship between Y_{t+k} and Y_t without depending on the intermediate values. On the other hand, the auto-correlation function ρ_k reflects the relationship between Y_{t+k} and Y_t while accounting for the dependence on the intermediate values.

For a stationary stochastic series, ρ_k, ϕ_{kk} has the properties of truncation and tailing. The tailing is:

$$k \to \infty, |\rho_k| < ce^{-\delta k}, (c > 0, \delta > 0), \text{that is } \lim_{k \to \infty} \rho_k = 0 \tag{3.27}$$

The truncation is:

$$\phi_{kk} \quad \text{is} \quad \begin{cases} = 0 & \text{if} \quad k > p \\ \neq 0 & \text{if} \quad k \leq p \end{cases} \tag{3.28}$$

	AR(p)	MA(q)	ARMA(p,q)
ρ_k	tailing	truncation	tailing
ϕ_{kk}	truncation	tailing	tailing

These properties form the theoretical foundation for using linear models in our modelling approach. [47]

Order Determination for Linear Models

In theory, the order of the model can be determined by calculating the ρ_k and ϕ_{kk} of the stationary series. However, in practical engineering applications, the situation is quite complex. From an engineering perspective, a good linear model should satisfy two basic conditions: 1. The model's regression accuracy for the time series should meet the usage requirements; 2. The order of the model should be as low as possible.

Satisfying the regression accuracy is the primary concern, and under the condition of satisfying the accuracy, a low-order model obviously has the advantage of being structurally simple and easy to calculate. In addition, choosing an appropriate model category (AR, MA, or ARMA) is also important. The AR model can well describe the auto-regressiveness of the stationary stochastic series, while the MA model highlights the influence of the series' cumulative effect on future changes. ARMA can take into account the advantages of both, but it also makes the model more complex and inevitably introduces additional uncertainties.

When the model has an external input source, the variation patterns of the stationary stochastic series become more complex. From the ARMAX structure (3.20), it can be seen that the external input enters the model in an additive manner, which is the same as the behaviour of ϕ_t in the model. Therefore, on the one hand, u_t affects the change of y_{t+1}; on the other hand, it also has the effect of reducing the influence of ϕ_t on y_{t+1}. At the same time, the variation pattern of the external input source u_t is usually not a stationary stochastic series but rather a control added according to some subjective intention.

In the relationship between UAI and VMAF studied in this thesis, it is evident that using linear models for analysis cannot fully conform to the patterns expressed by linear models. In the interaction process of cloud gaming, factors such as game genres and user behaviour control intentions are key factors influencing UAI, and these factors inevitably have a crucial impact on the behaviour of linear models. This point increases the difficulty of using linear models to analyse the relationship between UAI and VMAF.

The classical method for determining the order of linear models is based on sample data. By calculating the estimates of ρ_k and ϕ_{kk} from the sample, denoted as $\hat{\rho}_k$ and $\hat{\phi}_{kk}$, respectively, and then using hypothesis testing theory to determine the truncation and tailing of $\hat{\rho}_k$ and $\hat{\phi}_{kk}$, the order of the model can be determined. That is,

$$\hat{\rho_k} \sim N(0, \frac{1}{N}[1+2\sum_{i=1}^{m}\rho_i^2]), k > m \tag{3.29}$$

$$\hat{\phi_{kk}} \sim N(0, \frac{1}{N}), k > n \tag{3.30}$$

where N is the sample length, n is the order of AR(n), and m is the order of MA(m).

For ARX or ARMAX models, due to the presence of external input sources, the above classical order determination method is evidently not entirely applicable.

3.5.4 Delay Analysis between UAI and VMAF

For most game content, user operations and game content often have a high degree of correlation. In addition, user operations and game content often have autocorrelation in time. For example, in FPS games, the game content of the next frame is generated based on the content of the previous frame's game scene, and the video content and corresponding VMAF will also produce auto-correlated changes as a result. For game operations, the player's operation in the next frame also depends on the user's decision and operation content in the previous frame. Therefore, we can find that both UAI and VMAF have a high degree of autocorrelation based on time series. As a result, in our research, determining the time delay of correlation is a key point.

VMAF is obtained through calculations based on sample videos and original videos, while UAI, due to the influence of system input and output delays, cannot be completely synchronised with the video content. This implies that UAI lags behind VMAF. Consequently, some degree of delay inevitably exists between the collected UAI and VMAF sequences. Accurately determining this delay is crucial for precisely regressing the relationship between UAI and VMAF. However, due to the influence of various factors, accurately estimating this delay is extremely challenging.

Generally, there are several methods for estimating the delay between two sequence signals:

Correlation Analysis Estimation Method

This method estimates the time delay between two signals by calculating the position of the maximum peak value of the cross-correlation function (where the similarity between the two signals is highest). The advantage of this method is its simplicity in computation, and when the signal-to-noise ratio is high, the estimation accuracy is very precise. However, when the signal-to-noise ratio is low, the peak value of the correlation function may experience jitter, blur, or even pseudo-peaks, leading to lower estimation accuracy.

Higher-order Cumulant Estimation Method

This method utilises the property that the higher-order (above second-order) cumulants of Gaussian signal sequences are identically zero to estimate the time delay between signals. Therefore, when the signal noise follows a Gaussian distribution well, this method provides relatively accurate estimates.

Feature Structure Analysis Delay Estimation Method

This method constructs the covariance matrix of the signal sample data, performs eigenvalue decomposition, and then uses the orthogonal spectrum of the signal and noise subspace to search for the pseudo-spectrum related to the time delay for delay estimation. This method requires the calculation of the Rayleigh resolution limit and belongs to the super-resolution time delay estimation approach. It is computationally complex and has stringent usage conditions.

Cost Function Estimation Method

This method establishes a cost function based on the minimum average cost criterion and iteratively solves for the time delay estimate that satisfies the criterion. The advantage of this method is its low dependence on the statistical characteristics of the signal. It has received significant attention in multipath time delay estimation, but its computational complexity is relatively high, making it unfavourable for engineering implementation.

Adaptive Estimation Method

Under a certain optimal criterion, this method automatically adjusts the system structure and filter parameters in real-time until steady-state convergence is achieved, thereby obtaining the optimal solution for the time delay information. Common criteria include MMSE, RLS, Max-SNR, LCMV, etc..

It is important to note that all the above methods require the estimated signals to be stationary. When the estimated signals are non-stationary, or there are sudden strong interferences, the effectiveness of these methods will be significantly compromised. In this study, analysing the relationship between UAI and VMAF is a key issue. Therefore, starting from the theoretical foundation of linear models, we attempt to use the correlation analysis estimation method to perform time delay analysis on the collected two sequence data.

In summary, the approach for determining the order of the model in this thesis is as follows: firstly, establish $ARX(2,2,1)$ and $ARMAX(2,2,2,1)$ models, then gradually build higher-order models $ARX(step, step-1)$ and $ARMAX(step, step-1)$ with a step size of $step = \{1,2\}$. Simultaneously, evaluate the constructed models using the evaluation metric until the value of the metric does not change significantly, ultimately determining the model to be used.

3.6 Non-linear AutoRegressive Model with eXogenous Inputs (NARX)

The analysis in the previous section was based on time series models. In fact, the relationship between UAI and VMAF is extremely complex and influenced by multiple factors, exhibiting strong non-linear characteristics. This poses significant challenges for polynomial fitting techniques based on the principle of linear superposition. Therefore, this section attempts to use Artificial Neural Network (ANN) technology to analyse and model UAI and VMAF.

Artificial Neural Network (ANN) is an AI tool that mimics the working mechanism of the human brain. [27] It possesses strong non-linear mapping capabilities and can perform various complex tasks such as classification, non-linear regression, and pattern recognition. ANN can achieve self-learning and self-adaptation through sample data without the need to design complex mathematical models, offering high flexibility and wide applicability. ANN can also enable parallel computing, providing high computational efficiency when processing large-scale datasets and complex models. ANN does not rely on complex mathematical models, making it robust and

fault-tolerant. Moreover, with the development of AI technology, its ease of use and integration capabilities are becoming increasingly evident.

NARX is a type of Recurrent Neural Network (RNN) that combines the learning ability of feedforward networks with the dynamic adaptation mechanism of feedback networks, based on linear model theory. It is suitable for capturing the characteristics of time series data with historical effects. This section uses NARX to analyse and model UAI and VMAF.

3.6.1 Artificial Neuron

The basic unit of a neural network is the artificial neuron, which is similar to the most basic biological neuron. It can perform the three most fundamental processing tasks of biological neurons: (i) evaluating input signals and determining the strength of each input signal; (ii) calculating the weighted sum of all input signals and comparing it with the threshold of the processing unit; (iii) determining the output of the processing unit. The basic structure of a neuron is as follows Fig 3.3:

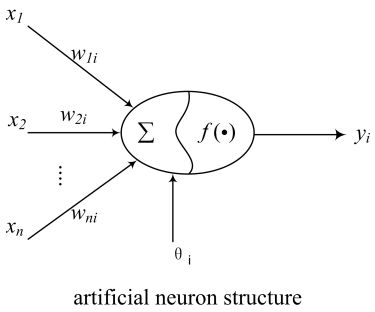


Fig. 3.3 Artificial Neuron Structure

where, $x_1, x_2, ..., x_n$ is the input of the neuron,$w_{1i}, w_{2i}, ...w_{ni}$ is the weight respectively, θ_i is the threshold of the neuron, $f(\cdot)$ is the activation function, y_i is the output of the neuron. The I/O of the neuron is as follows:

$$y_i = f(\sum_{j=1}^{n} w_{ji} + \theta_i) \tag{3.31}$$

In order to get a compact expression, let $x_0 = 1$, and

$$\boldsymbol{W_i} = \begin{pmatrix} \theta_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} \quad \boldsymbol{X_i} = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \tag{3.32}$$

so that the I/O of the neuron can be expressed:

$$y_i = W_i^T X \tag{3.33}$$

3.6.2 Activation Functions

The activation function is one of the key factors for a neural network by means of having the ability for non-linear mapping. In general, the activation function has such functions:

Add Non-linear Characteristics for Neural Network

Linear combinations in neural networks (i.e. the sum of the product of weights and inputs) can only represent linear relationships. However, many problems in the real world are non-linear. Activation functions (such as Sigmoid, ReLU, Tanh, etc.) introduce non-linearity, enabling neural networks to learn and represent complex non-linear relationships;

Control Output Range

Some activation functions can limit the output to a specific range. For example, the Sigmoid function limits the output to between 0 and 1, and the Tanh function limits the output to between -1 and 1. The control of this output range contributes to the stability and convergence of the network. This sparsity helps to reduce overfitting and improve the model's generalisation ability. Meanwhile, sparsity can also reduce computational complexity and accelerate training speed.

Gradient transfer

In the process of back-propagation, the activation function needs to have certain derivative characteristics in order to effectively transfer gradient information. This enables the network to adjust weights based on the loss function and achieve model optimisation.

So, selecting a proper activation function is very important to design an effective neural network. The often-used activation function is as follows:

Sigmoid Function

$$f(x) = \frac{1}{1+e^{-x}} \tag{3.34}$$

Tanh Function

$$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{3.35}$$

Swish Function

$$f(x) = \frac{x}{1+e^{-x}} \tag{3.36}$$

Exponential Linear Unit(ELU)

$$f(x) = \begin{cases} \alpha(e^x - 1), & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases} \tag{3.37}$$

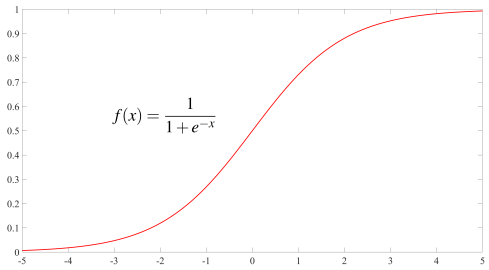
Rectified Linear Unit(ReLU)

$$f(x) = \max(0, x) \tag{3.38}$$

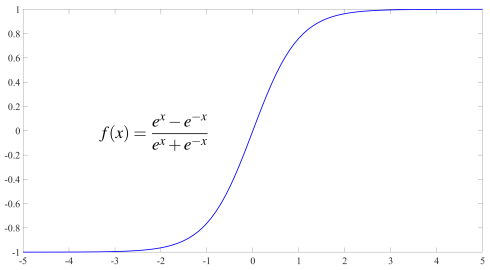
Leaky ReLU Function

$$f(x) = \begin{cases} \alpha x, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases} \tag{3.39}$$

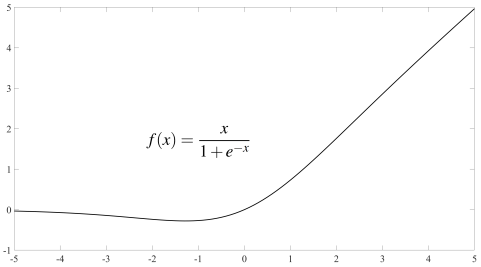
Its graphics are below:



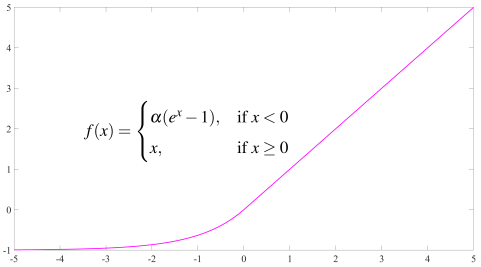
(a) Sigmod



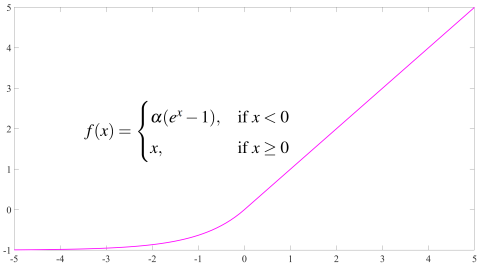
(b) Tanh



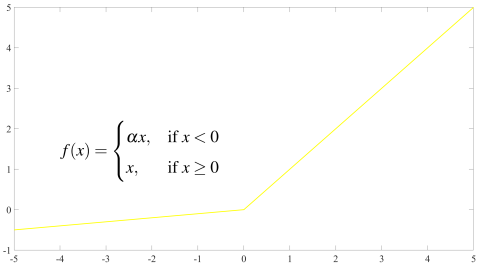
(c) Swish



(d) ELU



(e) ReLU



(f) Leaky ReLU

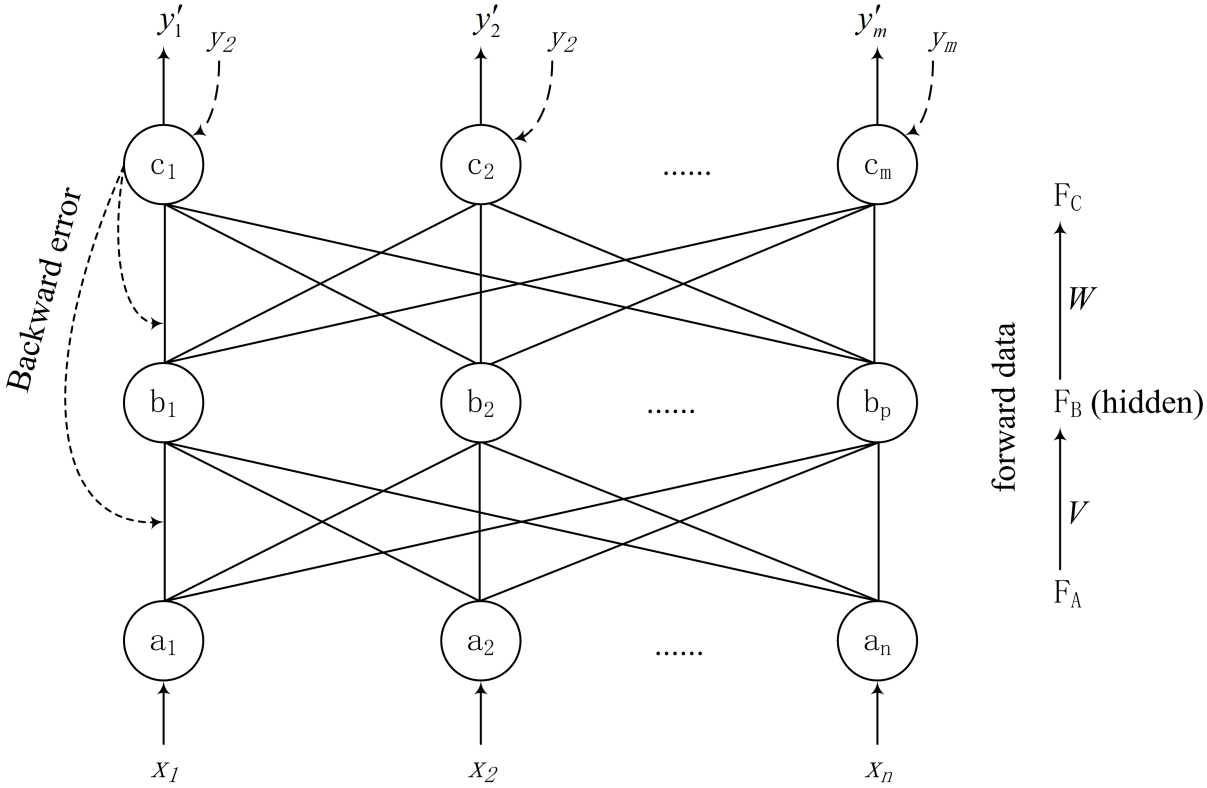
Fig. 3.4 Activation Functions

One neuron can be the simplest neural network, which can undertake classification tasks. Its weights and thresholds can be adjusted, and the activation should be selected as suitable to the tasks. [103]

3.6.3 Back-Propagation Mechanism and Feedforward Network

The propagation mechanism is the core of neural network self-learning. When a network structure is determined, the weights, thresholds, and transfer functions of the neurons in the network are initialised, and the sample data enters the network from the input end and generates output at the network output end after being calculated by the neurons. Since there is an error between the calculated output and the actual output of the sample, the error is decomposed from the output end and propagated backward layer by layer to correct the weights and thresholds of the neurons in each layer.

This principle is shown in the Fig 3.5 below:



Back propagation mechanism

Fig. 3.5 Back Propagation Mechanism

Assuming that when the nth sample is input, the calculated output of the jth neuron is $y_j(n)$, and the actual output of the neuron is $d_j(n)$, then the error of the neuron is:

$$e_j(n) = d_j(n) - y_j(n) \tag{3.40}$$

Let the square error of the neuron be $\frac{1}{2}e_j^2(n)$, then the total square error at the output of the network is:

$$E(n) = \frac{1}{2}\sum_{j\in c} e_j^2(n) \tag{3.41}$$

Assuming that the total number of training samples is N, the average total error generated after all samples are input into the network is:

$$E_{AV} = \frac{1}{N}\sum_{n=1}^{N} E(n) \tag{3.42}$$

E_{AV} is the objective function of network learning, and the purpose of network training is to minimise E_{AV}. When the network completes the minimisation of E_{AV}, the mapping of sample data from input to output can be realised. Obviously, once the sample data is determined, E_{AV} is a function of the network weight and threshold.

Assume that i, j are two adjacent layers of the network, $y_i(n)$ is the output of the ith layer, and there are p in total. Then the output of the jth layer is:

$$v_j(n) = \sum_{i=0}^{p} w_{ji}(n) y_i(n) \tag{3.43}$$

Where, $w_{ji}(n)$ is the weight of the jth layer. Let ϕ_j be the transfer function of this layer, so the output of this neuron is:

$$y_j(n) = \phi_j(v_j(n)) \tag{3.44}$$

In order to find the extreme value of E_{AV}, according to the extreme value theory of multivariate functions, find the partial derivative of $E(n)$ with respect to w_{ji}, then we have:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)} \cdot \frac{\partial e_j(n)}{\partial y_j(n)} \cdot \frac{\partial y_j(n)}{\partial v_j(n)} \cdot \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{3.45}$$

From (3.40) to (3.44), we can get $\frac{\partial E(n)}{\partial e_j(n)} = e_j(n)$, $\frac{\partial e_j(n)}{\partial y_j(n)} = -1$, $\frac{\partial y_j(n)}{\partial v_j(n)} = \phi_j'(v_j(n))$, $\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n)$ Then:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n)\phi_j'(v_j(n))y_i(n) \tag{3.46}$$

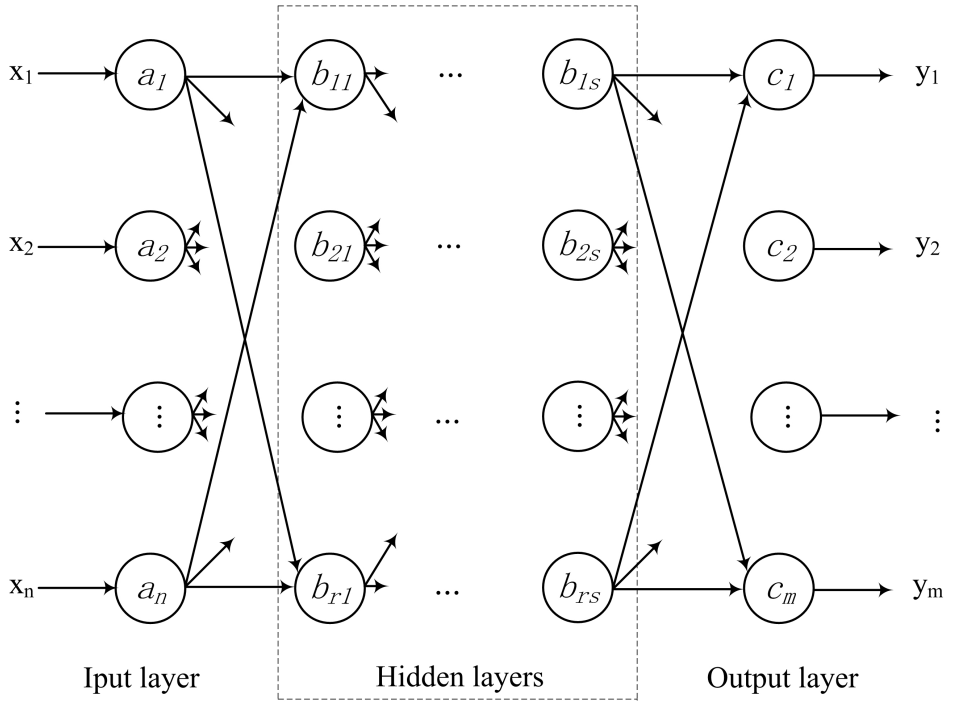
Then we get the weight correction:

$$\Delta w_{ji} = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)} = -\eta \delta_j(n) y_i(n) \tag{3.47}$$

Among them, $\delta_j(n) = e_j(n)\phi_j'(v_j(n))$ is called the local gradient, and $-\eta$ is called the learning step. After the network training starts, the first sample is input into the network, the first output is obtained, and then the first weight correction is calculated. The weight correction of each neuron is calculated layer by layer in reverse order. Based on this, all weights of the network are adjusted until all samples are input, and the first round of network training is completed. At

this time, check the total error of the network. If it meets the performance index requirements, stop training. If not, start the second round of training from the first sample again and repeat this cycle until the network meets the performance index requirements.

The back-propagation network is the basis of all kinds of neural networks. A typical back propagation network consists of an input layer, an output layer, and several hidden layers, as shown below Fig. 3.6:



Multilayer feed forward network

Fig. 3.6 Forward Layer Network

The number of neurons in the input and output layers is determined by the needs of the practical problem, while the number of hidden layers and the number of neurons in each layer are decided by the designer. Generally, the more hidden layers and neurons, the better the network's performance. However, a complex network structure also increases the difficulty of training and affects learning efficiency. Such a multi-layer feedforward network realises a mapping from an n-dimensional space to an m-dimensional space. Through weight learning, it can accomplish tasks such as non-linear approximation and classification. A multi-layer feedforward network is a supervised network, meaning that obtaining sample data is a prerequisite for completing weight learning.

An important characteristic of a multi-layer feedforward network is that input data can only be forwarded from front to back, and there are no connections between neurons within the same layer. Neurons in adjacent layers are fully connected. This characteristic determines that a feedforward network is a static network; that is, a well-trained feedforward network can reproduce the I/O relationship implied by the sample data. [52] However, when the sample data changes or new data exceeds the range of the training sample data, the mapping ability of the feedforward network will greatly decrease. This is the reason why feedforward networks have

poor generalisation ability. Additionally, the static nature of feedforward networks determines that they cannot handle data with I/O delays or historical effects. To address this, a feedback mechanism is added to make the network dynamic, giving rise to feedback networks.

3.6.4 Feedback Network and Time Delay Line

The feedback mechanism of the neural network is illustrated in the Fig. 3.7. A distinctive feature of this structure is the connectivity among neurons within the same layer, where the output of a neuron not only feeds back to its own input but also to the inputs of other neurons. Such architecture endows the network with dynamic effects, enabling memory and associative functions, making it suitable for processing data with historical implications. The learning of weights in the feedback network is also facilitated through the error back-propagation mechanism. In this process, the data flow of back-propagation includes not just the errors but also the output feedback from the neurons. The non-linear auto-regressive network with exogenous inputs (NARX) is a classical feedback neuron network. The research experimental department will analyse the series of relationships between UAI and VMAF through the NARX network.

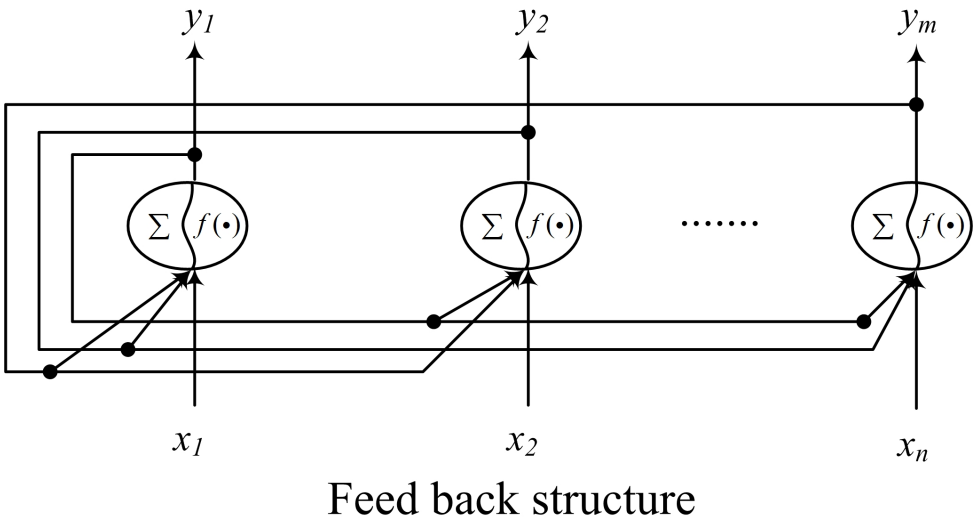


Fig. 3.7 Feed Back Network

Another mechanism to realise the use of historical data is to add time delay to the data flow of the network, including input delay and output delay. This idea can be realised by designing Tapped Delay Line (TDL).

A tapped delay line is a delay line with at least one 'tap'. A delay-line tap extracts a data output from somewhere within the delay line, optionally scales it, and usually sums with other taps to form output data. A tap may be interpolating or non-interpolating. A non-interpolating tap extracts the data at some fixed integer delay relative to the input. Thus, a tap implements a shorter delay line within a larger one, as shown in Fig 3.8. Tapped delay lines efficiently simulate multiple signals from the same source data. As a result, they are extensively used in the design of delay networks.

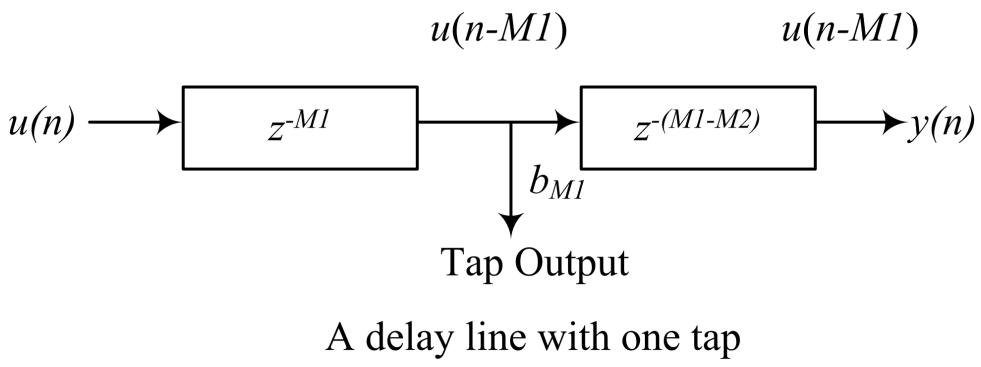


Fig. 3.8 Delay Line with One Tap

A complete structure of a TDL with two internal taps is shown in Fig. 3.8. The total delay line length is M3 samples, and the internal taps are located at delays of M1 and M2 samples, respectively. The output signal is a linear combination of the input signal $u(n)$, the delay-line output $u(n-M3)$, and the two tap signals $u(n-M1)$ and $u(n-M2)$.

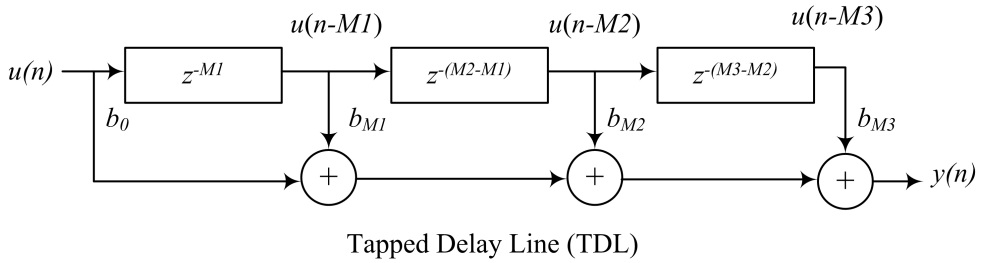


Fig. 3.9 Tapped Delay Line (TDL)

The difference equation of the TDL shown in Fig. 3.9 is, by inspection,

$$y_t(n) = b_0 u(n) + b_{M1} u(n-M1) + b_{M2} u(n-M2) + b_{M3} u(n-M3) \tag{3.48}$$

corresponding to the transfer function

$$\phi(z) = b_0 + b_{M1} z^{-M1} + b_{M2} z^{-M2} + b_{M3} z^{-M3} \tag{3.49}$$

3.6.5 Auto-Regressive Network with eXogenous Inputs

Generally, dynamic networks to be used usually have either been focused networks, with the dynamics only at the input layer, or feedforward networks. The non-linear auto-regressive network with exogenous inputs is a recurrent dynamic network, with feedback connections enclosing several layers of the network. The auto-regressive network with exogenous inputs is based on the linear ARX theory, which can be used in time-series modelling. The defining equation for the NARX is

$$y_t = f(y_{(t-1)}, y_{(t-2)}, \ldots, y_{(t-p)}, u_{(t-1)}, u_{(t-2)}, \ldots, u_{(t-q)}) \tag{3.50}$$

where, y_t is the prediction of the next time, which is regressed on previous values of the output signal and previous values of an independent (exogenous) input signal. Its structure is below Fig 3.10:

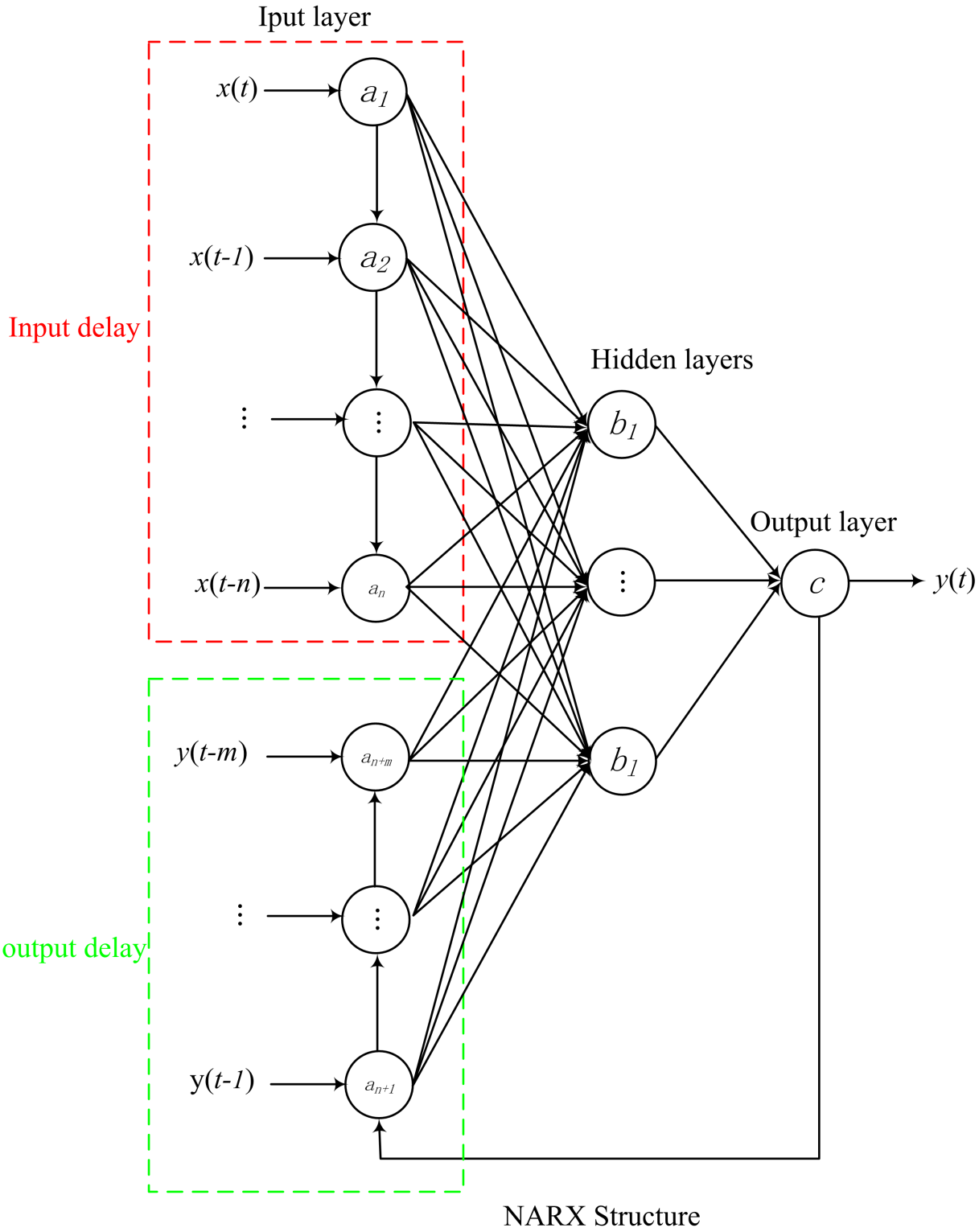
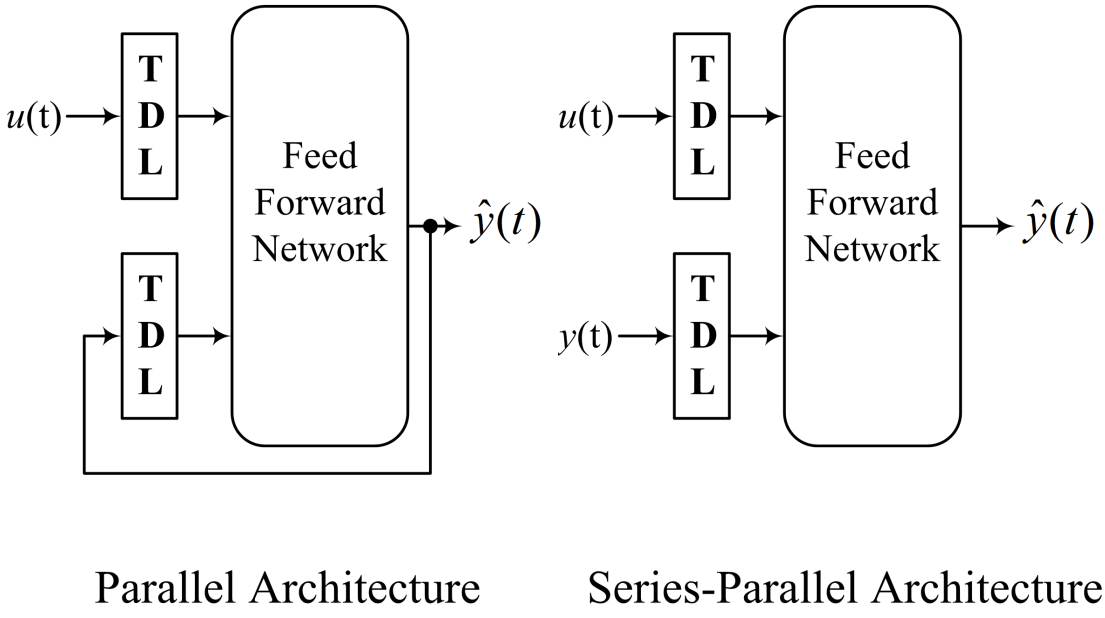


Fig. 3.10 NARX Structure

The NARX can be implemented by using a feedforward neural network to approximate the function f.

In order to get good training, the auto-regressive network with exogenous inputs can be configured as follows: two architectures, parallel architecture (left figure) and series-parallel architecture, shown as follows Fig. 3.11:



(a) Parallel Architecture (b) Series-Parallel Architecture

Fig. 3.11 Two NARX Architectures

In the parallel architecture, the output of the auto-regressive network with exogenous inputs can be considered as an estimate of the output of some non-linear dynamic system trying to model. The output is feedback to the input of the feedforward neural network as part of the standard NARX architecture. However, because the true output is available during the training of the network, a series-parallel architecture could be created in which the true output is used instead of feeding back the estimated output. This has two advantages. The first is that the input to the feedforward network is more accurate. The second is that the resulting network has a purely feedforward architecture, and static back-propagation can be used for training. Generally, the sample data are always available for the dynamic system to be modelled; the series-parallel architecture is configured first and used to train the auto-regressive network with exogenous inputs to model a dynamic system. After complete training, the program function can be employed to convert the auto-regressive network with exogenous inputs from the series-parallel configuration, which is useful for training, to the parallel configuration and then take the parallel configuration model to perform prediction.

3.6.6 Structure and Parameters

When using a NARX (Non-linear Auto-Regressive Network with Exogenous Inputs) network to analyse time series, a crucial issue is the determination of delay parameters for the network's inputs and outputs. In linear model approaches, the order of the model indicates the extent of its dependency on historical data; a higher order signifies a stronger dependency on this data.

The NARX network is built upon the ARX model, and the input-output delay parameters of the network reflect its level of dependency on historical data.

In the previous section, we identified a suitable order for the ARX model. This order serves as an important reference for constructing the NARX network in this section.

Network Structures

NARX networks are recurrent neural networks based on multi-layer feedforward networks with added feedback and delay mechanisms. Therefore, their non-linear mapping ability relies on the functionality of multi-layer feedforward networks. Both UAI and VMAF are one-dimensional data, which determines that the input/output layers of the network are also one-dimensional. Determining the number of hidden layers and the number of neurons in each layer is the core of network structure design. From the analysis of the back-propagation mechanism in the previous section, it is known that the training objective of feedforward networks is to minimise the error between the calculated output and the actual output of the sample data. The modification of network weights is guided by this objective, while the number of layers and neurons in the network does not directly affect the training objective from a theoretical perspective.

From practical application experience of multi-layer feedforward networks, generally, the more hidden layers and neurons, the stronger the non-linear mapping ability. However, this also leads to a more complex network structure, a sharp increase in weight parameters, slow training speed, and a tendency to fall into local optima. Too many neurons can also cause overfitting. Additionally, when it is necessary to increase the number of hidden layers or neurons in the network, prioritising increasing the number of layers or neurons is also a tricky problem. For example, a feedforward network with a total of 10 neurons can be designed as a single hidden layer structure containing 10 neurons, or it can be designed as a structure with 2 hidden layers, each containing 5 neurons, as shown in the Fig 3.12:

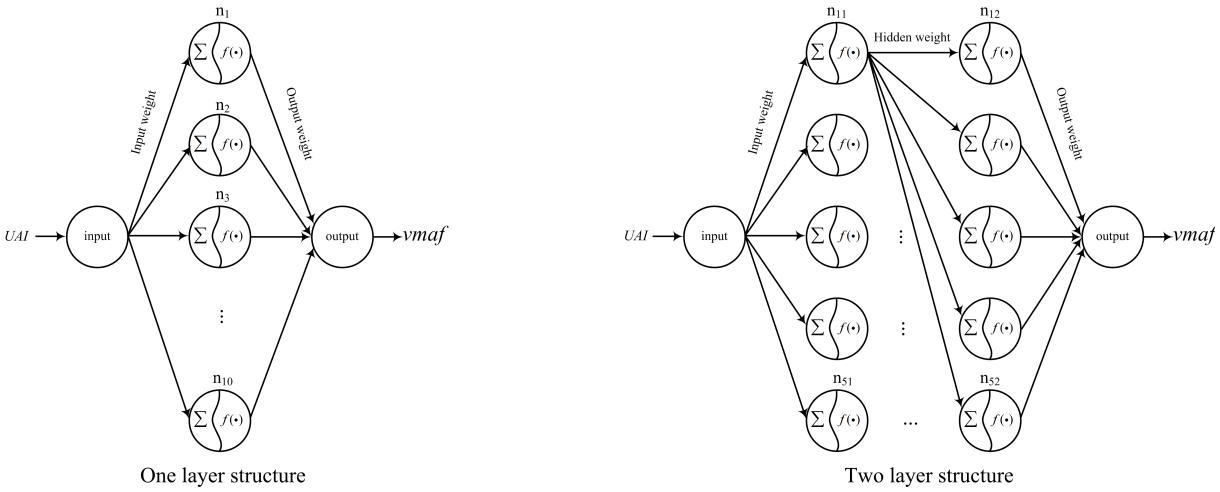


Fig. 3.12 10 Neuron Designed Structure

Clearly, under the same other settings, the two-layer structure increases the weight parameters by more than 50 per cent compared to the one-layer structure.

Practical application experience also suggests that increasing the number of hidden layers is beneficial for achieving better training accuracy while prioritising increasing the number of neurons, which is more advantageous for exploring the non-linear relationships hidden in completely unknown sample data. Therefore, the design approach for the network structure in this thesis is to first determine an upper bound for the number of neurons and hidden layers, then start training with the simplest network structure and initial number of neurons, and if the accuracy requirements are not met, gradually increase the number of neurons until the upper bound is reached. Then, change the number of hidden layers in the network and evenly distribute the number of neurons in each layer. Repeat this process until a satisfactory result is obtained.

After determining the basic structure of the NARX network, determining the delay of the network is another important issue. The inherent patterns of the sample data are the essential factors that determine the network delay. The original intention of designing NARX networks is to fully consider the delay characteristics present in the sample data, and linear model theory provides important guidance for designing the delay in NARX networks. The approach in this thesis is to use the results of the linear model analysis from the previous section as a basis and obtain a satisfactory network delay through continuous experimentation. It is important to note that different cloud gaming content and different users profoundly affect the relationship between UAI and VMAF. Especially in the cloud gaming service environment, these factors become extremely complex for streamed videos. A NARX network trained with one or several sets of sample data cannot be universally applicable. Extensive experimentation and improvement in the later stages are essential for perfecting the analysis of the relationship between UAI and VMAF.

Hyper-Parameters

In addition to the network architecture, hyper-parameters play a crucial role in determining the performance of neural networks. Hyper-parameters represent model configurations that are external to the model's parameters and cannot be estimated directly from sample data.

Generally, when sufficient background knowledge is available for the problem addressed by a neural network, these hyper-parameters can often be specified by the practitioner. [21] They may be set heuristically or adjusted based on the requirements of a specific predictive modelling task. [129] However, the optimal values for a model's hyper-parameters in a given context are typically unknown. Machine learning practice offers several approaches to managing hyper-parameters, including:

1. Applying heuristic rules of thumb
2. Adopting hyper-parameter values from similar problems
3. Searching for optimal values through trial and error
4. Employing optimisation algorithms, such as evolutionary algorithms, to fine-tune them [116]

Essentially, tailoring a machine learning algorithm to a specific problem involves tuning the model's hyper-parameters to identify the configuration that yields the most accurate predictions. [76]

Given the multitude of hyper-parameters present in a NARX network, identifying the optimal configuration can be a challenging, if not impossible, task. In this study, in addition to relying on commonly used or default values, the following hyper-parameters are the primary focus of adjustment:

net.divideMode- This parameter sets the ratio for dividing the sample dataset during network training. By calling the divideblock() function, it separates the target set into three sets: training, validation, and testing. The system's setting is 0.7:0.15:0.15. To check the network's ability to reconstruct the sample data, in the initial stages of analysis, one can attempt to use all the sample data for training.
net.trainParam.max_fail- This parameter sets the maximum number of validation failures during the network training process. When the patterns in the sample data are relatively complex, or when the network structure is set to be complex, too few validation failure attempts can lead to premature termination of network training. Conversely, too many validation failure attempts can lead to overfitting of the network.
net.trainParam.min_grad- This parameter sets the minimum value of the gradient during network training. Once this minimum value is reached, training will terminate regardless of whether the network meets the accuracy requirements. In other words, a trained network may not necessarily meet the requirements. By analysing this parameter, one can determine whether the training has reached the predetermined requirements.
net.trainParam.mu- This parameter, along with .mu_inc, .mu_dec, .mu_max, constitutes the Momentum adjustment strategy during network training. Momentum is a measure that maintains the stability of network training while accelerating network convergence. When network training is very difficult, one can try adjusting this parameter to change the training performance.
net.trainParam.epochs- This parameter sets the maximum number of iterations for network training. Once this maximum value is reached, training will stop regardless of whether the network's performance metrics are satisfactory. The .epoch parameter is also known as the 'hard condition' for training stoppage. Usually, for a network that stops training based on this parameter, further analysis is needed to determine whether its performance meets the target requirements. Increasing .epoch implies that more training time is required.

In addition to adjusting the aforementioned numeric hyper-parameters, the choice of network structure, time delay, activation functions, and training functions are also key hyper-parameters for the model. The network structure, time delay, and activation functions have been discussed in the preceding section; the primary training functions built into NARX are:

Gradient Descent This training function calculates the partial derivatives of the objective

function with respect to each network parameter. It then selects the direction corresponding to the largest partial derivative (i.e., gradient) to update the parameters, thereby gradually reducing the error of the objective function. It is a widely used method suitable for most back-propagation neural networks; however, its disadvantage is that it requires excessive training time. Due to the complexity of UAI/VMAF data and based on extensive experimentation, we have selected this training function as the initial choice for training our NARX network in the next chapter.

Momentum This training function is based on Gradient Descent; by incorporating a momentum term ($\pm\Delta$) into the gradient, it accelerates convergence and helps avoid local optima. When Δ equals zero, this method reduces to standard Gradient Descent. Therefore, selecting the initial value of Δ and its increment (_inc) and decrement (_dec) parameters is crucial for achieving optimal performance of the NARX network. All simulations in Chapter 4 employ a combination of these two methods, and the results presented in this thesis represent the best outcomes among numerous experiments.

BFGS(C.G.Broyden, R.Fletcher, D.Goldfarb, D.F.Shanno) This training function is known as the Quasi-Newton Method. Unlike the Newton Method, it does not require the computation and storage of complete Hessian matrices, thereby accelerating convergence. Since the Hessian matrix is often ill-conditioned, reducing its calculation enhances the robustness of the algorithm. This method is suitable for training with massive datasets. In this study, although the characteristics of the UAI/VMAF data vary greatly, the volume of data is not large; therefore, this method is seldom used.

Building upon the aforementioned discussion and considering the characteristics of the UAI/VMAF data analysed in this study, we will first utilise the data from CASE 1 to experiment with several key numerical hyper-parameters to determine their approximate ranges. This preparation is essential for the simulation research presented in the next chapter. The network architecture, delay configurations, and activation functions will be determined through various experiments conducted in the simulations of the subsequent chapter.

Hyper-parameters Configuration Trail

The primary principle of this experiment is that the selected set of hyper-parameters should ensure the stable operation of the NARX model and produce simulation results. Based on this foundation, we then aim to optimise the simulation efficiency of the NARX model. Referring to the parameters of the MATLAB/NARX object and in conjunction with the analysis from the previous section, we selected two sets of hyper-parameters for experimentation. The first set employs the upper limit values of each numerical parameter, aiming to achieve the fastest possible training speed. The second set uses the lower limit values of each numerical parameter to enhance the stability of the training process. The training results of these two sets of hyper-parameters are shown below:

The first set of training results:

Results	Reached maximum mu
Data Division	Random (dividerand)
Training	Bayesian Regularisation (trainbr)
Performance	Mean Squared Error (mse)
Calculations	MEX

Table 3.2 First Set Training Algorithm Table

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	165	500
Elapsed Time	-	0:00:01	-
Performance	5.24	0.00746	0
Gradient	8.8	0.000218	1.00×10^{-10}
Mu	0.001	1.25×10^{10}	1.00×10^{10}
Effective # Param	121	12.2	0
Sum Squared Param	248	3.67	0

Table 3.3 First Set Training Progress Table

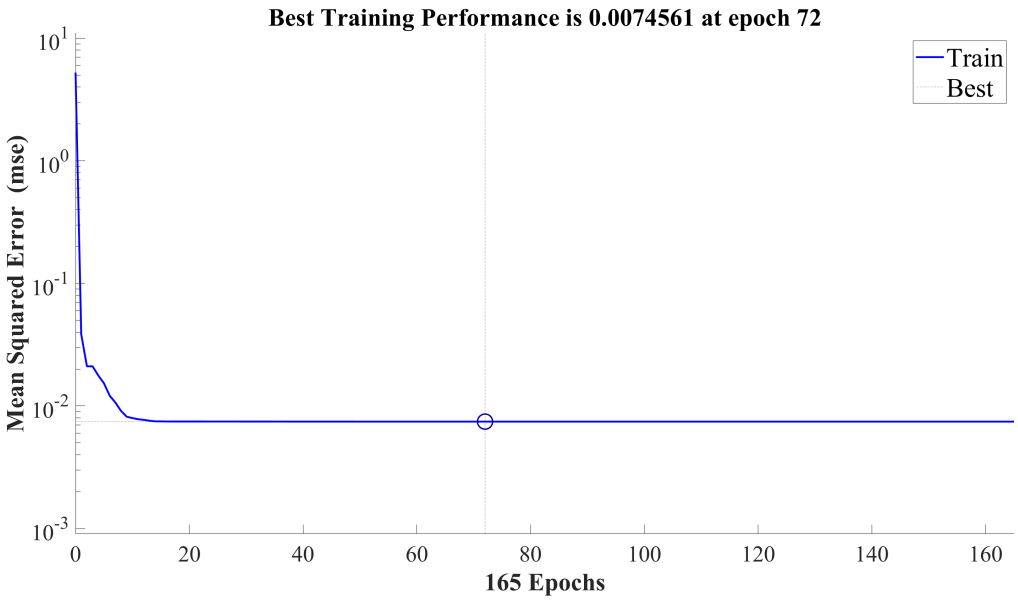


Fig. 3.13 First Set Training Performance

The simulation performance metrics for the first set of parameters are as follows:

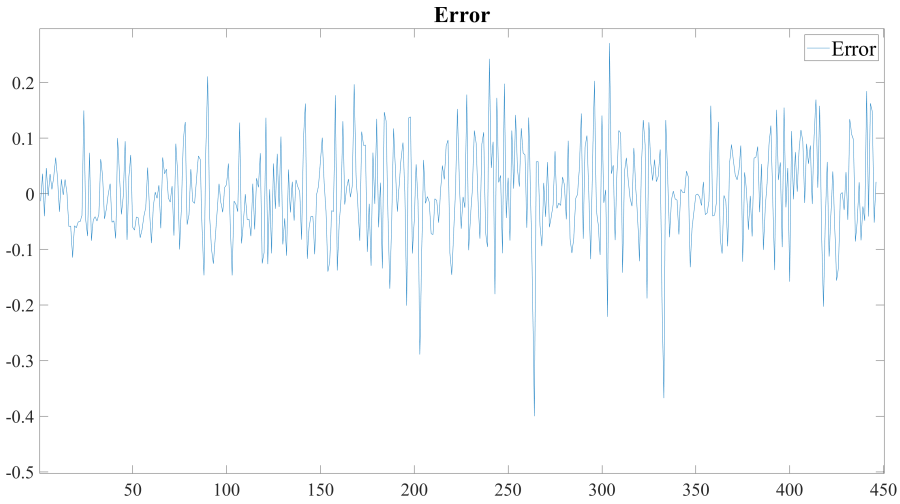


Fig. 3.14 First Set Training Error

Metrics	Performance Value
***e*-mean**	0.643 954 143 397 379
***e*-max**	$-3.733\,833\,876\,491\,061 \times 10^{-6}$
***e*-sum**	0.400 012 402 016 974
***e*-fit**	3.325 401 653 467 803

Table 3.4 First Set Simulation Performance Metrics

The second set of training results:

Results	Reached maximum number of epochs
Data Division	Random (dividerand)
Training	Bayesian Regularisation (trainbr)
Performance	Mean Squared Error (mse)
Calculations	MEX

Table 3.5 Second Set Training Algorithm Table

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	500	500
Elapsed Time	-	0:00:01	-
Performance	0.933	0.00714	0
Gradient	1,58	0.00021	1.00×10^{-5}
Mu	0.01	0.01	1.00×10^{8}
Effective # Param	121	8.76	0
Sum Squared Param	244	2.58	0

Table 3.6 Second Set Training Progress Table

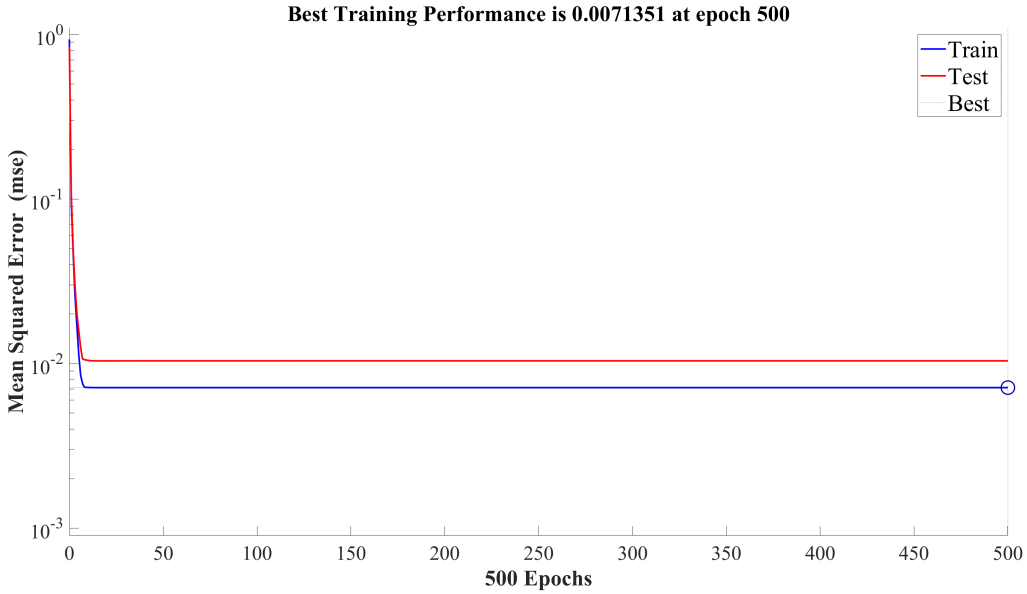


Fig. 3.15 Second Set Training Performance

The simulation performance metrics for the second set of parameters are as follows:

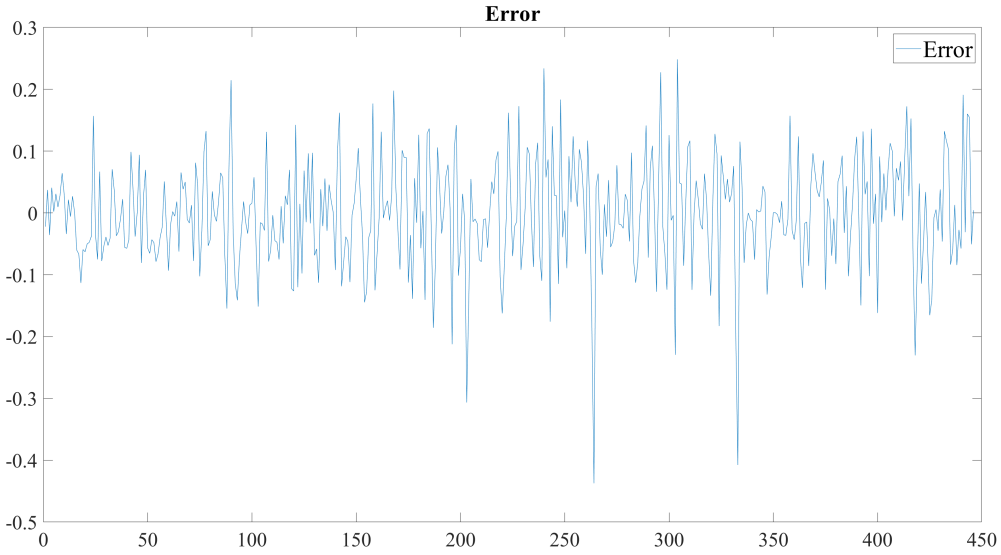


Fig. 3.16 Second Set Training Error

Metrics	Performance Value
***e*-mean**	0.639 996 587 442 842
***e*-max**	0.002 351 972 137 746
***e*-sum**	0.437 261 567 289 583
***e*-fit**	3.399 738 169 998 280

Table 3.7 Second Set Simulation Performance Metrics

By comparing the results of the two sets of hyper-parameters, it can be observed that the simulation using the first set of parameters terminated with the message "maximum mu reached" at epoch = 165. This indicates that the network did not sufficiently learn from the sample data and may have converged to a local optimum. In contrast, the second set of parameters stopped at the maximum number of epochs, and there was excellent consistency between the training performance metrics and the validation performance metrics, suggesting that the training with this set of parameters was more thorough. We attempted to further expand the search range; however, there was no significant change in either the training metrics or the simulation performance indicators.

The parameters of the MATLAB/NARX object referenced in this experiment, the settings of the search range, and the finally determined intervals for selecting hyper-parameters are shown in Table 3.8.

	Default	**Searching range**	**Selected**
.trainParam.goal	0	0	0
.trainParam.time	Inf'	Inf'	Inf'
.performFcn	'mse'	'mse'	'mse'
.trainFcn	trainlm'	trainlm/traingd'	trainlm'
.divideMode	'sample'	0.7:0.3/0.8:0.2	0.7:0.3
.trainParam.epochs	1000	[500 ~2 000]	1000
.trainParam.max_fail	6	[6 ~ 10]	10
.trainParam.min_grad	1.00×10^{-7}	$[1.0\times10^{-5} \sim 1.0\times10^{-10}]$	1.00×10^{-10}
.trainParam.mu	1.00×10^{-3}	[0.001~0.01]	1.00×10^{-3}
.trainParam.mu_max	1.00×10^{10}	$[1.0\times10^{8} \sim 1.0\times10^{10}]$	1.00×10^{10}
.trainParam.mu_inc	10	[5 ~ 10]	5
.trainParam.mu_dec	0.1	[0.05 ~ 0.1]	0.05

Table 3.8 NARX Hyper-parameters Searching Table

It is important to note that the two sets of parameters we selected correspond to the two extremes of the search range and exhibit significant differences; however, their simulation performance metrics do not vary substantially. This indicates that, for the UAI/VMAF data examined in this study, the numerical hyper-parameters of the NARX model are not particularly sensitive. Consequently, in the experiments conducted in Chapter 4, we will select the numerical hyper-parameters of the NARX model based on these experimental results. Our primary focus will be on investigating the selection of network architecture, delay configurations, and activation functions to achieve optimal simulation performance.

Performance Index

According to the back-propagation mechanism, the main indicator for evaluating the training effect of a NARX network is the Mean Squared Error (MSE). However, in practical applications, it is also necessary to consider indicators such as the complexity of the network structure, the number of training iterations, and the network's generalisation ability. Generalisation ability refers to the ability of a trained network to produce correct outputs for new input data. [147] Poor

generalisation ability is an inherent defect of feedforward networks, and NARX is no exception. This thesis uses NARX to analyse the relationship between UAI and VMAF, and special attention needs to be paid to this defect. The quality of generalisation ability will directly affect the effect of using the NARX network for multi-step prediction.

In addition, the network training effect indicators only reflect the performance of the network on the training sample dataset and do not fully reflect the regression effect of UAI and VMAF. Therefore, it is also necessary to use the trained NARX network to simulate each game dataset and comprehensively evaluate it in combination with the indicators in the next chapter.

Based on the above analysis, the network structure and training parameter settings are all in service of the network's performance indicators. Moreover, these settings vary depending on changes in sample data, usage environment, and other circumstances. This necessitates continuous adjustment of the network structure and training parameters based on actual situations, ultimately obtaining an effective NARX model through multiple experiments.

3.7 Summary

Through the content of this chapter, we have learned in detail that in the current research, we collected user operation data and video quality parameter VMAF for different games through various programs. During the collection process, in order to better quantify user operation data, we established the User Activity Index (UAI) to measure and unify different user operation behaviours within a unit of time. This provides a reliable data source for subsequent modelling and analysis. Following that, based on the problems faced by cloud gaming services and their characteristics, we proposed two modelling schemes. We conducted an in-depth analysis of the time series analysis and NARX used and discussed them in combination with the characteristics of cloud gaming video quality parameters.

Chapter 4

Results and Evaluations

4.1 Overview

Based on the above research content, we first introduce and discuss the data processing procedure in this chapter. After obtaining a large amount of experimental data through the programs and methods introduced in Chapter 3, we perform multiple pre-processing steps on the data and conduct standardisation. Following that, according to the modelling theories and methods proposed in Chapter 3, we model UAI and VMAF using time series analysis methods and the NARX model, respectively. Next, we verify the reliability of our model by modelling and analysing multiple sets of data from different types of games. At the same time, we also perform simple predictive analysis through the model and validate the prediction results to prove the correlation and predictability between UAI and VMAF. The simulation process is shown in Fig 4.1 in the next page.

4.2 User Activity Index and VMAF Data Processing

During the process of collecting User Activity Index (UAI) data, we undertook the following steps. Initially, we designed a program to ensure that the recording of the original video and the collection of UAI commenced simultaneously. Subsequently, for different games, we applied different weights to invoke the respective game collection programs, thus specifically capturing each frame's user behaviour data. We then conducted a per unit time analysis of the data obtained from different games by the collection program, removing erroneous and invalid operation data.

For VMAF, we employed a similar approach, initiating video recording simultaneously with the collection of user behaviour data. Moreover, we performed corresponding trimmings for the compressed videos with the original videos. Since compressed videos may eliminate a certain number of key-frames depending on the compression format used, during the trimming process, we ensured the consistency of the key frame lengths to prevent discrepancies in video quality calculations that could arise from missing key-frames.

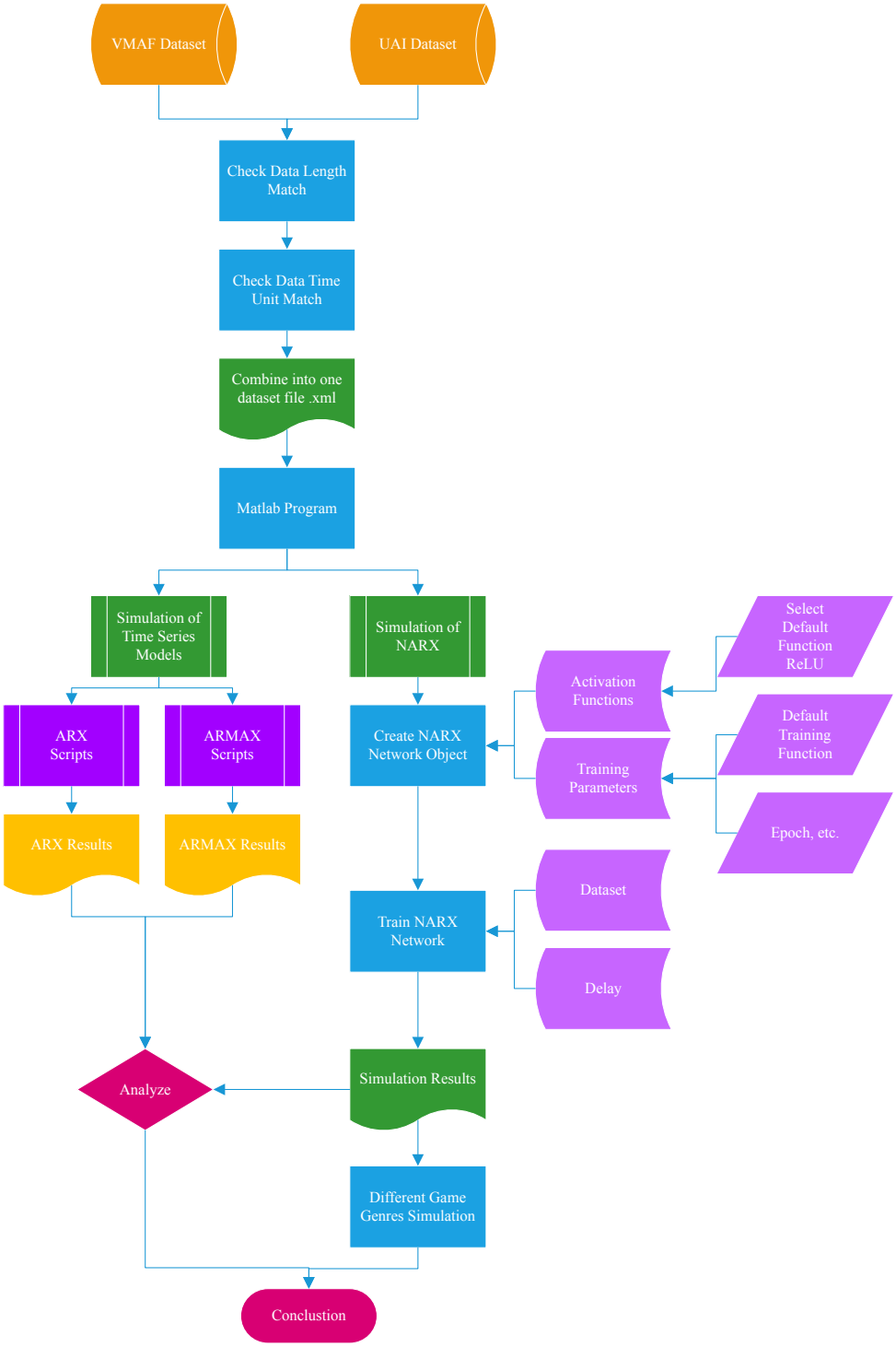


Fig. 4.1 Matlab Simulation Process Workflow

We also filtered and weighted the VMAF data to remove outliers that deviate from the video quality curve and calculated the average per unit time, thus correlating it with the user behaviour data. The codes used for these processes are provided in the appendix.

Finally, we selected the Min-Max normalisation method, which is introduced in Chapter 3. Since the VMAF index is presented in the form of a percentage, we mapped the UAI data to the interval [0, 1] as well. This approach dimensionality transforms the two key model input indicators, unifying their ranges to the same scale.

4.3 Overall Latency and White Noise

4.3.1 Overall Latency for Cloud Gaming

In Chapter 2, we discussed that cloud gaming is an application scenario centred on cloud computing technology. In this framework, game rendering and the processing of user inputs and graphical outputs are executed on cloud servers, with streaming technology serving as a critical enabler. [20] The player's control inputs are transmitted over the Internet to cloud servers and processed there, and the resultant video stream is delivered back to the player's screen. Achieving high-quality video streaming and low-latency performance is essential for delivering a seamless cloud gaming experience. [114]

The end-to-end latency in cloud gaming comprises several factors that directly affect streaming quality and user experience. These components include transmission latency, instruction latency, encoding latency, packetisation latency, packet reception latency, decoding latency, and frame interval latency. Transmission latency refers to the time required for transmitting media and command streams between the client and the server. Cloud providers often use edge nodes to reduce network distances, ensuring that round-trip time (RTT) typically remains under 10 ms in optimal conditions, equivalent to 5 ms one-way latency. However, during high-demand scenarios where centralised scheduling connects users to more distant nodes, RTT may reach approximately 30 ms (15 ms one-way). Low transmission latency is crucial for maintaining synchronisation between user inputs and game visuals. [2]

Instruction latency is the delay between a user's input and its recognition by the rendering engine (e.g., Unity or Unreal Engine) on the server. Efficient inter-process communication between the streaming layer and rendering application can bypass operating system-level interaction, enabling virtualised operations. This approach ensures instruction latency is controlled within 1 ms. Minimising instruction latency enhances responsiveness, a key factor for maintaining immersive gameplay in action-intensive scenarios. Encoding latency refers to the delay incurred during the video encoding process. By leveraging GPU hardware acceleration and a zero-latency encoding strategy, texture data can be captured and encoded within the GPU, avoiding costly GPU-CPU data transfers. This reduces power consumption and keeps encoding latency under 2 ms. Optimised encoding directly contributes to higher video stream quality, enabling high-definition (HD) visuals without compromising performance.

Packetisation latency occurs during the encapsulation of encoded video into RTP packets before transmission. Using advanced congestion control strategies tailored for cloud gaming, servers can dynamically adjust network buffers to reduce packetisation delays to under 5 ms. Effective packetisation ensures minimal buffering while maintaining smooth video playback. Packet reception latency arises from the jitter buffer at the client side, which handles out-of-order packets, packet loss, and network jitter. A typical configuration buffers 1–2 frames, resulting in a latency of 20–30 ms. While this introduces minor delays, it stabilises the video stream and minimises visual artefacts such as stuttering or freezing, thereby enhancing overall video quality.

Decoding latency depends on the client device's hardware and software capabilities. For mid-range devices, software decoding of HD video typically incurs an average latency of 8 ms. While this latency is statistically consistent, further optimisations may be required for H5 platforms to improve performance. Efficient decoding ensures that high-resolution frames are displayed with minimal delay. Frame interval latency is determined by the frame rate and video encoding mechanisms. At 60 fps, the interval between consecutive frames is 20 ms, introducing a delay of 0–20 ms between command response and video encoding. This inherent delay cannot be eliminated due to the nature of the encoding process but is manageable within the overall latency budget.

For a 1080P 60fps cloud gaming scenario, the total latency range comfortably enables a sub-100 ms experience. Achieving this level of latency is critical for preserving high-quality video streaming in cloud gaming. Low latency ensures that player inputs feel instantaneous, reducing the perception of lag, which is particularly important in genres such as first-person shooters or racing games. [63] Furthermore, advanced encoding techniques and adaptive streaming ensure HD or even 4K visuals are delivered with minimal compression artefacts, maintaining clarity and visual fidelity even during complex scenes.

Cloud gaming's reliance on high-speed, low-latency streaming technologies underscores the importance of optimising each component in the end-to-end pipeline. [31] By balancing latency reduction and video quality enhancement, it is possible to deliver a gaming experience that is both visually immersive and highly responsive, meeting the expectations of modern players. Future research should focus on further reducing latency while exploring innovative streaming strategies for delivering ultra-high-definition content with minimal network or computational overhead.

4.3.2 White Noise in NARX

In the context of a Non-linear Autoregressive model with Exogenous inputs (NARX), "white noise" refers to a random signal or sequence typically used to represent uncorrelated, unpredictable variations or disturbances in a system. Understanding white noise in NARX involves analysing its statistical properties, role in the model, and its effects on performance. [85]

White noise is characterised by several properties. [87] Firstly, it has a zero mean, meaning the average value of the noise signal over time is zero, expressed mathematically as $E[w(t)] = 0$, where $w(t)$ is the noise at time t. Secondly, white noise has a constant power spectral density, signifying equal power across all frequency components. This characteristic gives it the analogy to white light, which contains all visible colours (frequencies). Thirdly, the values of white noise are uncorrelated, implying that each value is statistically independent of the others. For two different time points t_1 and t_2, the correlation satisfies $E[w(t_1) \cdot w(t_2)] = 0$ if $t_1 \neq t_2$. Additionally, in many cases, white noise is assumed to follow a Gaussian (normal) distribution, often referred to as Gaussian white noise. [13]

The role of white noise in NARX models is significant. A NARX model predicts the future output of a system based on its past outputs (e.g., $y(t-1)$, $y(t-2)$, ...), external inputs (e.g.,

$x(t), x(t-1), \ldots$), and a noise term, which is often represented as white noise $w(t)$. The general form of the NARX model can be expressed as:

$$y(t) = f(y(t-1), y(t-2), \ldots, x(t), x(t-1), \ldots) + w(t),$$

Where $f(\cdot)$ is a non-linear function approximating the system dynamics, and $w(t)$ accounts for stochastic components such as measurement noise, model uncertainties, or external disturbances.

White noise serves several key purposes in NARX models. It represents measurement noise, capturing random variations not explained by the deterministic system dynamics, such as sensor errors or unpredictable environmental factors. Because white noise has a zero mean, it introduces unbiased disturbances into the model. Over time, the positive and negative deviations introduced by white noise cancel each other out, ensuring no systematic bias in the predictions.

During training or validation of NARX models, residual analysis often involves checking if the residuals—the differences between predicted and actual outputs—behave like white noise. If the residuals exhibit correlation or patterns, this suggests the model has failed to fully capture the underlying dynamics. Excessive white noise in data, however, can make it difficult for the NARX model to distinguish between true system behaviour and random disturbances. In such cases, filtering techniques, such as low-pass filters, are applied to reduce noise while preserving essential system dynamics.

In simulations, white noise may be artificially introduced to test the robustness of a NARX model or simulate real-world conditions. This ensures the model's performance remains reliable even under stochastic disturbances.

4.3.3 Simulation Latency as White noise

Latency in cloud gaming introduces random variability into the system, stemming from network dynamics, hardware processing times, and environmental factors. These variabilities make precise latency measurement and modelling challenging. Instead of treating overall latency as a deterministic or directly measurable factor, its stochastic nature allows it to be treated as white noise in predictive models like Non-linear Autoregressive models with Exogenous inputs (NARX). This approach leverages the inherent statistical properties of white noise to account for latency effects without requiring granular data collection.

In a NARX model, white noise ϕ_t is a representation of the system's stochastic disturbances, which includes latency-induced variations. This random signal is defined by its zero mean, constant variance, and lack of temporal correlation, enabling it to capture the unpredictable fluctuations present in a cloud gaming environment. For instance, while specific components of latency—such as transmission or decoding delays—may vary based on conditions, the aggregate latency behaves as a random disturbance relative to the deterministic relationship between user inputs and video quality metrics like UAI-VMAF.

The advantage of modelling overall latency as white noise lies in its abstraction. Latency components in cloud gaming often interact in complex, non-linear ways, making direct measure-

ment or deterministic modelling infeasible. By incorporating ϕ_t as part of the NARX framework, the predictive model can effectively separate the deterministic input-output dynamics from random disturbances. This ensures that the system can generalise across various conditions, such as network jitter or hardware performance variability, without overfitting to specific latency patterns.

Importantly, treating latency as white noise aligns with practical considerations in cloud gaming. Ideal white noise does not exist in reality, but as long as the system's bandwidth significantly exceeds the effective spectral width of the noise, the approximation holds. This principle enables the use of NARX networks to predict video quality with high accuracy, even when latency variations occur across multiple dimensions of the streaming process. [25] For example, the noise sequence ϕ_t in the NARX model encapsulates not only transmission delays but also encoding and packetisation variations, providing a unified representation of latency effects.

This abstraction avoids the need for direct measurement of latency components, simplifying model implementation and enhancing scalability. As cloud gaming systems evolve, the diversity of network architectures and device capabilities increases, leading to greater heterogeneity in latency profiles. By assuming a white noise structure for these profiles, NARX models remain robust to changes in infrastructure or usage patterns, making them highly applicable in dynamic, real-world scenarios.

Finally, simulations using NARX networks have demonstrated their effectiveness in handling stochastic disturbances while preserving predictive accuracy for video quality metrics. By capturing the relationship between deterministic factors and video quality outputs and by treating overall latency as a stochastic input, NARX-based approaches ensure reliable predictions without exhaustive data collection or system-specific tuning. This methodology underscores the adaptability and efficiency of white noise modelling in cloud gaming contexts. Therefore, in this thesis, we reasonably treat the overall latency as white noise within the NARX model for UAI-VMAF in cloud gaming. This approach simplifies the modelling complexity by abstracting external interference factors, allowing for a deeper exploration of the relationship between UAI and VMAF and their mutual influences. By eliminating the need to model external disturbances explicitly, this method enables a more precise investigation into the determinants of video quality in cloud gaming. Furthermore, this abstraction facilitates the development of a more accurate predictive model for video quality improvement while significantly reducing reliance on detailed latency data. By adopting this strategy, the proposed NARX-based framework offers enhanced scalability and adaptability across diverse cloud gaming scenarios.

4.4 Evaluation Index

The purpose of this study is to predict the quality of cloud gaming streamed video; therefore, using UAI as input and regressing VMAF is our research objective. To evaluate the performance of the established models, this thesis defines the following four metrics to assess the regression

results of the models, namely

$$e_{mean} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i), \tag{4.1}$$

$$e_{max} = max|y_i - \hat{y}_i|, \tag{4.2}$$

$$e_{sum} = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{4.3}$$

$$e_{fit} = 100\% * \left(1 - \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y_i})^2}}\right), \tag{4.4}$$

$$i = 1, 2, \ldots N$$

In the above metrics, e_{mean} reflects the mean variation of the error between the model's calculated output and the actual output; e_{max} reflects the variation of the maximum value of the error between the model's calculated output and the actual output; e_{sum} reflects the variation of the total error between the model's calculated output and the actual output; and e_{fit} reflects the percentage of agreement between the calculated output and the actual output. These four metrics evaluate the regression performance of the model from different perspectives. The e_{mean} and e_{fit} show the overall performance between the model's calculated outputs and actual outputs; they determine whether the model can be used or not. If one of them is too low, that proves the model cannot be used to predict. The e_{max} shows individual samples' abnormal variation. In other words, for a whole period prediction, a few individual abnormal points could be generally acceptable. The e_{sum} shows the total error for a whole period prediction, which tells the user how much the prediction can be trusted.

4.5 Simulation of Time Series Models

Game A is a single-player action game with rapidly changing visual content and a high user behaviour index. We collected 20 minutes of UAI and VMAF data for this game. After processing, the sample raw data of the VMAF coefficient and UAI are shown in the following Table 4.1:

Next, we proceed with modelling and analysis. First, we plot the time-domain graph of UAI and VMAF as follows: Fig. 4.2 and Fig. 4.3:

No	UAI	VMAF
1	10	0.703805456
2	10	0.850330242
3	6	0.853323176
4	3	0.809064276
5	10	0.851081111
...	...	...

Table 4.1 UAI-VMAF Date Table Example

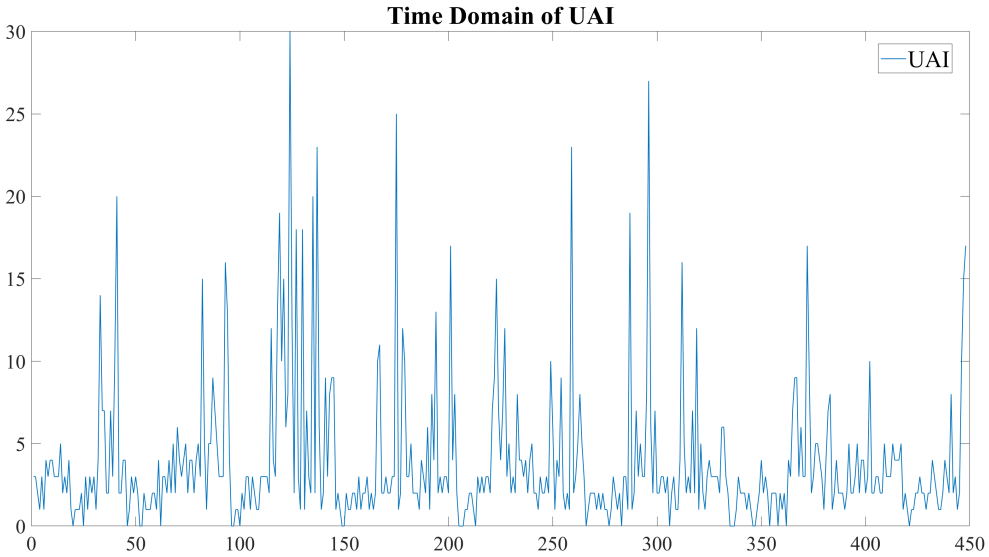


Fig. 4.2 Time domain of UAI for Simulation of Time Series Models

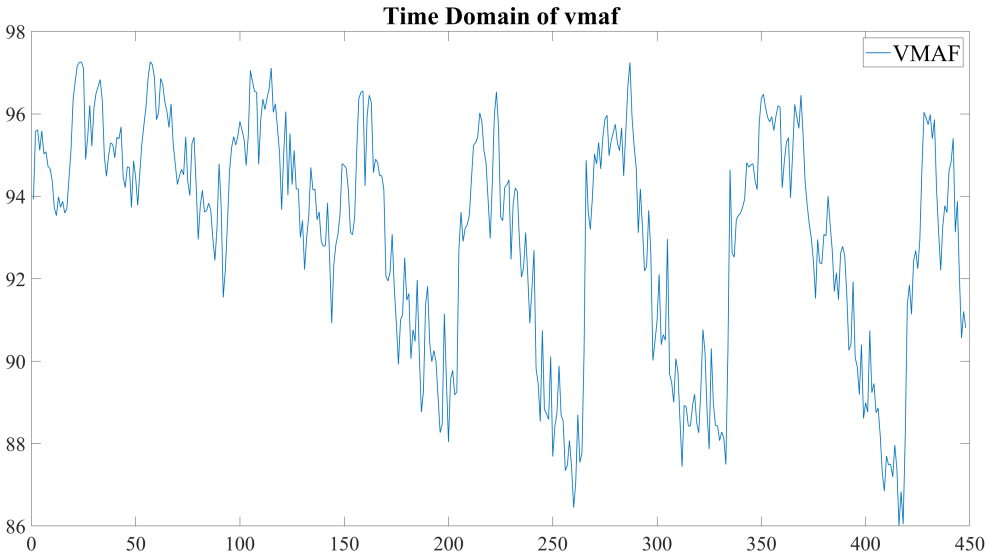


Fig. 4.3 Time domain of VMAF for Simulation of Time Series Models

ARX Simulation

To begin with, starting from the simplest linear model, we use ARX(2,2,1) to fit the relationship between UAI and VMAF. The modelling process is as follows: $data = iddata(y,u)$; $arx221 = arx(data, [2 \quad 2 \quad 1])$; $\hat{y} = sim(arx221, data)$; The simulation results are shown in the Fig. 4.4:

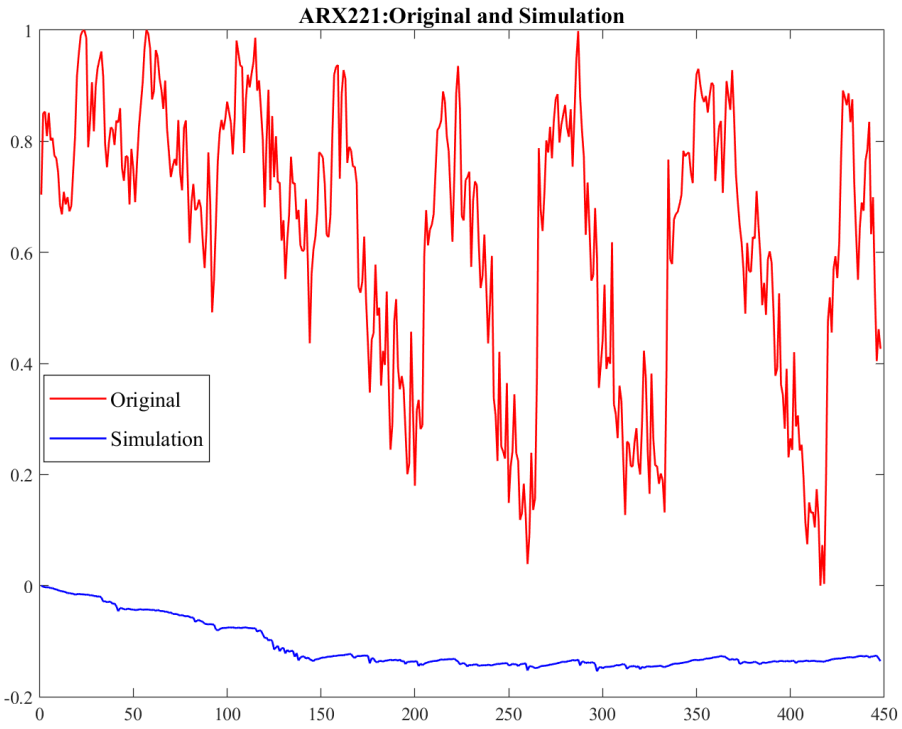


Fig. 4.4 ARX(2,2,1) Simulation Results

It is evident that the simulated data differs greatly from the original data, indicating that the $ARX(2,2,1)$ model cannot reflect the variation patterns of VMAF and is fundamentally inapplicable. Further improvement of the model is necessary.

Although linear model theory provides a method to determine the order of the model by calculating $\hat{\rho}_k$ and $\hat{\phi}_{kk}$, this approach is clearly not suitable for ARX and ARMAX models with external input sources and can only serve as a reference. Therefore, we estimate the order of the ARX model through experimental trial calculations.

Through program settings, we preset the model order to 40 (about 10 per cent of the total number of samples) and simulate ARX and $ARMAX$, respectively. The simulation results of ARX:Order=2-40 are as Fig. 4.5:

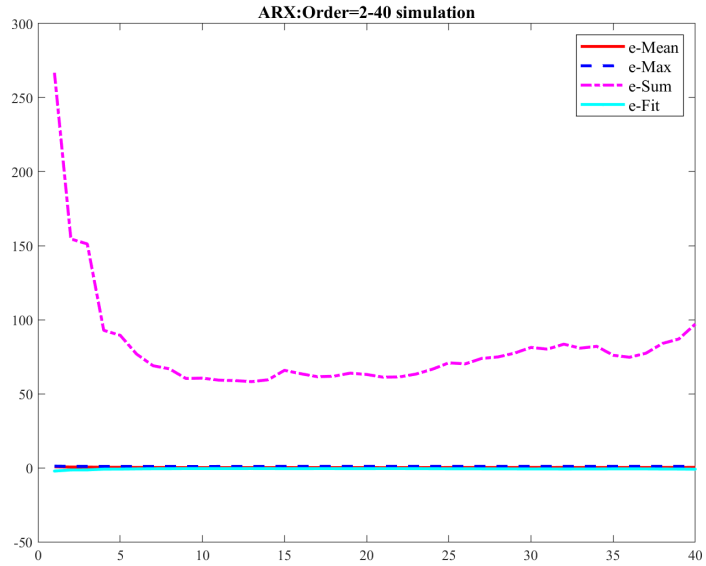


Fig. 4.5 ARX:Order=2~40 Simulation Results

In order to conduct an in-depth analysis, the curves of various performance indicators are plotted as Fig. 4.6, Fig. 4.7, Fig. 4.8, Fig. 4.9:

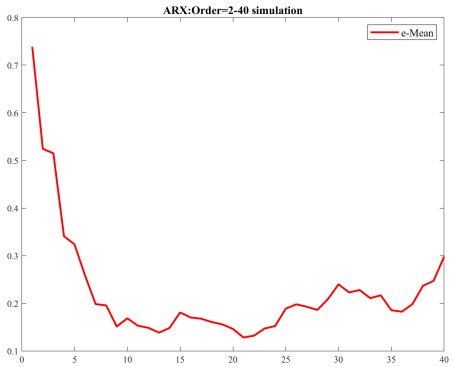


Fig. 4.6 $e_{mean}$of ARX:Order=2~40 Results

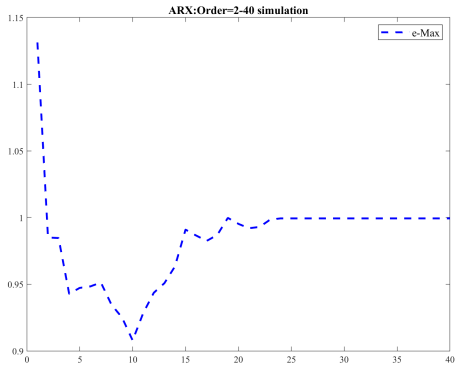


Fig. 4.7 e_{max} of ARX:Order=2~40 Results

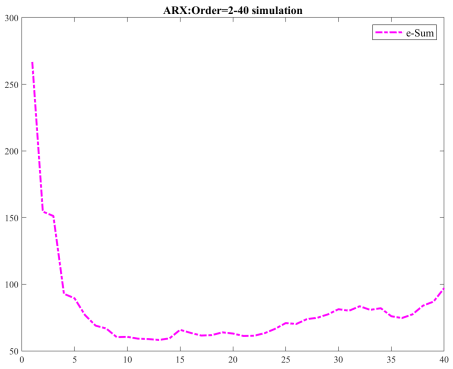


Fig. 4.8 e_{sum} of ARX:Order=2~40 Results

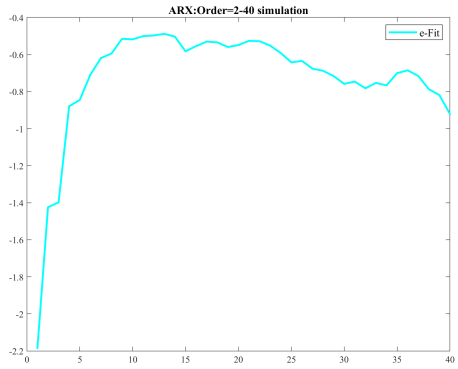


Fig. 4.9 e_{fit} of ARX:Order=2~40 Results

From the above simulation results, we can see that when the order of the ARX model increases to 10, the changes in various performance indicators tend to be stable. The four indicators $e_{mean}, e_{max}, e_{sum}, and e_{fit}$ are in the optimal value range when they are in the range of 10-15. After the 15th order, the performance indicators gradually begin to deteriorate. For the e_{fit} indicator, negative values appear, reflecting that the point-by-point agreement between the model calculation value and the actual value is very poor. There are many reasons for this phenomenon, such as, (i) The estimation algorithm failed to converge; (ii) The model was not estimated by minimising $|y - \hat{y}|$. Best Fit can be negative when you minimised 1-step-ahead prediction during the estimation, but validate using the simulated output $\hat{y}$; (iii) The validation data set was not pre-processed in the same way as the estimation data set; (iv) The two random sequences being regressed are in a highly non-linear relationship.

ARMAX Simulations

ARMAX: The simulation results of Order=2-40 are shown in the Fig. 4.10:

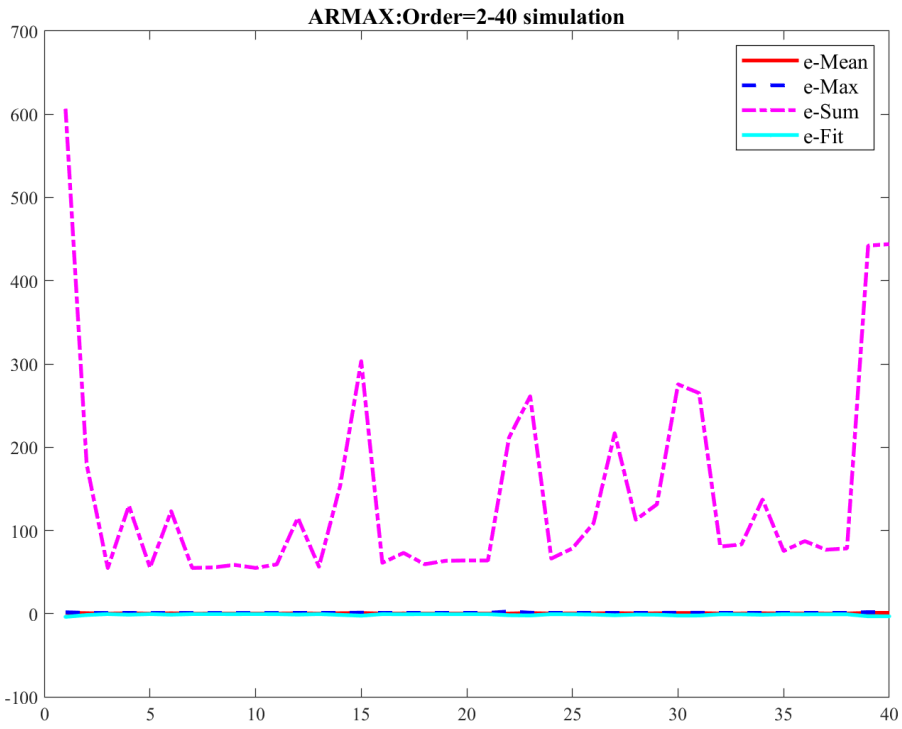


Fig. 4.10 ARMAX:Order=2~40 Simulation Results

Each curve is plotted separately as follows Fig. 4.11, Fig. 4.12, Fig. 4.13, Fig. 4.14:

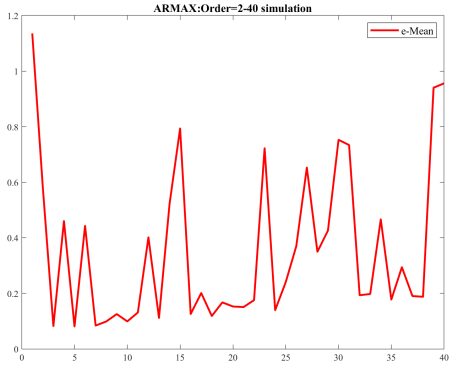


Fig. 4.11 e_{mean} of ARMAX:Order=2~40 Results

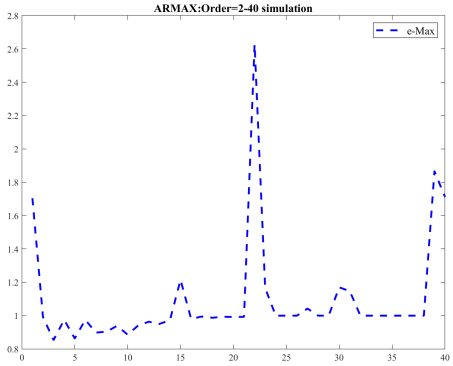


Fig. 4.12 e_{max} of ARMAX:Order=2~40 Results

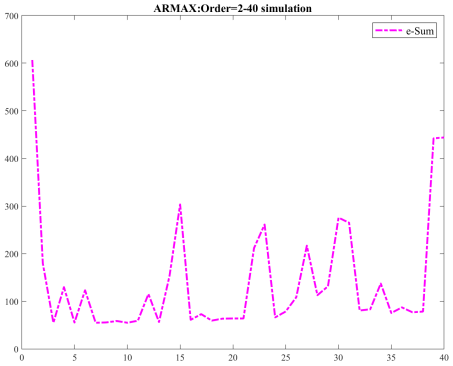


Fig. 4.13 e_{sum} of ARMAX:Order=2~40 Results

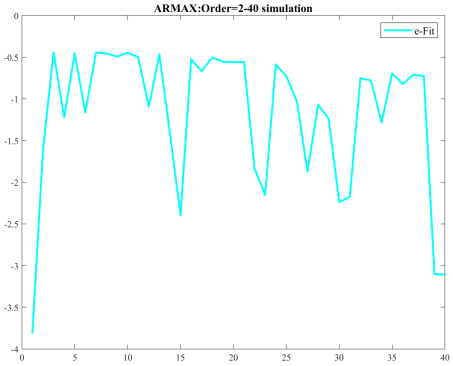


Fig. 4.14 e_{fit} of ARMAX:Order=2~40 Results

Based on the above simulation results, the four indicators e_{mean}, e_{max}, e_{sum}, and e_{fit} can reflect the relationship between the variation patterns of keyboard inputs and VMAF sequence data and the model order. As the order increases, the four indicators stabilise in an optimal range, and further increasing the order will gradually worsen the indicators. The ARX model is more valuable than the ARMAX model in reflecting the variation patterns of keyboard inputs and VMAF sequence data. High-order ARMAX models have poor stability.

During the research process, attempts were made to further increase the model order, but the stability of the algorithm deteriorated. When the order exceeded 50, the algorithm became unusable. This phenomenon indicates that the parameter estimation algorithm for linear models is based on the estimated values of the auto-correlation function and partial auto-correlation function of the sample data, denoted as $\hat{\rho}_k$ and $\hat{\phi}_{kk}$. To ensure $\rho_k \rightarrow \hat{\rho}_k$, the sample size N must be sufficiently large. At the same time, to ensure that $N-k$ is also sufficiently large, k cannot be too large. Theory suggests that generally, k should be approximately $\frac{1}{10}$ of the sample size, meaning that the model order should be between $(2-k)$ and determined by comprehensively considering the (4.1) indicators.

To more clearly analyse the model performance indicators, the four indicators of the ARX and ARMAX models are plotted separately for comparison, as shown below:

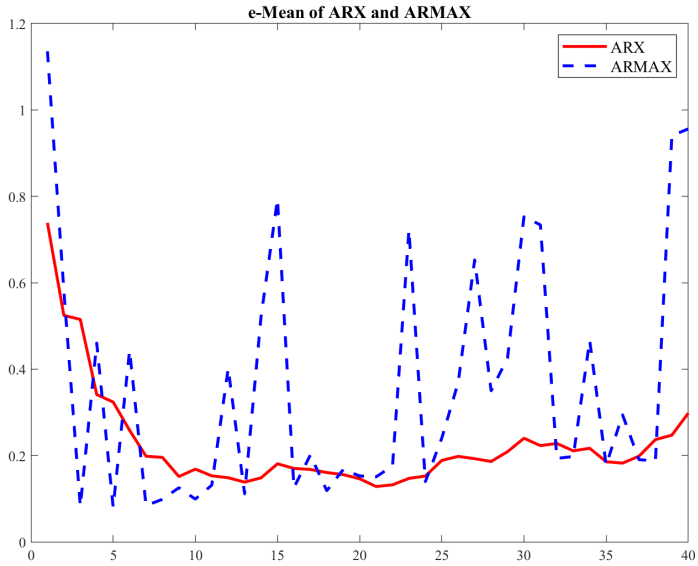


Fig. 4.15 e_{mean} of ARX & ARMAX

From Fig 4.15, we can see that the index e_{mean} gradually stabilises after the 10th order, the e_{mean} of the ARX model decreases slowly, but the fluctuation is small, and the e_{mean} of the ARMAX model decreases quickly, but the fluctuation is also large. Taking all factors into consideration, it is more reasonable to select the order of the linear model within the range of 10~15.

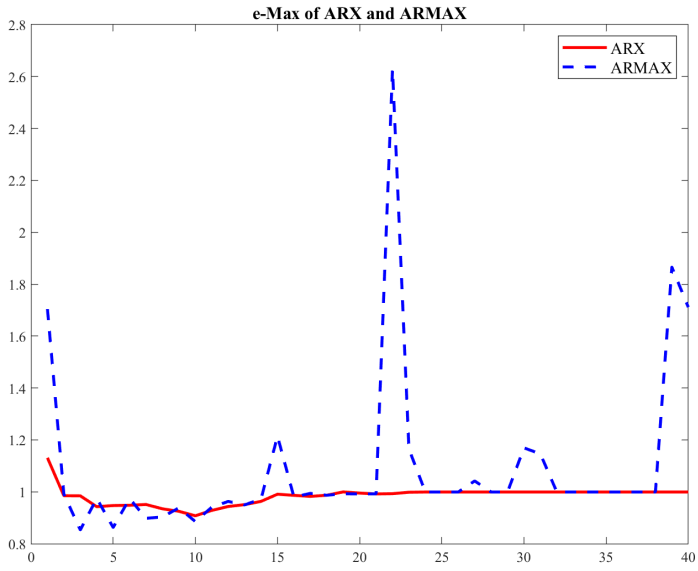


Fig. 4.16 e_{max} of ARX & ARMAX

From Fig 4.16, we can see that the decline rate of e_{max} of the ARMAX model is better than that of the ARX model, but its fluctuation is also larger. Taking all factors into consideration, it is more reasonable to choose the order of the model within the range of 10~20.

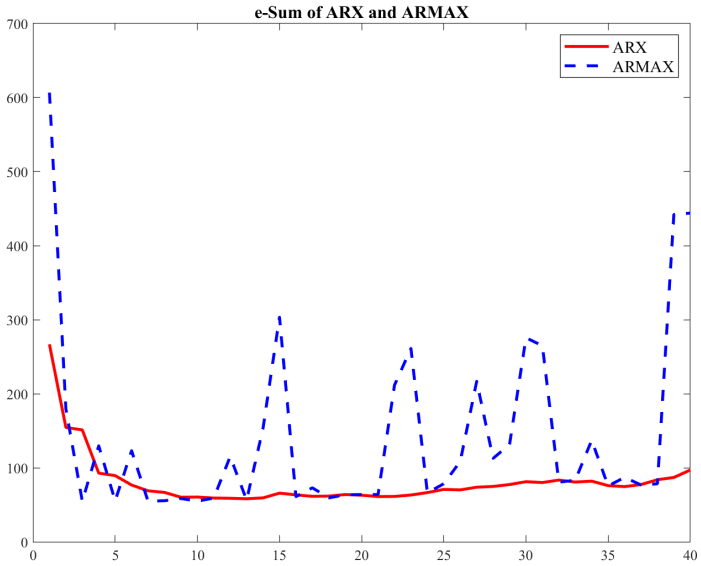


Fig. 4.17 e_{sum} of ARX & ARMAX

In Fig 4.17, the variation pattern of the indicator e_{sum} is similar to the above two. The order of the two models is more appropriately selected within the range of 10~15.

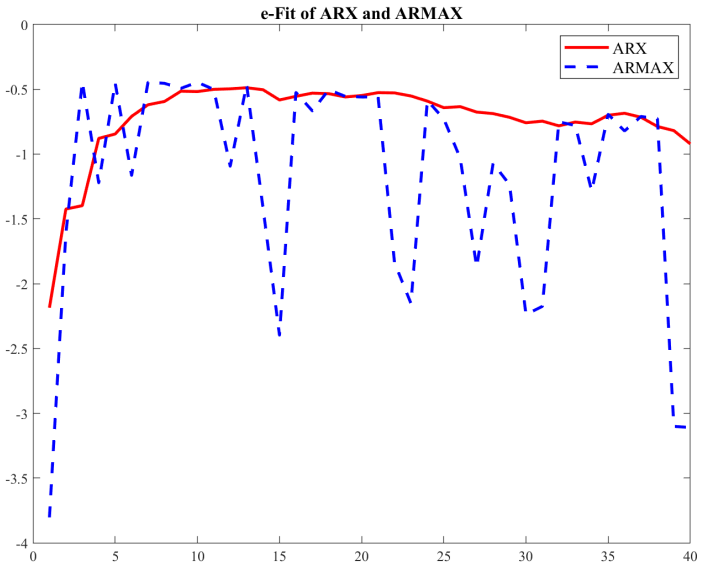


Fig. 4.18 e_{fit} of ARX & ARMAX

As can be seen from Fig 4.18, the indicator e_{fit} has a negative value, indicating that the point-by-point Fit of the two models is not high.

Based on the above analysis and the computational efficiency factors of the ARX and ARMAX models, the 12th order can be used as the reference order of the linear model.

All the above simulations prove that ARX/ARMAX models can not complete the task of predicting VMAF through UAI. This illustrates that the relations between UAI and VMAF are strong non-linear instead of linear, which no longer results in the failure of the linear model.

However, it also obviously demonstrated the causality between UAI and VMAF, as well as their dynamic auto-regressive characteristics. This provides the theoretical essentials for us to employ the NARX to analyse the UAI and VMAF. Furthermore, it also proves that the key step for predicting VMAF through UAI is capturing the non-linear map relation of UAI/VMAF by utilising the output feedback and delay mechanism of RNN. In the following, we will launch the NARX to simulate multiple cloud game datasets.

4.6 Simulation of NARX

To verify the effectiveness of NARX modelling, in this section, we conduct simulation experiments using 5 collected game datasets. The Case 1 dataset is consistent with the one used in the linear model simulation in the previous section. We use NARX to simulate it as a comparison and analyse the delay parameter settings of NARX with this dataset. The Case 2 and Case 3 datasets come from action games, where the game content changes and video features are significantly different. These two datasets use different data compression rates to verify the wide applicability of NARX modelling. The Case 4 and Case 5 datasets are collected from other types of games, and their game content changes and video features are different from the previous datasets. The presented UAI/VMAF relationship is more complex. We use data augmentation techniques to compensate for the poor model simulation effect caused by defects in the data samples, proving that when using NARX to model and analyse the UAI/VMAF relationship, it is necessary to use some data augmentation techniques in a timely manner according to the characteristics of the original dataset.

4.6.1 Matlab and NARX

This section discusses the NARX network structure and other key hyper-parameters, except for those already covered in Chapter 3.

Structure

A NARX network object (with two layers as shown in Fig 4.19) is implemented in the programme; it serves as an example of a two-layer feedforward network used for approximation purposes. [75]

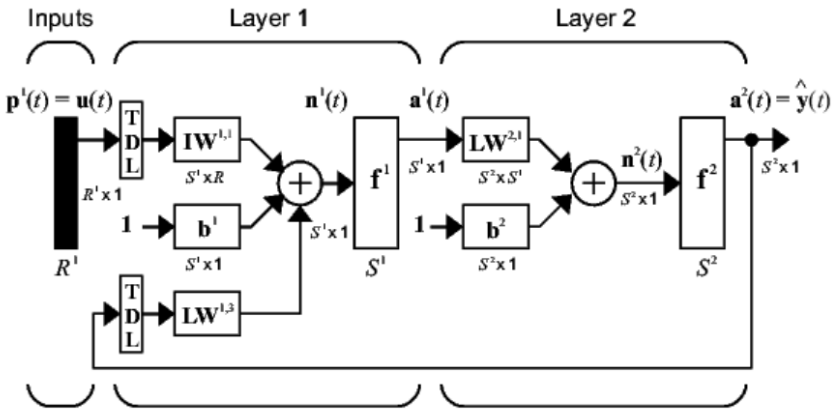


Fig. 4.19 Example NARX Network [147]

If the data relationship to be approximated is relatively simple, a single-layer network may suffice; conversely, if the relationship is more complex, three or more layers may be constructed. In this study, a two-layer configuration with ten neurons in each layer serves as the fundamental setup. This configuration may be adjusted according to the specific requirements of different game cases.

Delay Analysis

Utilising a NARX network to model the relationship between UAI and VMAF involves employing a feedforward network to realise the non-linear mapping from UAI to VMAF. The dynamic feedback mechanisms inherent in the NARX network encapsulate the autoregressive characteristics of the random sequence. Consequently, the design of delays within the NARX network becomes a critical consideration, and analysing the time series delays is crucial for accurately modelling the relationship between UAI and VMAF.

From the perspective of cloud gaming applications, the visual quality at a player's current time t (expressed as the VMAF score $\mathrm{VMAF}(t)$) depends on the player's User Activity Index (UAI) at time t and the UAI values at the preceding N time steps, namely $\mathrm{UAI}(t-1), \mathrm{UAI}(t-2), \ldots, \mathrm{UAI}(t-N)$. As time t progresses, these previous UAI values also shift forward, forming a time window of width N. Therefore, the delay design can also be referred to as the *sliding window design*—that is, designing a window $N(t)$ that evolves over time to accurately reflect changes in $\mathrm{UAI}(t)$.

In this process, the *time-step* is a critical parameter representing the number of consecutive frames displayed on the player's screen. It is evident that different types of games have varying numbers of consecutive frames. For action games, the time-step should be relatively large due to rapid changes in visuals, whereas for music games, which feature slower visual transitions, the time-step can be smaller. In this study, Case 1 to Case 5 corresponds to an FPS game, an Action 2D Game, an Action 3D Game, an RPG Game, and a Music Game, respectively, with time-step values ranging from 0 to 60.

Of course, besides being influenced by the game type, both the sliding window and time-step are directly affected by factors such as hardware performance, network environment, and various

forms of noise (e.g., programme response speed, image compression rate). Clearly, accurately designing the sliding window and time-step is highly challenging and can only be determined through simulation experiments tailored to different game types.

Based on the delay analysis of UAI and VMAF in the previous section, we first use the correlation function analysis method to estimate the delay of the collected sample data sequence. Then, using this estimate as a reference, we simulate the impact of delay on the regression performance of the NARX network. The correlation analysis steps for the UAI and VMAF sequences are illustrated in the following Fig 4.20:

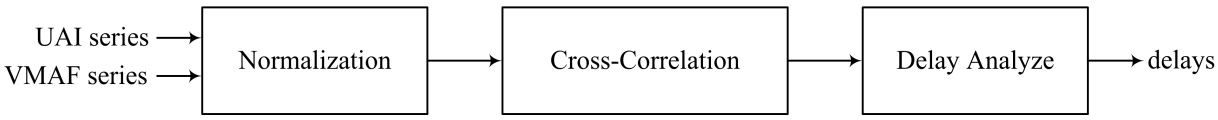


Fig. 4.20 Cross Correlation Method

Objective Function

NARX is a type of dynamic network; however, its weight adjustments are accomplished through an error backpropagation mechanism (as shown in Figure 3.3). All hyper-parameters of the NARX network are configured to minimise the objective function. The objective function, presented in Equation 3.40, is based on Equations 3.41 and 3.42.

For example, suppose we have designed a NARX network with a total of 20 neurons, and there are 500 sample data points available for training. When the n-th sample ($n = 1 \ldots 500$) is input, the output generated at the j-th neuron ($j = 1 \ldots 20$) is denoted as $d_j(n)$. At this time, the actual output of this neuron is denoted as $y_j(n)$. Thus, the error of this neuron is denoted as $e_j(n)$, that is,

$$e_j(n) = d_j(n) - y_j(n) \tag{4.5}$$

Let the square error of the neuron be $\frac{1}{2}e_j^2(n)$, then the total square error at the output of the network is:

$$E(n) = \frac{1}{2}\sum_{j=1}^{20} e_j^2(n) \tag{4.6}$$

Assuming that the total number of training samples is 500, the average total error generated after all samples are input into the network is:

$$E_{AV} = \frac{1}{500}\sum_{n=1}^{500} E(n) \tag{4.7}$$

This constitutes the first round of training. If the average error E_{AV} reaches the training target, the training is terminated, and the weights at this time are considered the optimal weights of the network. If E_{AV} does not meet the target, the next round of training begins, and this cycle continues until E_{AV} satisfies the requirement. However, training the network with E_{AV} as the objective merely ensures the optimal fitting of the input/output relations of the training sample

data under a specific hyper-parameter configuration. To determine whether this network is appropriate for modelling the UAI/VMAF relationship, it must be evaluated using the performance metrics defined by Equations 4.4, 4.4, 4.4 and 4.4.

4.6.2 NARX Case 1 - FPS Game

For comparison with the linear model, we still use the data from Game A for NARX modelling.

First, set up a relatively simple network structure, and select the parameters as shown in the following table 4.2:

Parameter	1	2	3	4	5
	1^{th} layer	2^{th} layer	Start delay	End delay	epoch
Value	10	10	1	32	1000

Table 4.2 Parameter Table for NARX Case 1

Parameter	6	7	8	9	10
	TFi	BTF	BLF	PF	others
Value	tansig/purelin	trainlm	learngdm	mse	Searched

Table 4.3 Parameter Table for NARX Case 1 Continue

The code is as follows:

Algorithm 1 Training and Simulating NARX Network Program

```
narx_net ← newnarxsp(data, input_delay, output_delay, [20], TFi, BTF, BLF, PF)
Set max_epochs to 1000
narx_net ← train(narx_net, data, delay, max_epochs)
y_sim ← sim(narx_net, data, delay)
```

The results are shown in the following figure:

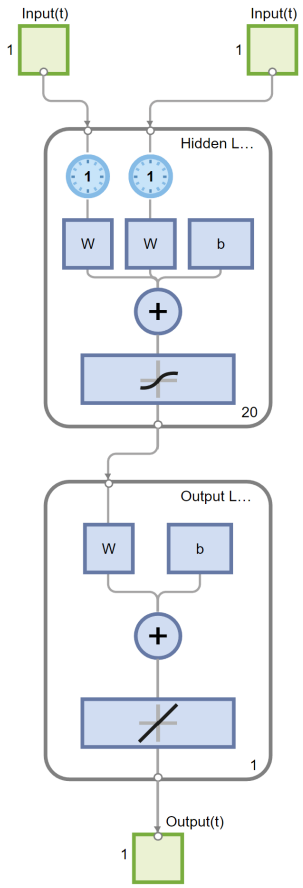


Fig. 4.21 Network Structure for NARX Case 1

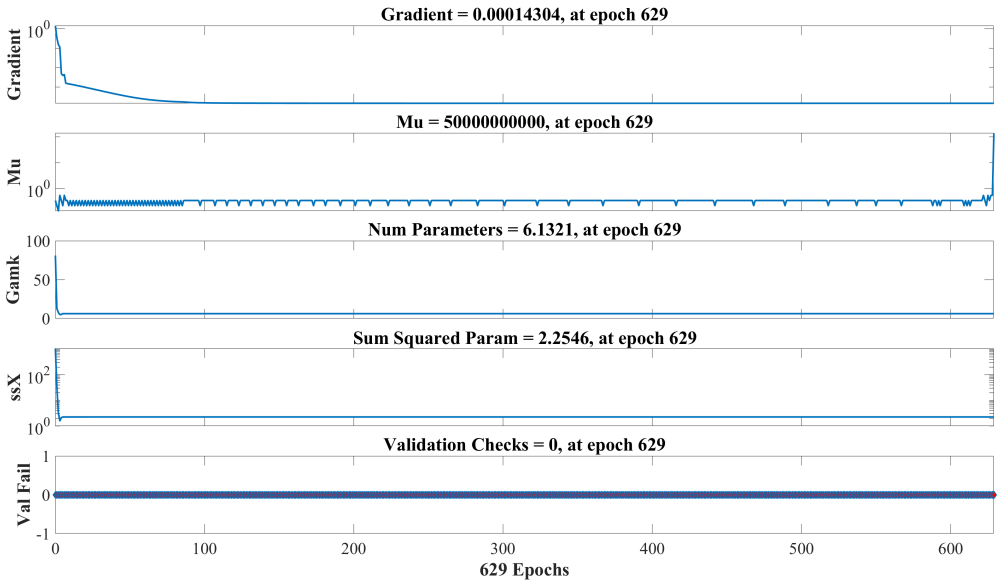


Fig. 4.22 Training States for NARX Case 1

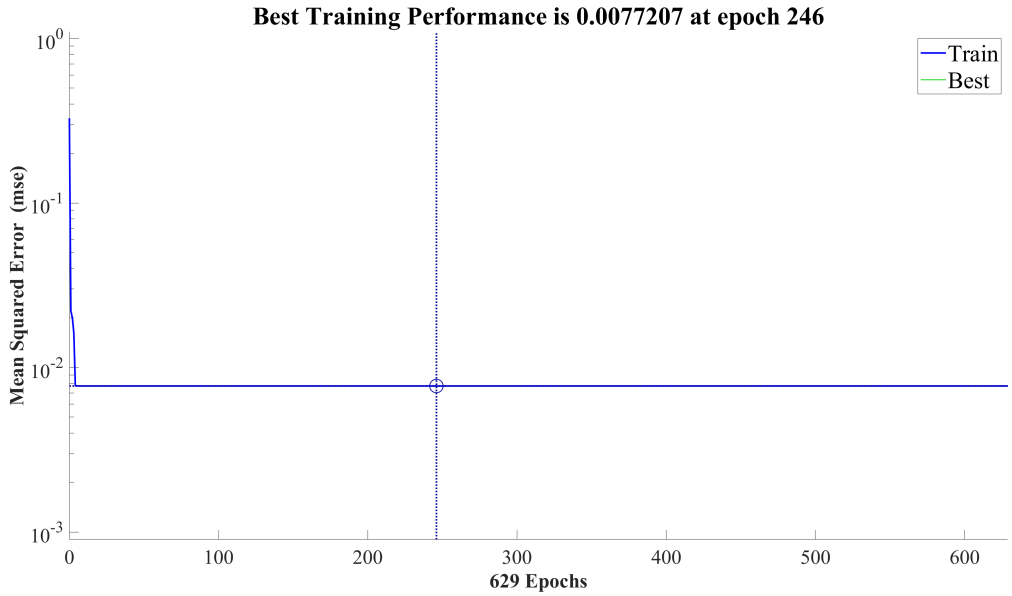


Fig. 4.23 Performance for NARX Case 1

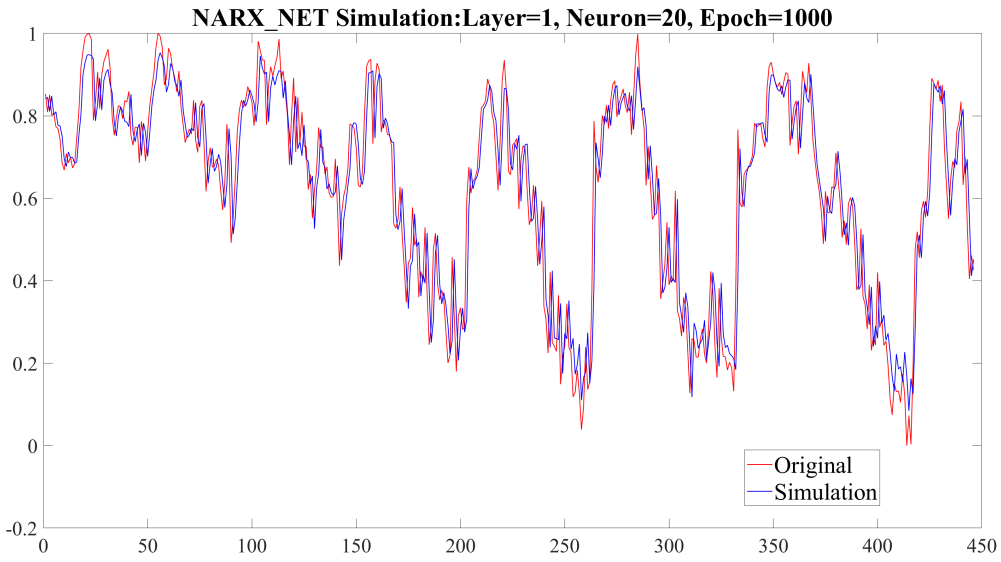


Fig. 4.24 NARX_NET Simulation Result for NARX Case 1

The calculated results for the performance metrics of the model are: narx_net: layer=[20], delay=1, epoch=1000, $e_{mean} = -6.3148 \times 10^{-6}$, $e_{max} = 0.4080$, $e_{sum} = 3.4512$, $e_{fit} = 63.76\%$, Analysing the simulation results above, the training of the network was terminated when the net.trainParam.mu reached its maximum value, and the network training stopped at the 961st epoch (with the preset maximum epoch set at 1000). At the same time, it is possible to retrain the network by adjusting net.trainParam.mu and other training parameters, the overall performance metrics of the model were not satisfactory. Only the e_{mean} metric was relatively ideal, while the other three metrics were poor. Further error analysis is illustrated in the following graph:

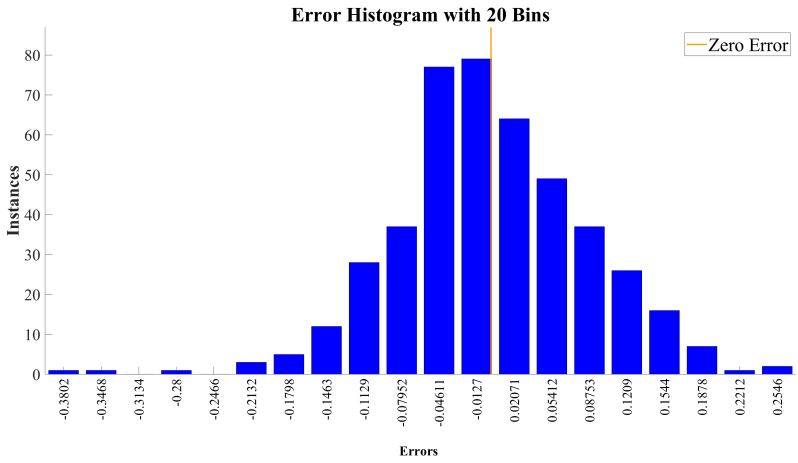


Fig. 4.25 Histogram for NARX Case 1 Simulation

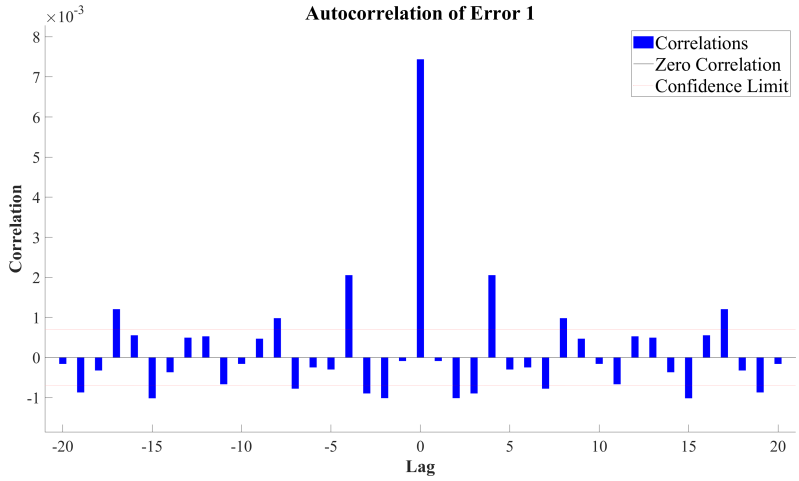


Fig. 4.26 Error correction for NARX Case 1 Simulation

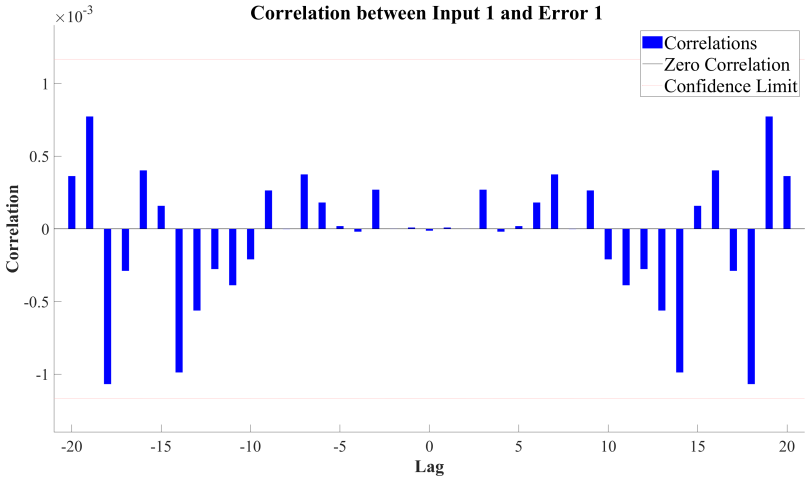


Fig. 4.27 Error Input for NARX Case 1 Simulation

From the error distribution histogram, it can be seen that the proportion of instances with an absolute error greater than 0.1129 is very high. The proportion of instances that fall outside

the confidence interval in the error autocorrelation histogram is also very high. The input-error histogram exhibits multi-peak characteristics, indicating the presence of periodic trends between input-error.

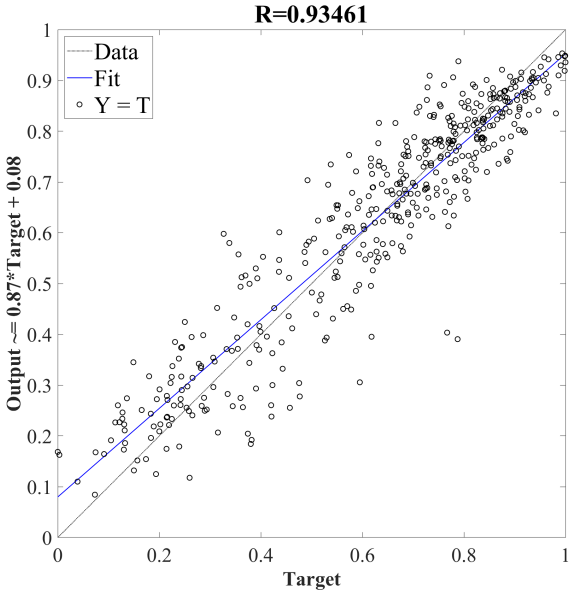


Fig. 4.28 Regression for NARX Case 1 Simulation

The regression plot shows that although R=0.9364, the data points are scattered over a wide range and are not highly concentrated. The response plot reveals that there are generally large errors between the Outputs and Targets across the entire dataset.

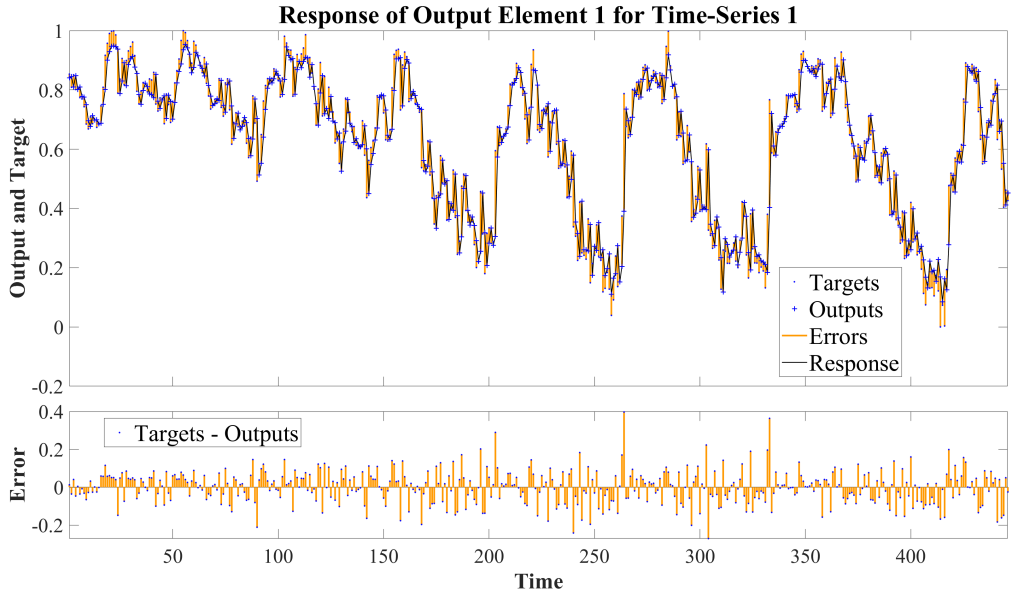


Fig. 4.29 Response for NARX Case 1 Simulation

In summary, the calculated model performance metrics are consistent with the error analysis results, indicating that the preliminary simulation results are not acceptable. It should be noted that the delay analysis in the previous chapter pointed out that delay settings are crucial for NARX

networks and are key parameters affecting the capture of dynamic relationships in UAI-VMAF sequence data. However, the correct delay settings were not used in this preliminary simulation. Therefore, dedicated simulation experiments on delays will be conducted below.

Delay Analysis for NARX Case 1

Using the delay_estimator.m script program written in this thesis, the delay analysis of UAI and VMAF for Game 1 yields a result of 16. Taking this delay value as a reference, we test the impact of delay values in the range of [1~32] on the regression performance. The network structure parameters are as follows Table 4.4 and 4.5:

Parameter	1	2	3	4	5
	1^{th} layer	2^{th} layer	Start delay	End delay	epoch
Value	10	10	1	32	1000

Table 4.4 Parameter Table for NARX Case 1 including Delay Analysis

Parameter	6	7	8	9	10
	TFi	BTF	BLF	PF	others
Value	tansig/purelin	trainlm	learngdm	mse	Searched

Table 4.5 Parameter Table for NARX Case 1 including Delay Analysis Continue

Network training and convergence indicators are detailed in Appendix 3. The table of network regression performance for different delay values is as follows Table 4.6:

Param / **Delays**	$\mathbf{e_{mean}}$	$\mathbf{e_{max}}$	$\mathbf{e_{sum}}$	$\mathbf{e_{fit}}$
1	-3.4754×10^{-6}	4.0841×10^{-1}	3.4499	63.77%
2	1.1489×10^{-5}	3.9689×10^{-1}	3.3182	64.43%
3	2.9390×10^{-6}	1.8238×10^{-1}	1.0990	79.51%
4	-5.6526×10^{-6}	1.4566×10^{-1}	8.7861×10^{-1}	81.67%
5	-2.5070×10^{-6}	1.3403×10^{-1}	7.4965×10^{-1}	83.05%
6	-7.4864×10^{-5}	2.1986×10^{-1}	1.6317	74.98%
7	-8.3266×10^{-6}	2.2296×10^{-1}	1.4692	76.24%
8	3.1386×10^{-5}	1.9716×10^{-1}	1.2517	78.06%
9	8.6684×10^{-6}	1.7703×10^{-1}	1.2770	77.83%
10	5.8188×10^{-6}	1.9699×10^{-1}	8.2781×10^{-1}	82.15%
11	-1.3485×10^{-5}	1.5099×10^{-1}	5.3732×10^{-1}	85.62%
12	4.4926×10^{-4}	1.1299×10^{-1}	4.2069×10^{-1}	87.27%

Delays \ Param	e_{mean}	e_{max}	e_{sum}	e_{fit}
13	1.2762×10^{-5}	1.5709×10^{-1}	5.8137×10^{-1}	85.04%
14	3.8489×10^{-6}	1.1514×10^{-1}	4.1035×10^{-1}	87.43%
15	1.3870×10^{-4}	1.2553×10^{-1}	3.2414×10^{-1}	88.82%
16	1.1368×10^{-3}	6.2146×10^{-1}	2.5949×10^{1}	0.00%
17	2.1781×10^{-6}	9.2710×10^{-2}	1.5280×10^{-1}	92.33%
18	4.7098×10^{-6}	8.6263×10^{-2}	1.1112×10^{-1}	93.45%
19	8.2776×10^{-7}	9.9164×10^{-2}	8.5787×10^{-2}	94.24%
20	4.3734×10^{-5}	1.4197×10^{-2}	4.0605×10^{-3}	98.75%
21	-1.0706×10^{-8}	1.6025×10^{-7}	4.0484×10^{-13}	100.00%
22	-6.9990×10^{-10}	1.2735×10^{-7}	4.7152×10^{-13}	100.00%
23	1.1901×10^{-3}	6.1731×10^{-1}	5.413×10^{1}	0.00%
24	1.1980×10^{-3}	6.1640×10^{-1}	2.5268×10^{1}	0.00%
25	4.9033×10^{-10}	1.5526×10^{-7}	4.1686×10^{-13}	100.00%
26	1.2147×10^{-3}	6.1510×10^{-1}	2.5102×10^{1}	0.00%
27	-2.9299×10^{-9}	4.3318×10^{-7}	3.7180×10^{-12}	100.00%
28	1.2326×10^{-3}	6.1387×10^{-1}	2.4969×10^{1}	0.00%
29	1.1980×10^{-3}	6.1342×10^{-1}	2.4928×10^{1}	0.00%
30	1.2461×10^{-3}	6.1269×10^{-1}	2.4846×10^{1}	0.00%
31	4.2324×10^{-9}	1.4965×10^{-6}	1.2473×10^{-11}	100.00%
32	1.1453×10^{-8}	6.8302×10^{-8}	1.6592×10^{-13}	100.00%

Table 4.6 Table of network regression performance

The performance metrics for network regression are graphed as follows:

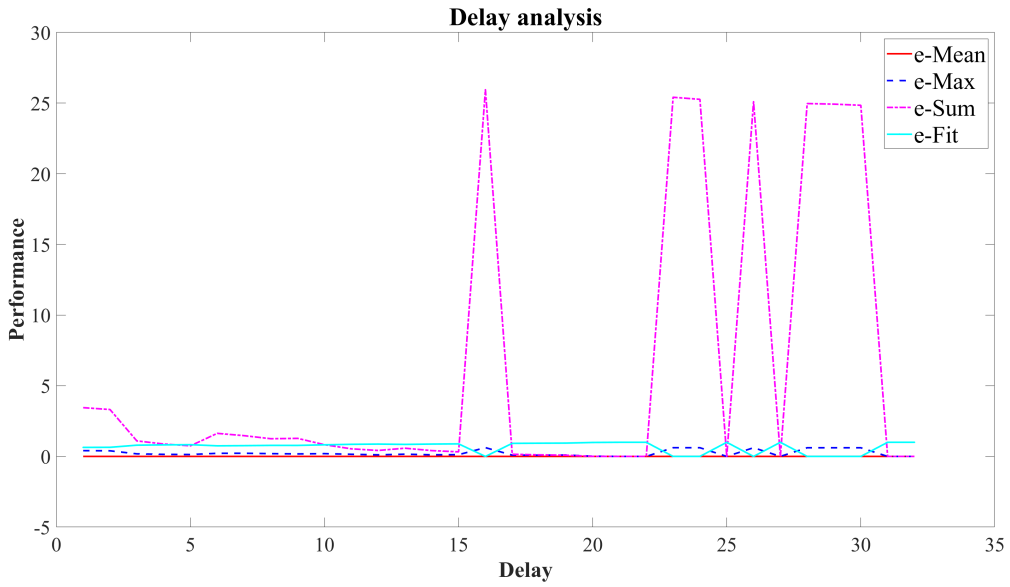


Fig. 4.30 Delay analysis

The data in the network training performance table shows that when the delay value is set to 1,2,3,4,16,23,24,26,28,29,30, network training terminates due to mu reaching its maximum value. The best network performance metric among these is only 1.9788×10^{-3}, which is far from the preset target of 0.0000, indicating that the network training algorithm is unstable and training terminates prematurely. Excessively large mu values may also cause network training to fall into local minima.

When the delay value is set to 22,25,27, network training terminates due to the gradient reaching its minimum value. The network performance metric reaches 8.8313×10^{-15} or higher, indicating that the network performance metric is in a relatively flat decreasing interval, and it is difficult to achieve significant improvement. In this case, although the network performance metric is close to optimal, the network's generalisation performance is poor, and there is no value in further adjusting the network.

For the remaining 18 delay value settings, network training terminates when the maximum number of iterations is reached. When the delay value is set to 21,31,32, the network performance metric reaches 2.9912×10^{-14} or higher, and both mu and gradient indicators are within normal ranges, indicating stable network training and optimal performance metrics. Of course, these three delay values also result in the longest training time for the network.

The data in the network regression performance table shows that the trend of the four regression performance metrics is the same as the trend of the network training performance metrics.

Based on the above analysis, the following conclusions can be drawn: (i) For this game dataset, using NARX modelling, the simulation data effect is good and can satisfy the analysis of UAI and VMAF sequence data; (ii) The delay value calculated based on the principle of the cross-correlation function for the UAI and VMAF sequences is only a reference. The actual situation is much more complex and closely related to the training settings of the NARX network; (iii) When the delay value is set below 9, the network regression performance metric e-fit is below 80 per cent, and when the delay value is set above 17, the e-fit metric reaches above 90 per cent; (iv) The greater the network training cost (epoch, time), the better the network regression performance.

It is worth noting that the training performance of neural networks is affected by the initialisation of network parameters. With the same parameter settings, the randomness of initialisation parameters leads to different results for multiple training and simulation runs. Therefore, once the network parameters are determined, multiple training and simulation runs are required to obtain stable network objects and simulation results.

Based on the delay analysis above, setting delay = 22 and without altering other parameters, the simulation results are as follows:

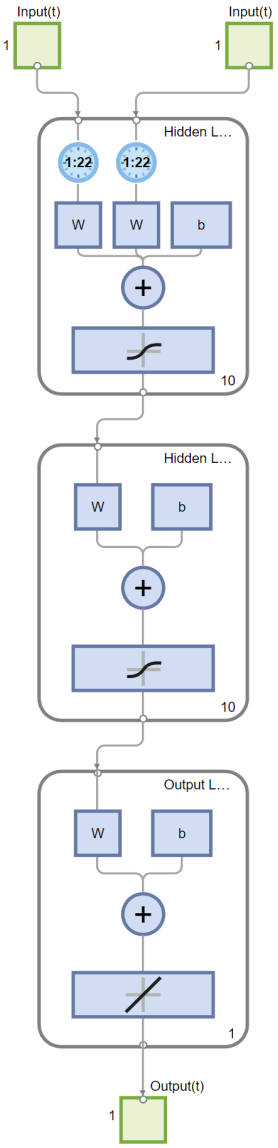


Fig. 4.31 Network Structure

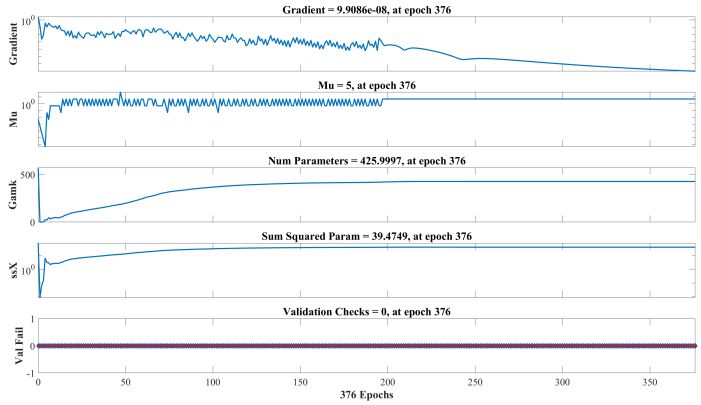


Fig. 4.32 Training States

Fig. 4.33 Performance

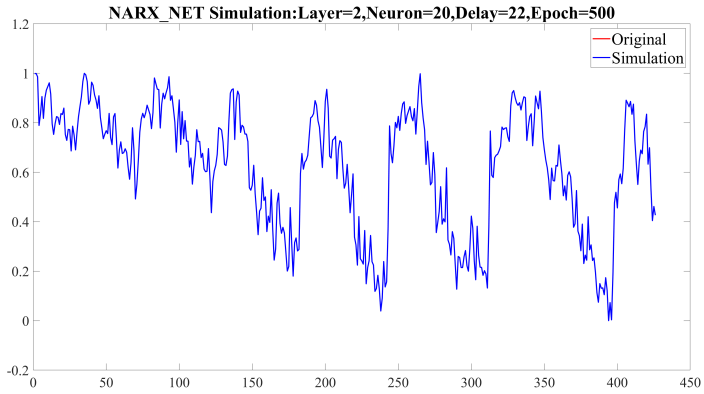


Fig. 4.34 Simulation

The calculated performance metrics for the NARX model are: narx_net: layer=[10 10], delay=22, epoch=500,$e_{mean} = -2.0242 \times 10^{-9}$, $e_{max} = 1.3373 \times 10^{-7}$, $e_{sum} = 2.6931 \times 10^{-13}$, $e_{fit} = 99.99\%$

Clearly, this result is significantly better than the previous outcomes shown.

To verify the accuracy of the simulation results, further analysis of the errors is conducted: First, a histogram of the errors between the model's simulated output values and the actual values is constructed.

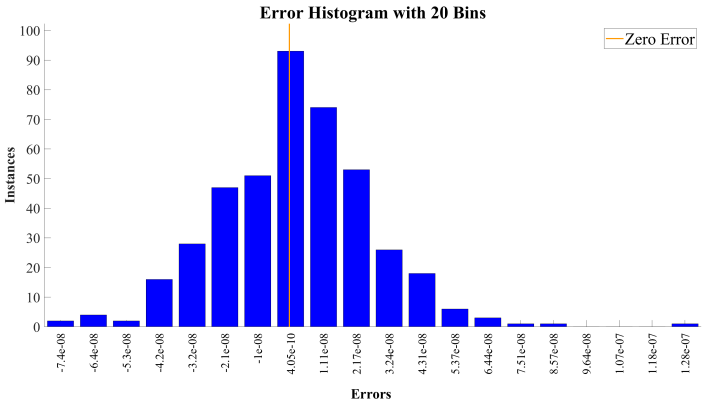


Fig. 4.35 Error Histogram

The error histogram displays the distribution of errors. The graph shows a single peak centred in the middle, with a symmetrical and even distribution on both sides and no obvious outliers at the tails. The errors are mainly concentrated within the interval $[-4.2\times10^{-8}, 4.31\times10^{-8}]$, with a maximum absolute error value of 1.28×10^{-7} and a zero-error-bin of 4.05×10^{-10}, indicating a reasonable error distribution and excellent model simulation performance.

The autocorrelation function graph of the errors is as follows:

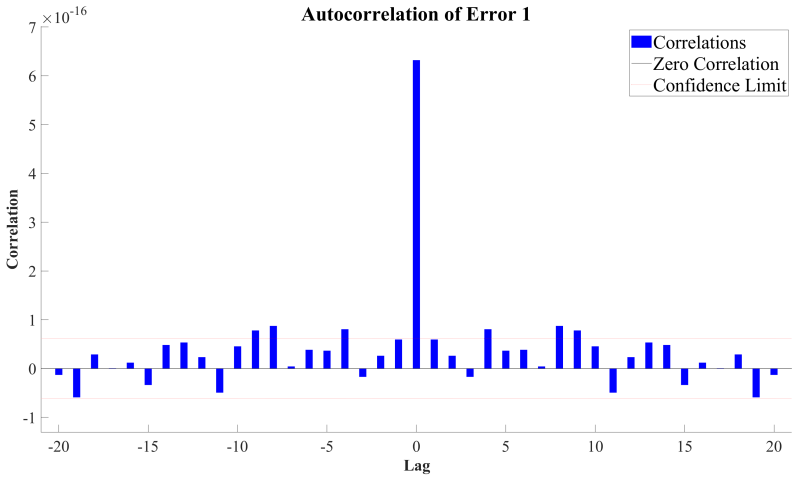


Fig. 4.36 Autocorrelation

Calculating the autocorrelation function is done to discover the similarity or dependence of the error sequence with itself at different time points, as well as any potential periodic trends. According to the graph, the autocorrelation coefficient of the error between the simulated output and the actual value is 10^{-3}, maximised at a lag of 0 and almost always within the 95% confidence interval in other cases. This indicates that the simulation error sequence is a random process with no correlation.

Input-error correlation function graph as Fig. 4.37:

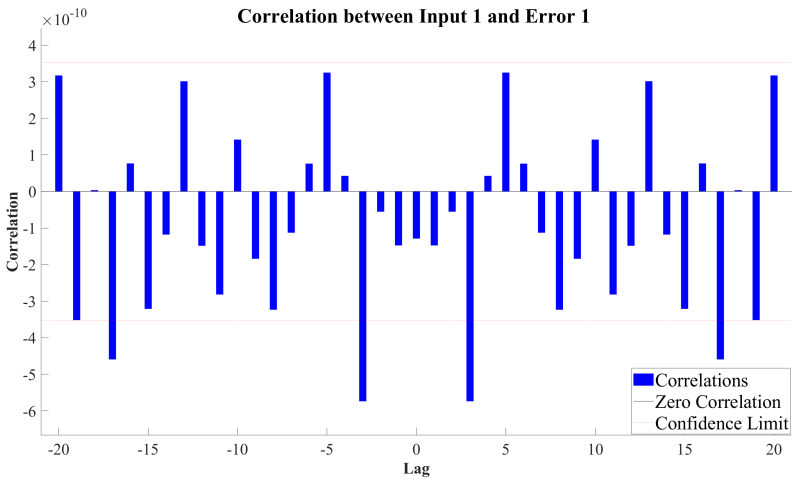


Fig. 4.37 Input-error correlation

Calculating the correlation function between the model input and error is done to analyse the relationship between the model input and error and to determine the quality of the simulation results. The lower the correlation, the better the results and the correlation coefficient values should be distributed around zero. The graph shows that the correlation between the model input and error is very low, with the maximum absolute value of the correlation coefficient not exceeding 6×10^{-10}, uniformly distributed, with no abnormal peaks, and all values within the consistency interval.

Combining the results of the above error distribution, error autocorrelation analysis, and input-error correlation analysis indicates that the simulation of the constructed NARX model is completely reliable.

Regression performance analysis, Regression and Response graphs are as follows:

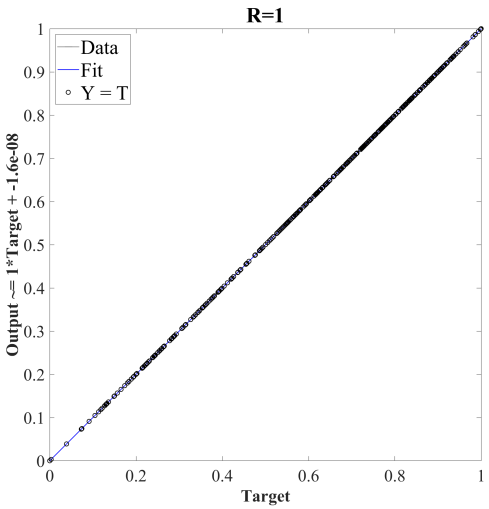


Fig. 4.38 Regression performance

The above graph shows the fitting effect between the simulated output and the actual values. The error points are evenly scattered along the 45° fitting line with very small deviations, and $R \approx 1$.

Fig. 4.39 Response

The above graph shows the correspondence between the model output and the actual VMAF sequence. The error for the entire sequence lies within the interval $[-1.5\times10^{-7}, 1.0\times10^{-7}]$, and the Outputs respond very well to the Targets.

Summary

This section demonstrates that using the NARX network to study the relationship between UAI and VMAF sequences is feasible. The non-linear mapping capabilities of feedforward networks provide effective support for the non-linear relationship between UAI and VMAF sequences, while the NARX network with dynamic feedback can effectively utilise the historical effects and I/O delay characteristics of time series data. Due to the non-linear characteristics of the relationship between UAI and VMAF sequences, linear model methods are insufficient to capture the complexity of this relationship, and the advantages of recurrent neural networks are thus apparent.

However, the quality of cloud gaming streaming video is influenced by various factors. Even with the same network environment and game content, user behaviour and operations will lead to changes in UAI and VMAF sequence data, ultimately affecting the accuracy of using UAI to predict VMAF. A model learned and trained using a single dataset lacks universality. Therefore, in the next section, this thesis will use another action game dataset for simulation research.

4.6.3 NARX Case 2 - Action Game 2D

In the 2D action game, the content changes periodically, and players need to make different equipment selections, enter game levels, and defeat enemies to achieve level victories. In the game levels, player operations are more frequent, while in the menu equipment selection, player operations are reduced and require players to think, so the overall operation has periodicity.

The time domain diagram of the original game data is as follows Fig. 4.40 and Fig. 4.41:

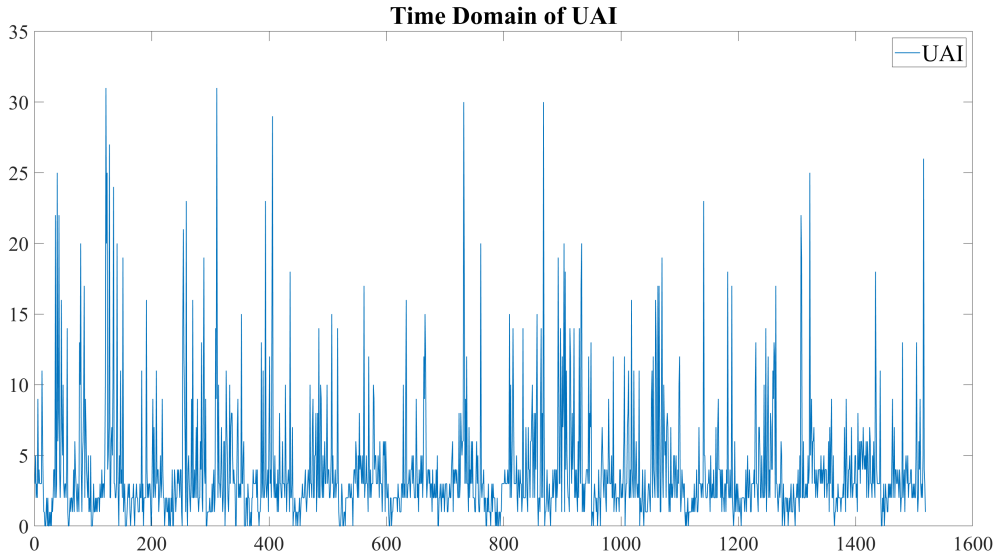


Fig. 4.40 Time Domain of UAI

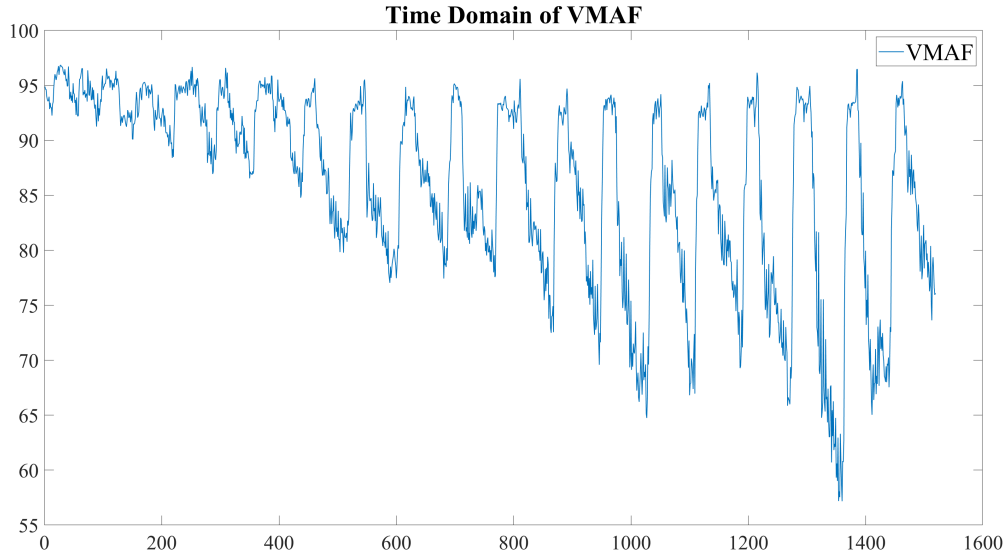


Fig. 4.41 Time Domain of VMAF

From the above time-domain graph, it can be seen that the UAI and VMAF sequence data of Case 2 are more complex than the sequence data of Case 1. First, use the previous process to perform delay analysis on the sequence data and set the basic parameters of the NARX network as shown in the following Table 4.7 and 4.8:

parameter	$1^{th}layer$	$2^{th}layer$	Start delay	TFi
value	10	10	39	tansig/purelin

Table 4.7 Network Parameters for Game 2

parameter	BTF	BLF	PF	othres
value	trainlm	learngdm	mse	searched

Table 4.8 Network Parameters for Game 2 Table Continue

Create a NARX network object, train it, and simulate it, with the results as follows:

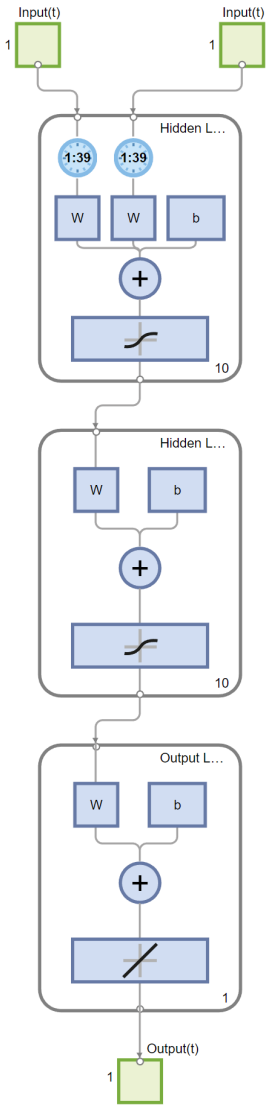


Fig. 4.42 Network for Game 2

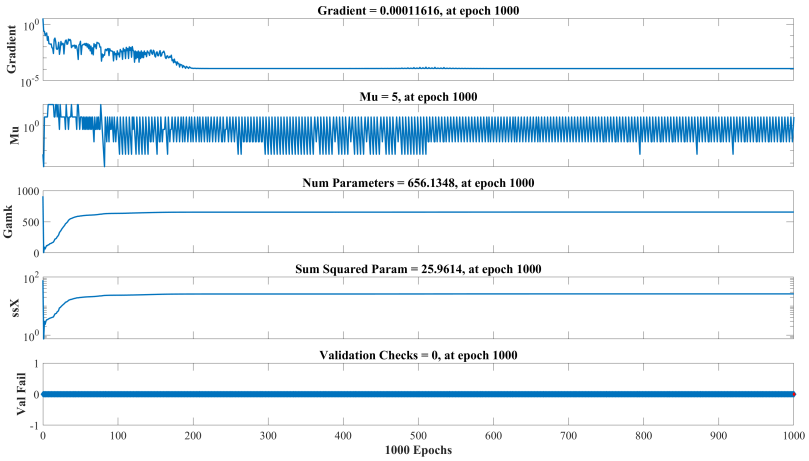


Fig. 4.43 Training States

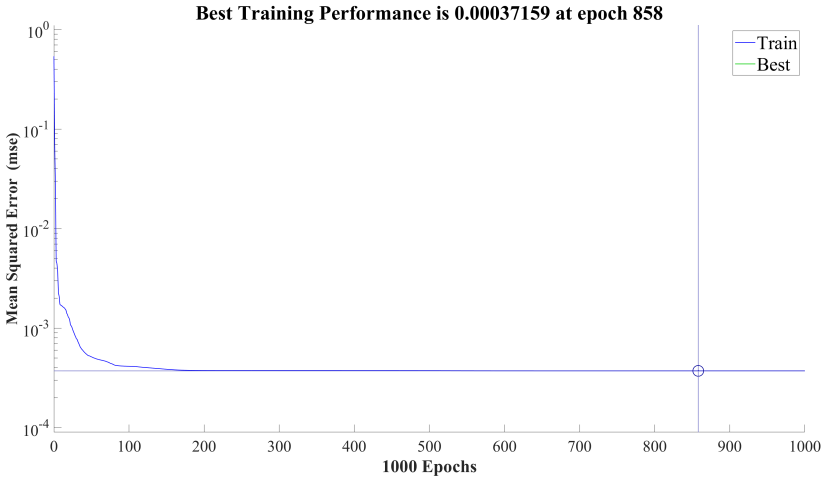


Fig. 4.44 Performance

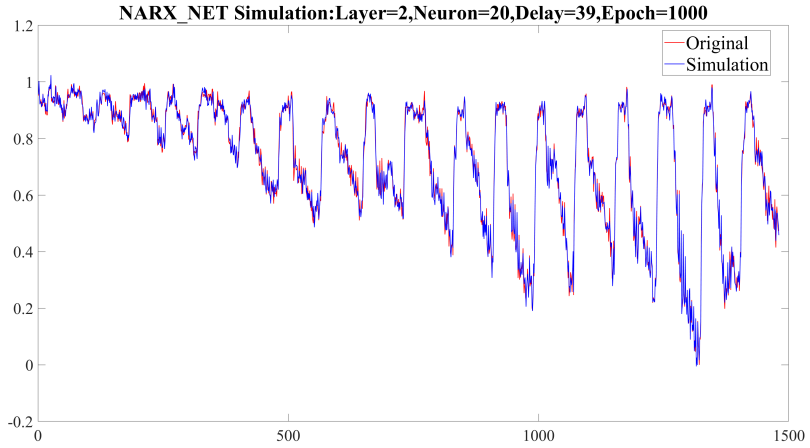


Fig. 4.45 Simulation

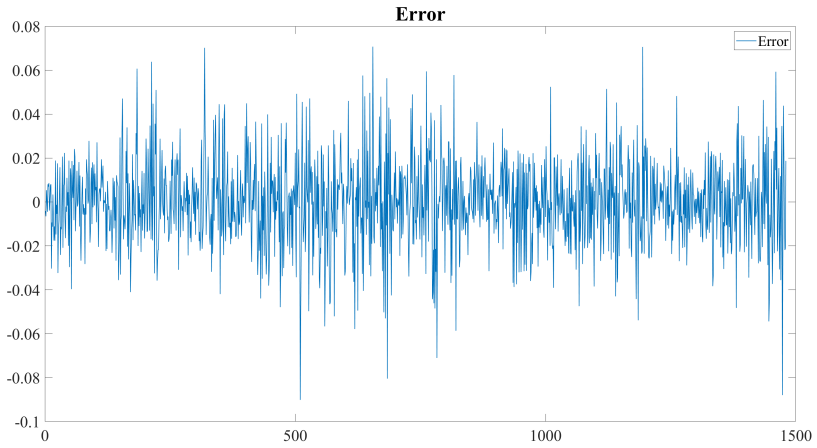


Fig. 4.46 Error

The calculated performance metrics for the NARX model are as follows:
narx_net: layer=[10 10], Delay=39, Epoch=1000, Training time=73.4458s,

$e_{mean} = -4.8241 \times 10^{-9}$, $e_{max} = 0.0901$, $e_{sum} = 0.5500$, $e_{fit} = 91.17\%$

Further analysis of the above simulation results is conducted, and a correlation function analysis graph is plotted:

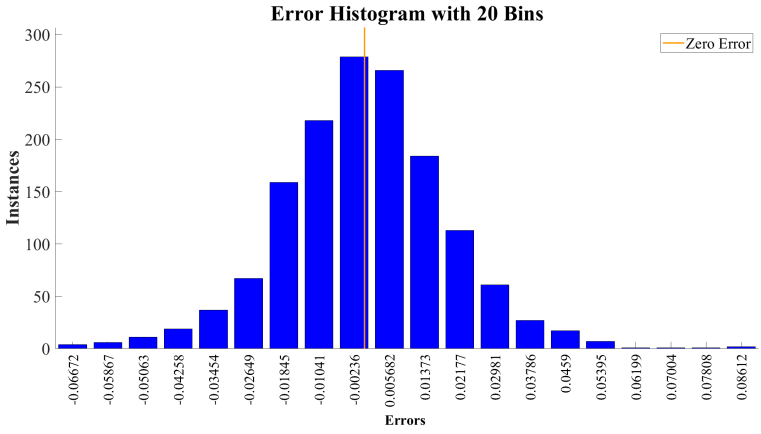


Fig. 4.47 Histogram

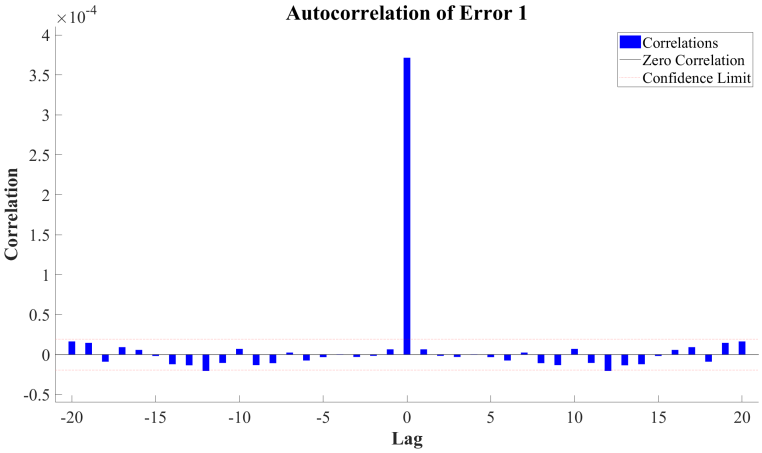


Fig. 4.48 Error Correction

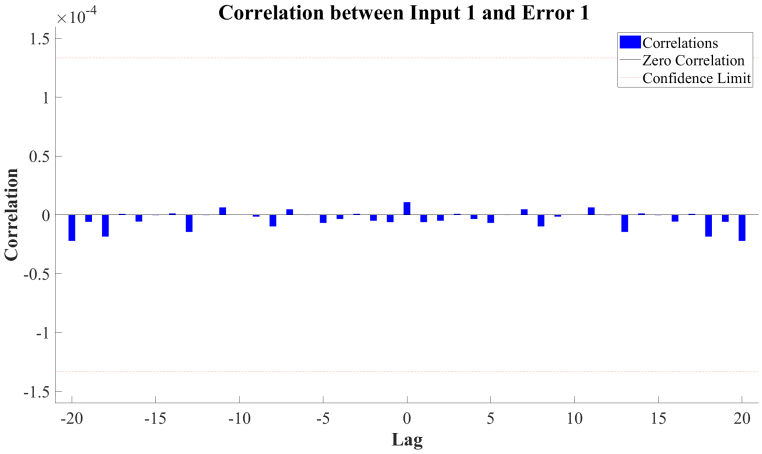


Fig. 4.49 Error Input

The graph indicates that the error distribution, the autocorrelation function of the errors, and the correlation function between input and error all demonstrate good reliability in the simulation output.

The regression and response effect graphs are plotted as follows Fig. 4.50 and Fig. 4.51:

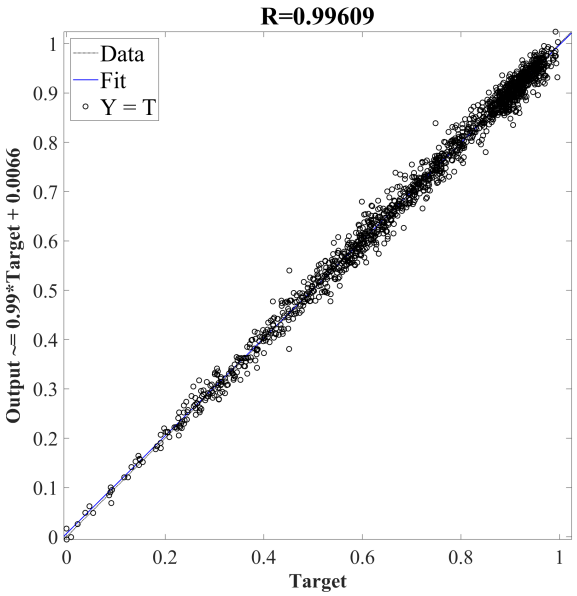


Fig. 4.50 Regression

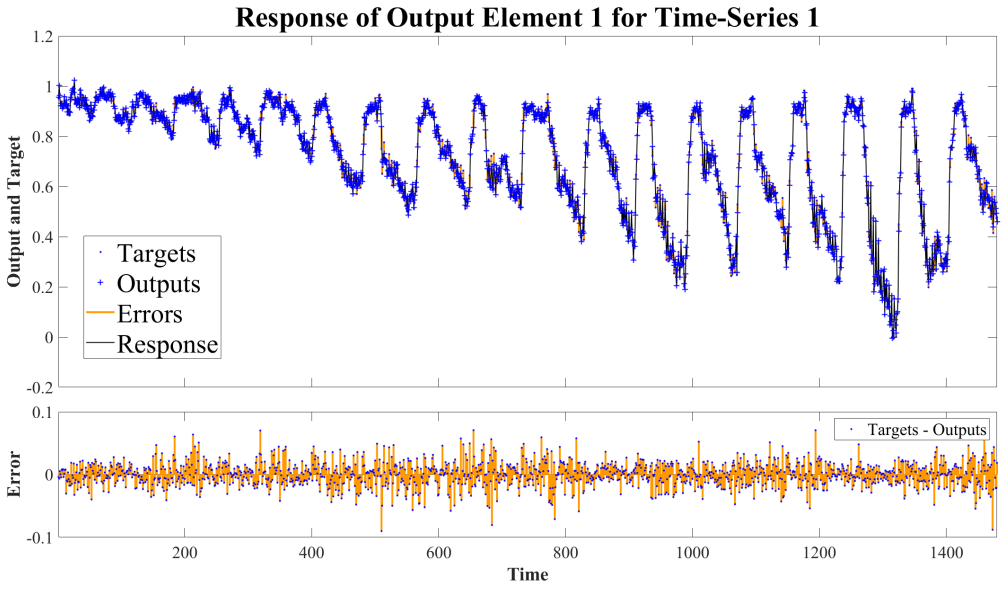


Fig. 4.51 Response

The above figure shows that the regression effect and responsiveness are good.

As a verification, take a part of the game 2 data set, and its time domain graph is as follows Fig. 4.52 and Fig. 4.53:

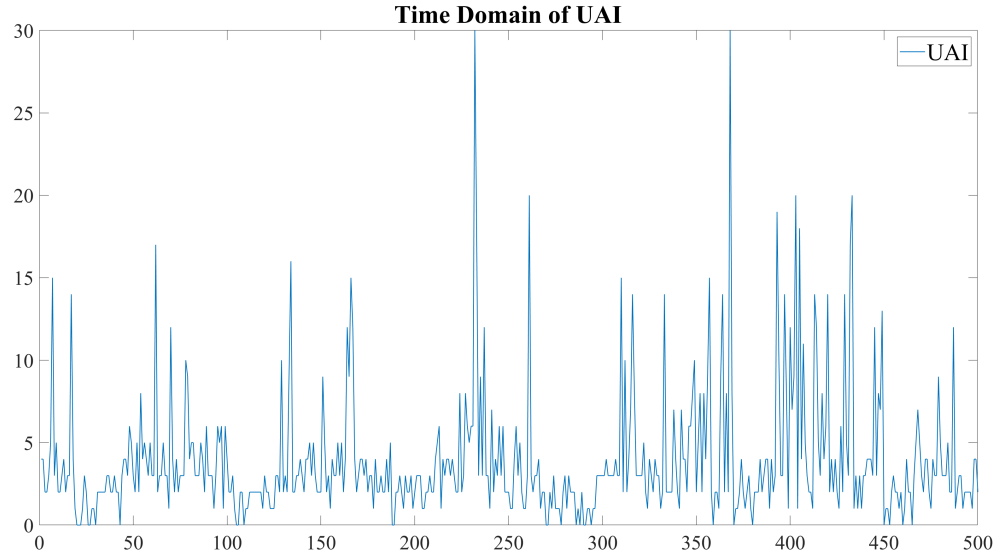


Fig. 4.52 UAI

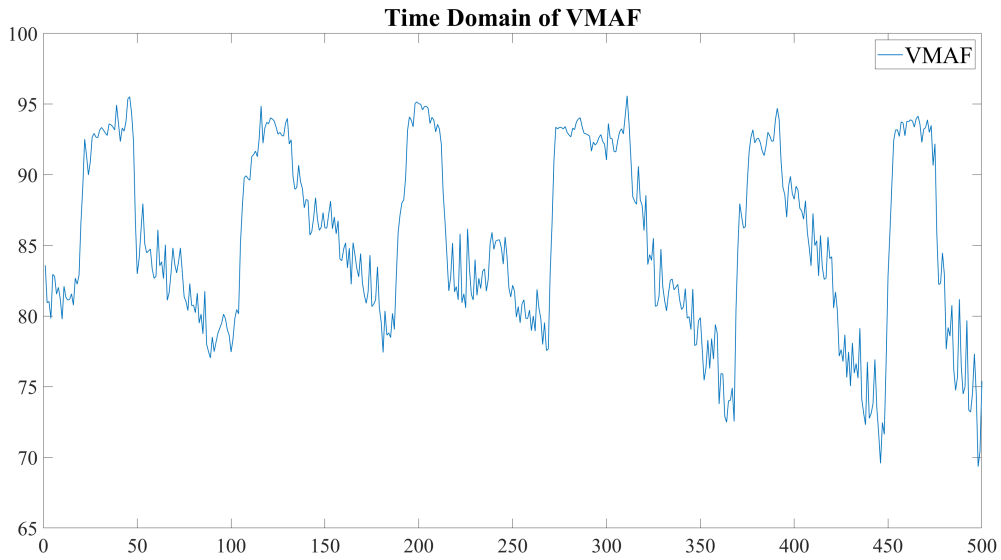


Fig. 4.53 VMAF

The simulation results using the trained NARX network are as follows Fig. 4.54 and Fig. 4.55:

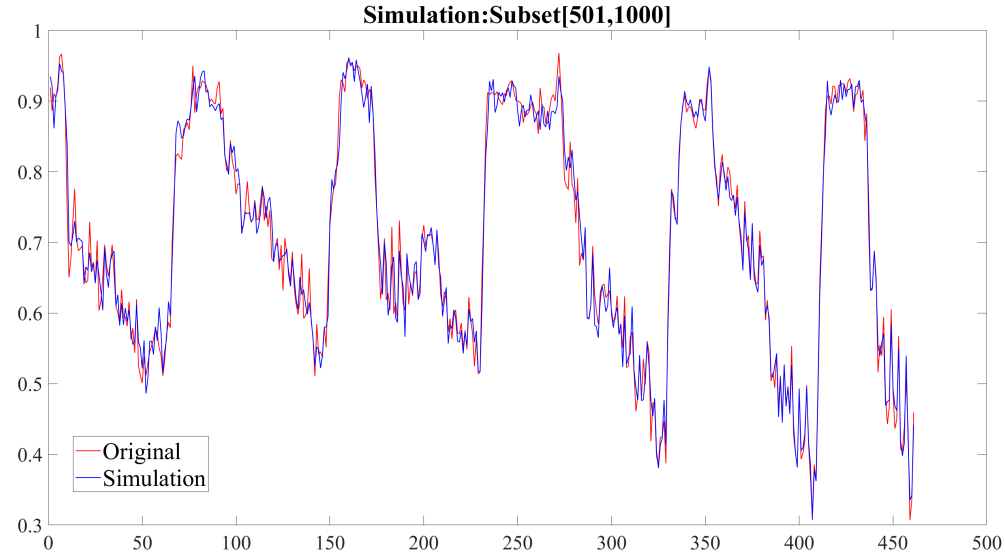


Fig. 4.54 Simulation

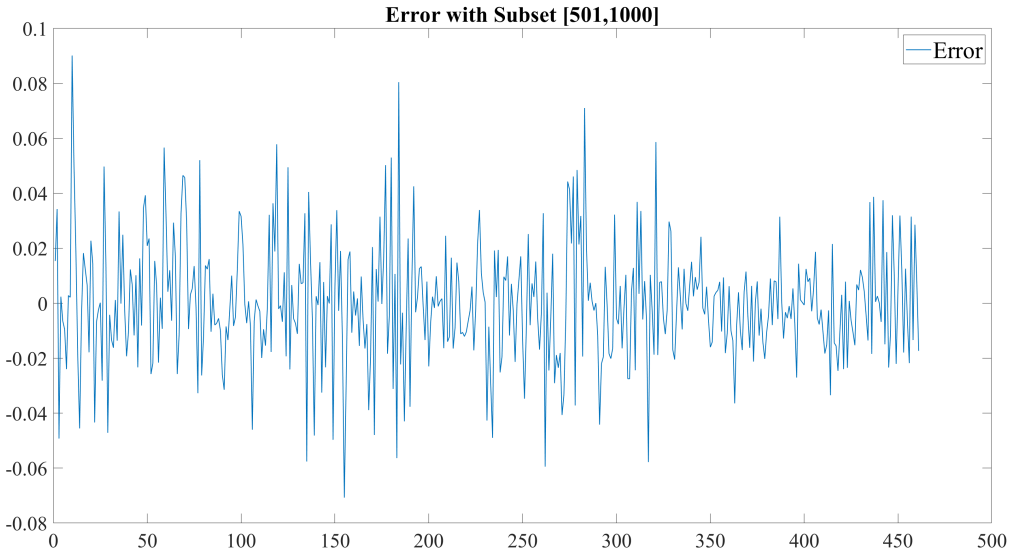


Fig. 4.55 Verify-error

The performance index calculation results of the NARX model are:

narx_net: layer=[10 10], Delay=39, Epoch=1000, Training times=73.4458s,

$e_{mean} = 5.55484 \times 10^{-5}$, $e_{max} = 0.0901$, $e_{sum} = 0.2191$, $e_{fit} = 86.36\%$

Similarly, the error analysis diagram of the simulation results is drawn as follows Fig. 4.56, Fig. 4.57 and Fig. 4.58:

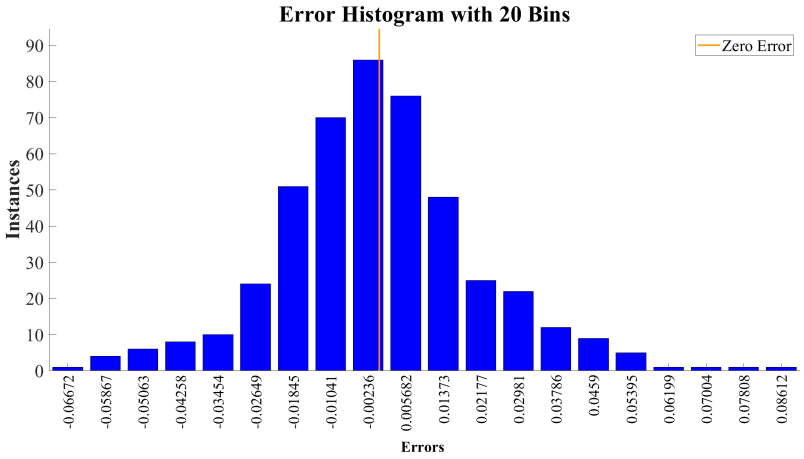


Fig. 4.56 Histogram

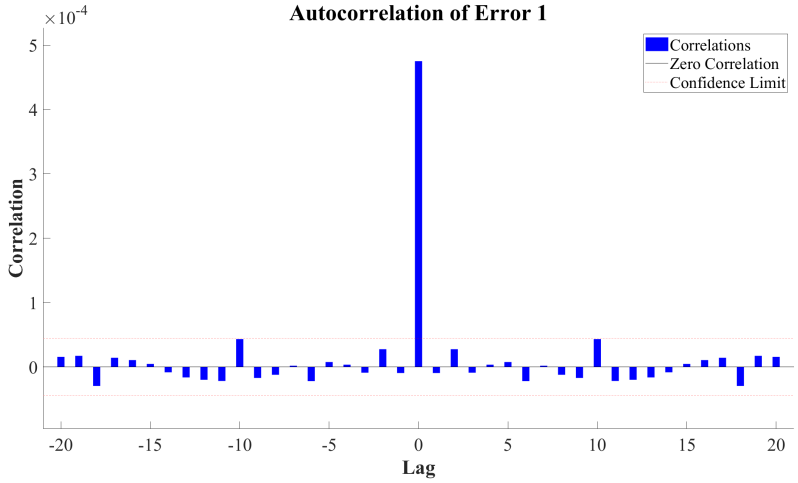


Fig. 4.57 Error Correction

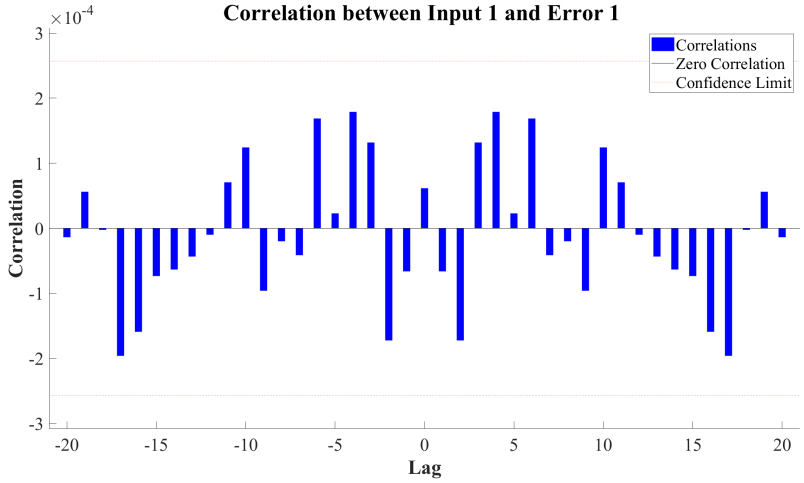


Fig. 4.58 Error Input

The error histogram indicates that the errors for the vast majority of instances fall within the interval [-0.04258, 0.05395]. The autocorrelation function of the errors remains within the

confidence interval, and the correlation between input and error is very small, showing no regular pattern.

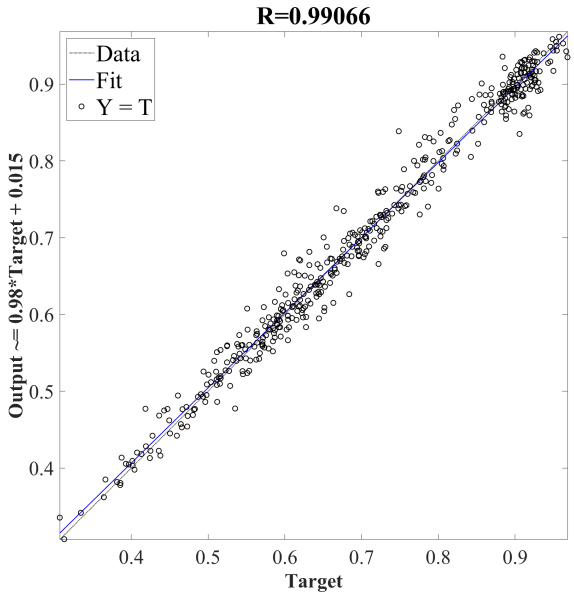


Fig. 4.59 Regression

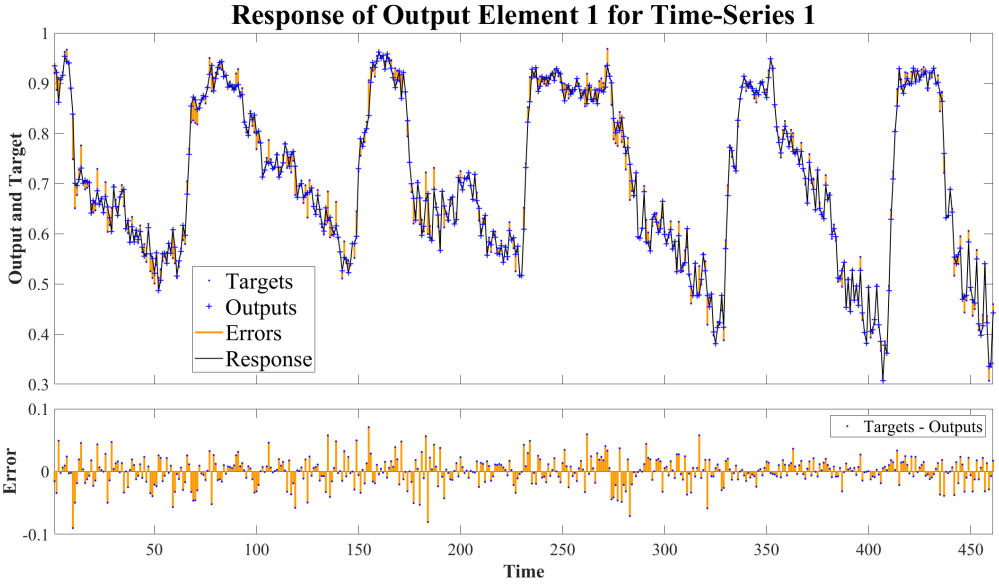


Fig. 4.60 Response

The regression graph shows that Y=T coincides completely with Fit, with an R-value of 0.99066. The response graph indicates that the error in the response of Outputs to Targets lies within the interval [-0.05, 0.05], demonstrating good responsiveness.

4.6.4 NARX Case 3 - Action Game 3D

To further verify the effectiveness of the NARX model, we collected a set of 3D Action game data as a dataset:

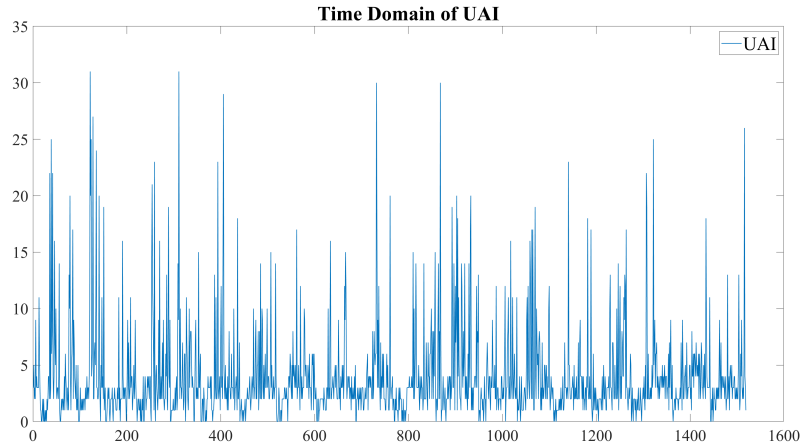


Fig. 4.61 Time Domain of UAI

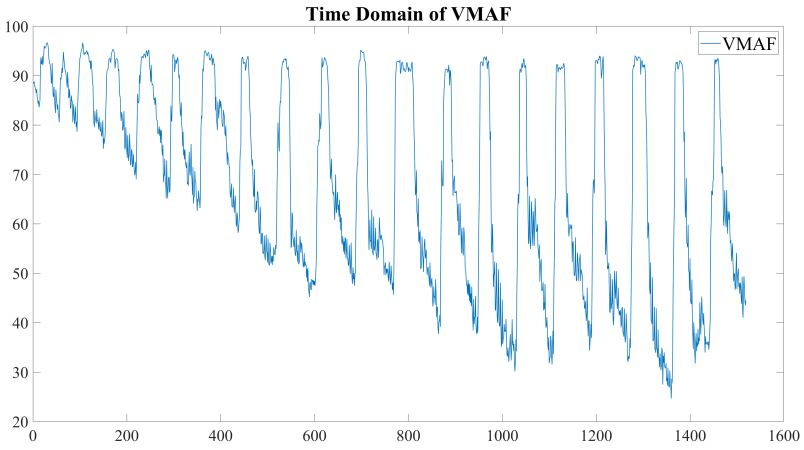


Fig. 4.62 Time Domain of VMAF

The simulation results using the trained NARX network are as follows Fig. 4.63 and Fig. 4.64:

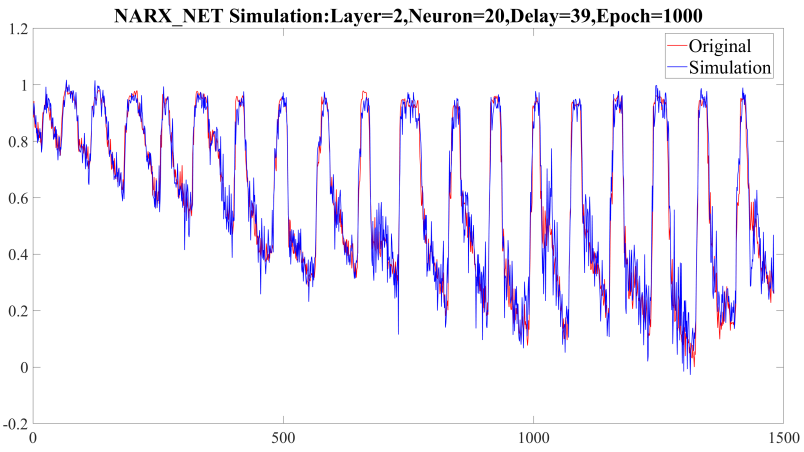


Fig. 4.63 Simulation

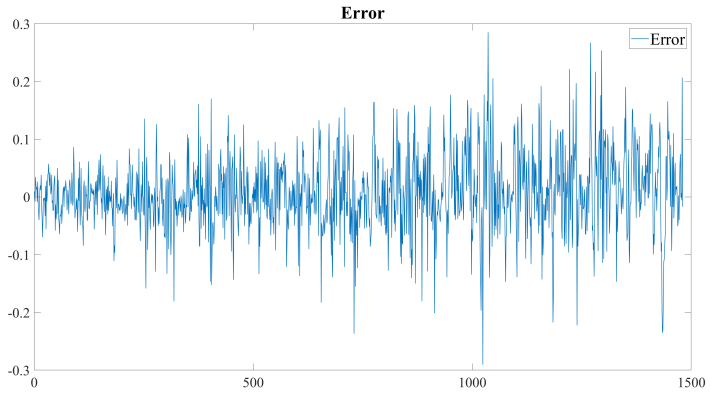


Fig. 4.64 Error

The performance index calculation results of the NARX model are:

narx_net: layer=[10 10], Delay=39, Epoch=1000, $e_{mean} = -0.0075$, $e_{max} = 0.2907$, $e_{sum} = 6.1770$, $e_{fit} = 76.98\%$

The figure above shows that the performance index calculation results of the validation data set are lower than those of the modelling data set, but it still verifies the effectiveness of the model well. The error analysis diagram is further drawn to illustrate this simulation result:

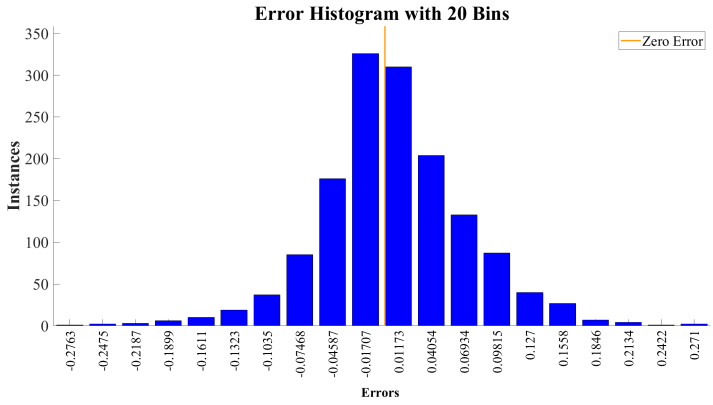


Fig. 4.65 Histogram

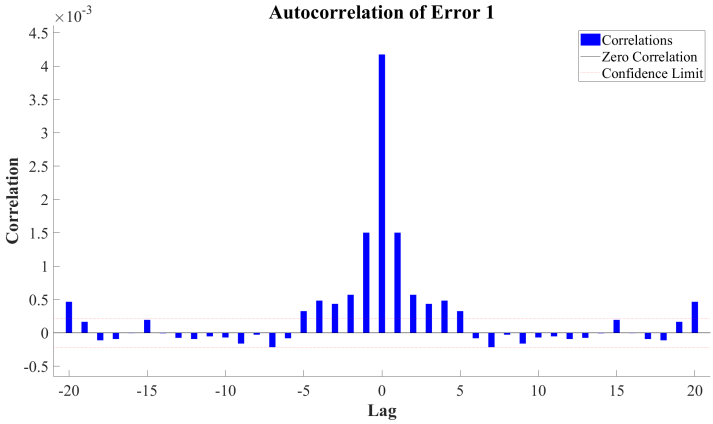


Fig. 4.66 Error Correction

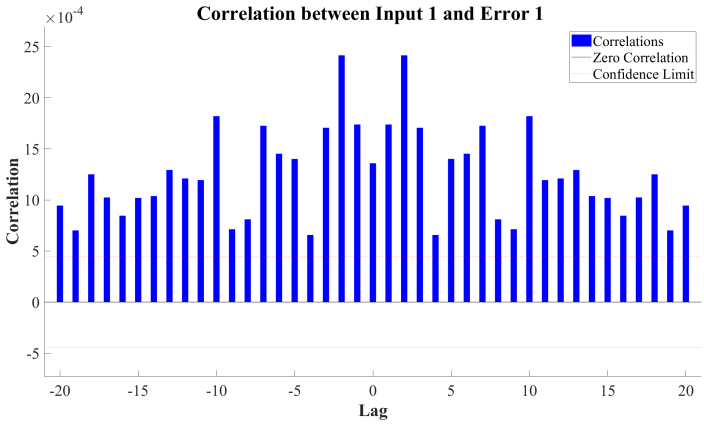


Fig. 4.67 Error Input

The error histogram indicates that the errors for the majority of instances are concentrated within the interval [-0.07468, 0.09815]. The autocorrelation of the errors is very small, showing no trend-like patterns, and there is no correlation between the input and the errors.

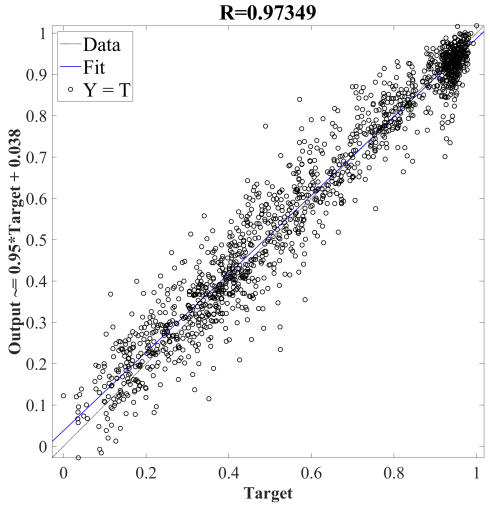


Fig. 4.68 Regression

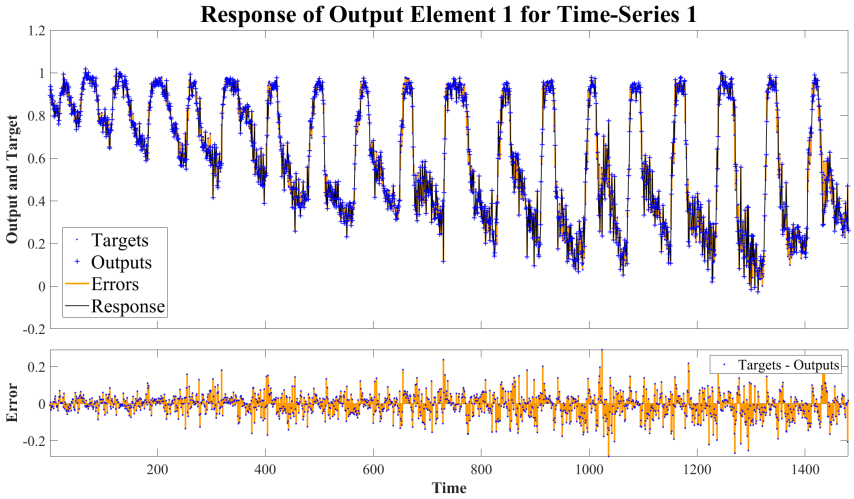


Fig. 4.69 Response

In the regression graph, Y=T and Fit completely overlap, with R=0.97349, but in some regions, the data points are more concentrated, indicating that there are slightly uneven areas of fitting error. The response graph shows that the response error of Outputs to Targets is relatively large, and the responsiveness is slightly poor. This is consistent with the calculation results of the model performance metrics.

4.6.5 NARX Case 4 - RPG Game

Next, we selected the RPG game, in which users need to perform continuous high-frequency operations to complete each level.

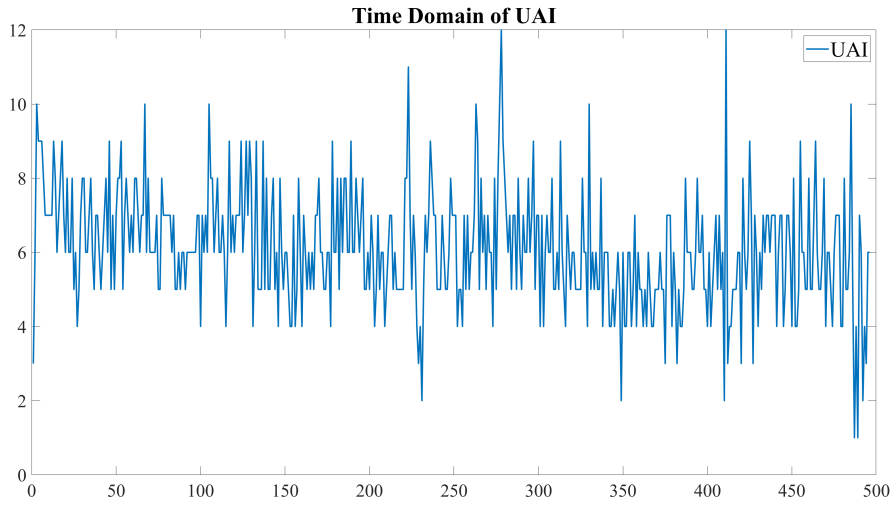


Fig. 4.70 Time Domain of UAI

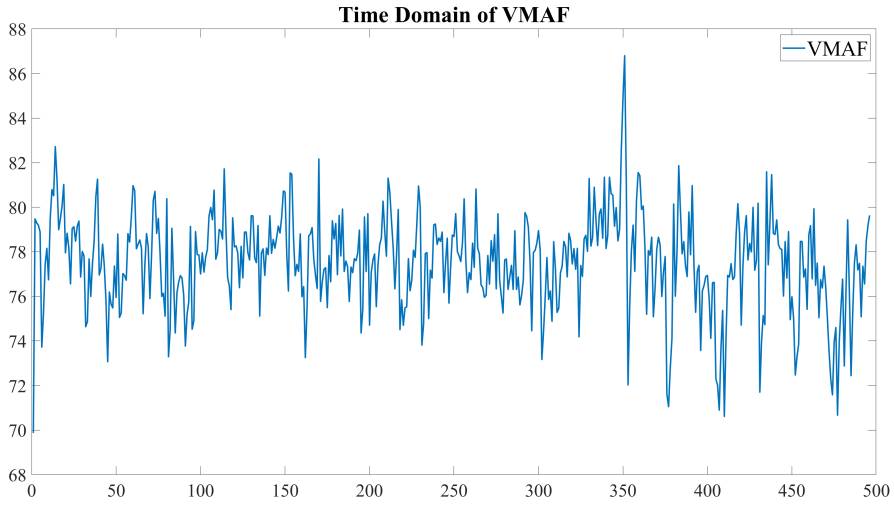


Fig. 4.71 Time Domain of VMAF

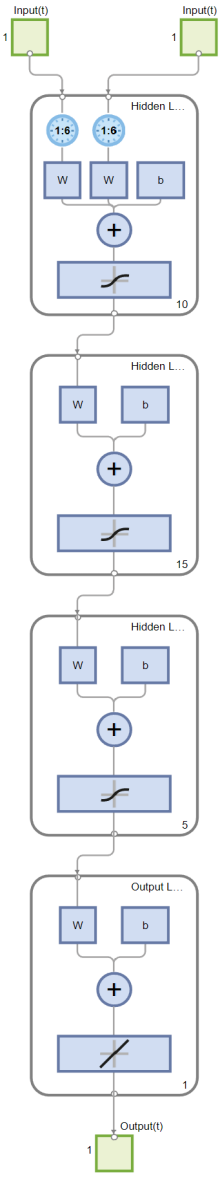


Fig. 4.72 Network for Game 4

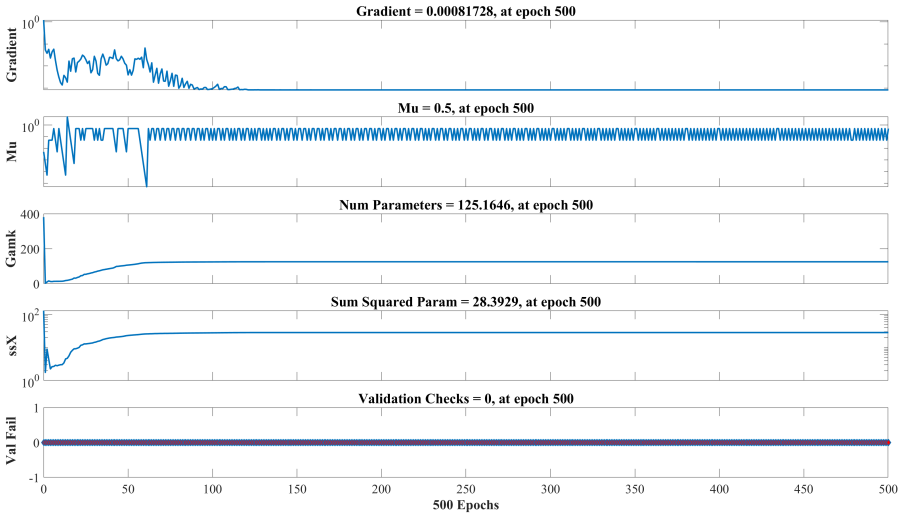


Fig. 4.73 Training States

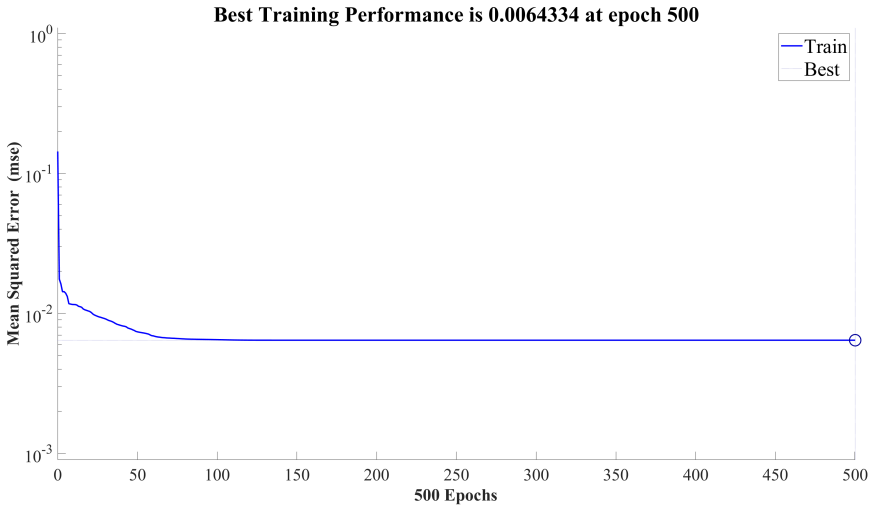


Fig. 4.74 Performance

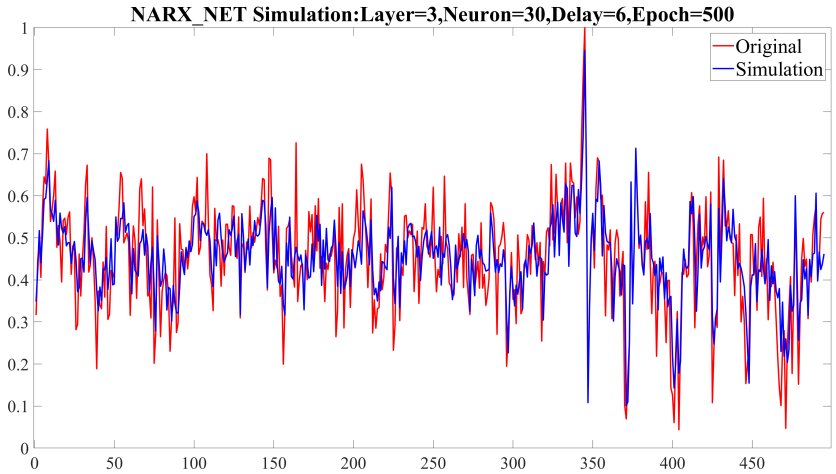


Fig. 4.75 Simulation

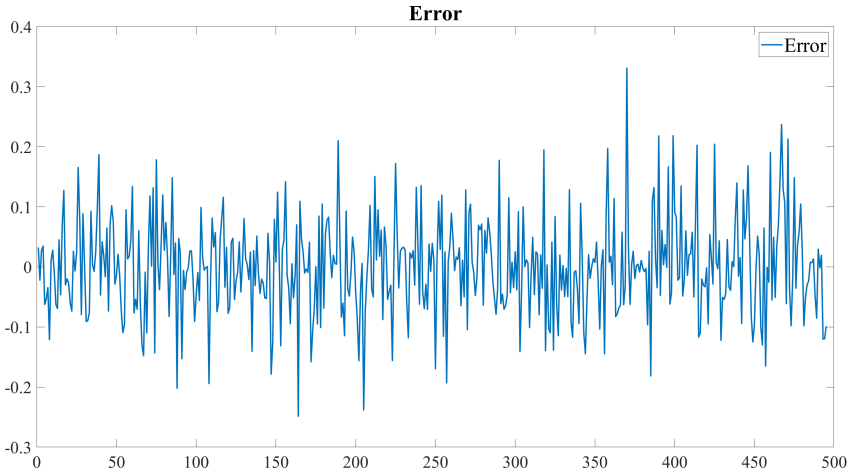


Fig. 4.76 Error

The performance index calculation results of the NARX model of Game 4 are as follows:

narx_net: layer=[10 15 5], Delay=6, Epoch=500, $e_{mean} = -8.8855 \times 10^{-6}$, $e_{max} = 0.3310$, $e_{sum} = 3.1845$, $e_{fit} = 36.33\%$

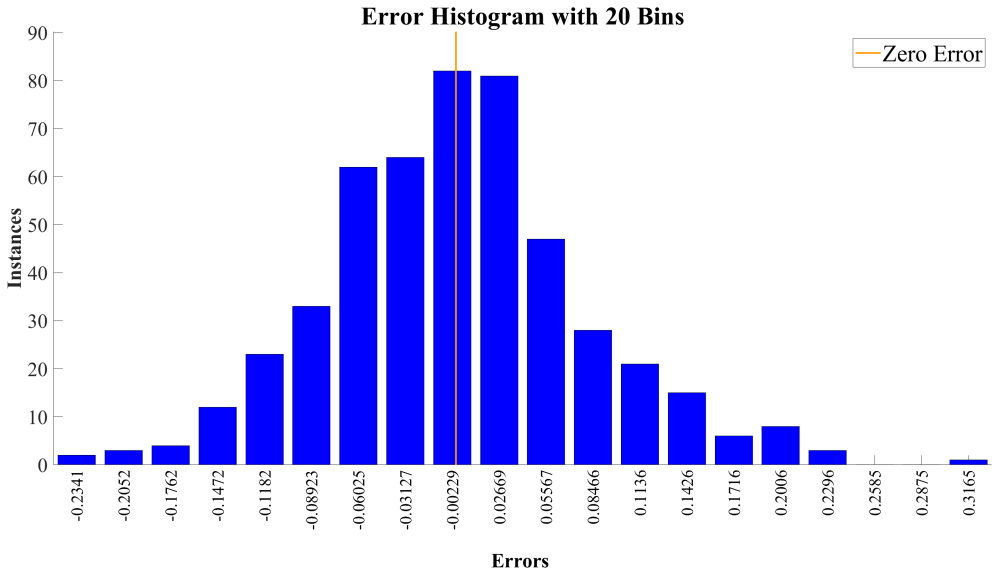


Fig. 4.77 Histogram

The error histogram essentially exhibits an unimodal symmetric distribution, but the peak range is relatively large, appearing across two bins. The absolute values of the errors are also slightly higher, reflecting a significant discrepancy between the model outputs and the actual values. This result is consistent with the calculated performance metrics.

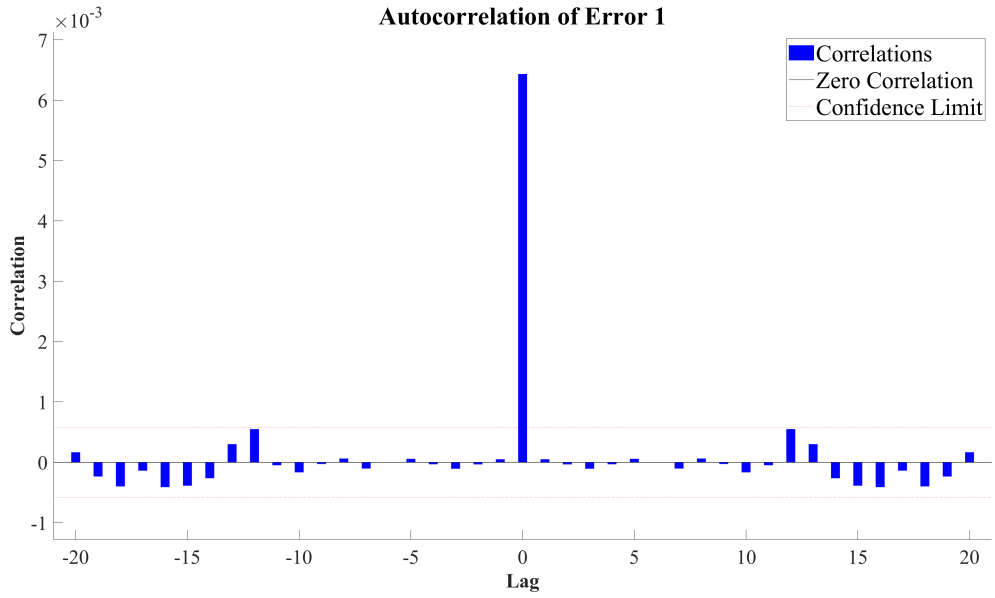


Fig. 4.78 Error Correction

The error autocorrelation exhibits a certain periodicity, showing the same trends within the intervals of delays [-20, -10] and [10, 20]. This indicates that the training sample data contains

certain periodic patterns, which have prevented the training results from achieving the expected outcomes.

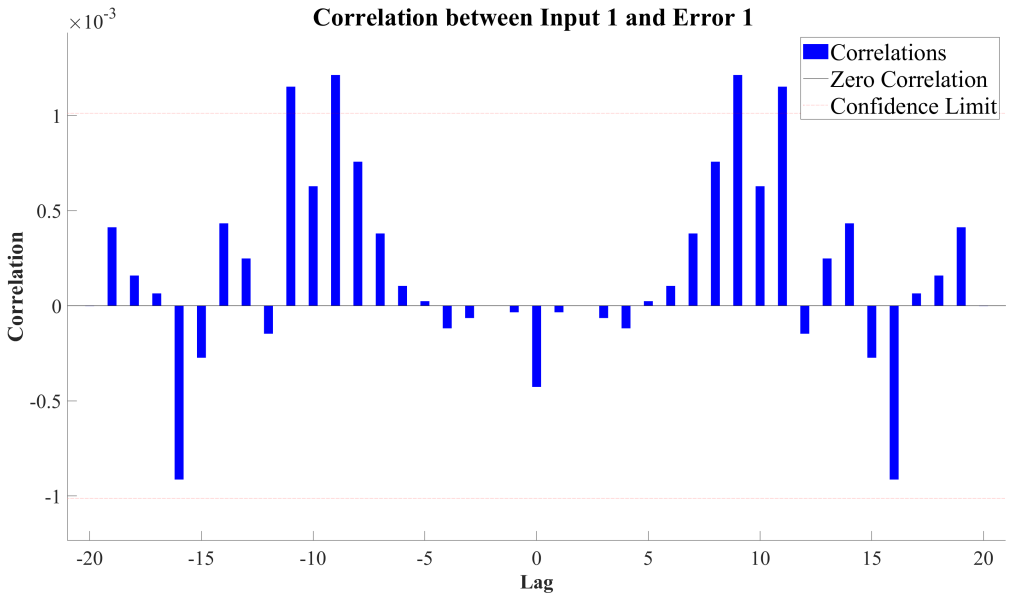


Fig. 4.79 Error Input

The correlation function graph between input and error exhibits more pronounced periodicity, displaying a bimodal shape with a certain level of symmetry. This indicates a high correlation between errors and inputs, suggesting that the periodic variation in errors may be caused by trend-like patterns in the input sample data or by incomplete sample data that fails to fully reflect the characteristics and patterns of UAI and VMAF in a complete game session. Further, this also indicates that the sample dataset is not sufficiently comprehensive and fails to adequately support the training of the NARX model.

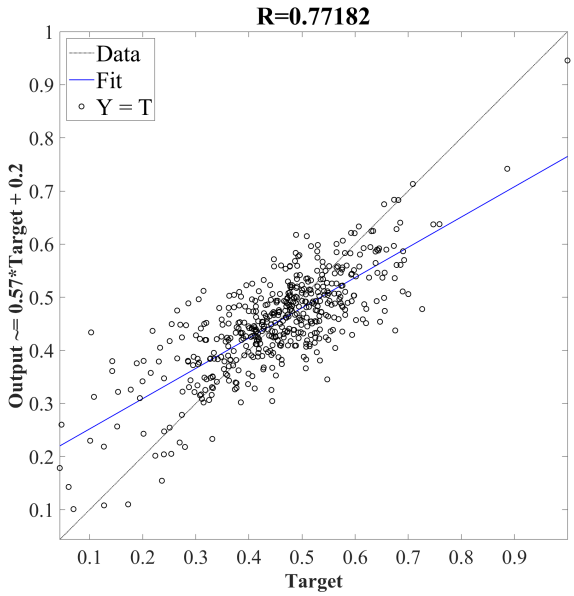


Fig. 4.80 Regression

The regression graph reveals that the fit line Y=T does not coincide with Fit, indicating significant discrepancies. The data is more concentrated within the [0.4, 0.6] interval, and the correlation coefficient, R=0.7718, is also relatively poor.

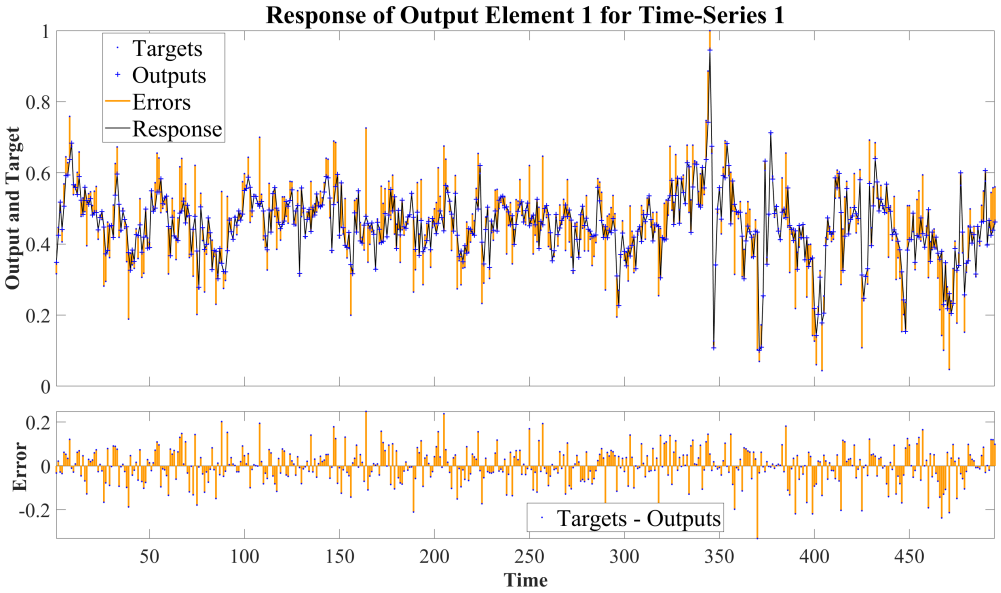


Fig. 4.81 Response

The response graph shows that there is a significant error in the model's output in response to the Targets, especially within the marked areas, where most of the outputs have not adequately responded to the Targets.

From the above content, we found that when the content of a level of this RPG game does not change periodically, our prediction effect on this RPG game is not good. Therefore, we played two games of the same level, and the results are as follows Fig. 4.82, Fig. 4.83, Fig. 4.84, Fig. 4.85, Fig. 4.86, Fig. 4.87 and Fig. 4.88:

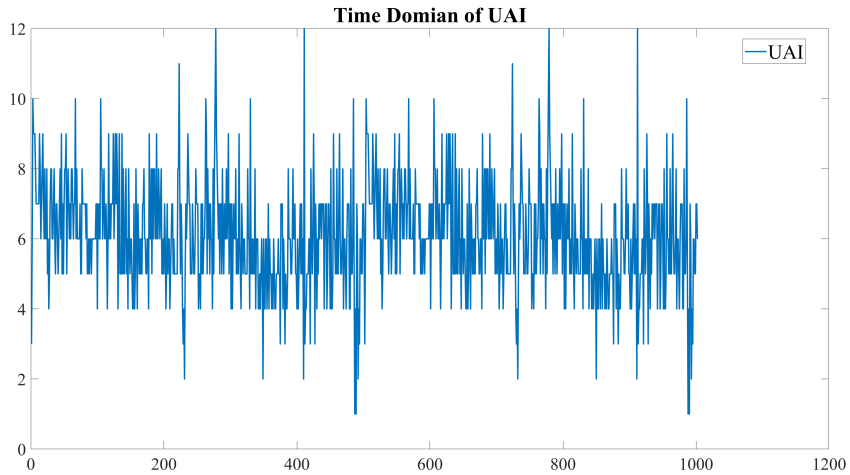


Fig. 4.82 Time Domain of UAI

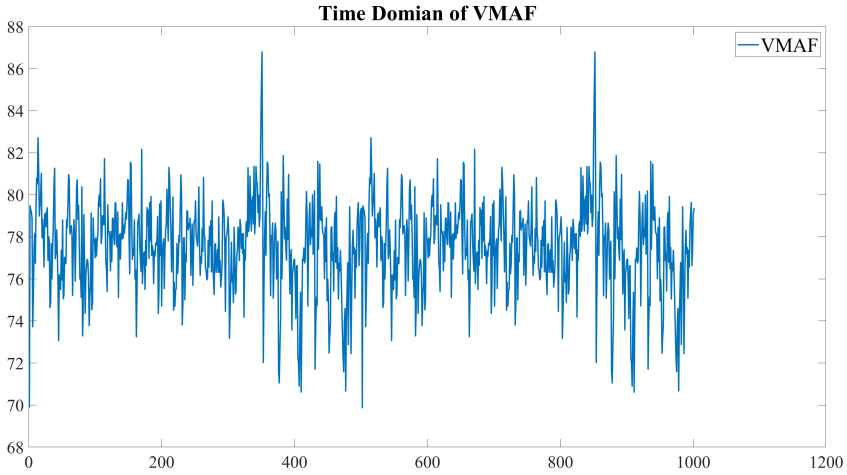


Fig. 4.83 Time Domain of VMAF

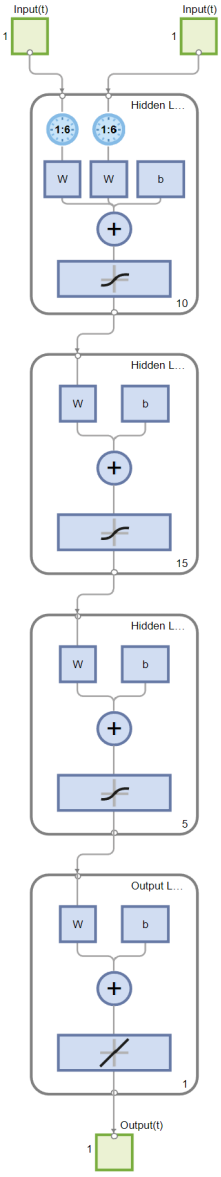


Fig. 4.84 Network for Game 4 Repeat Level

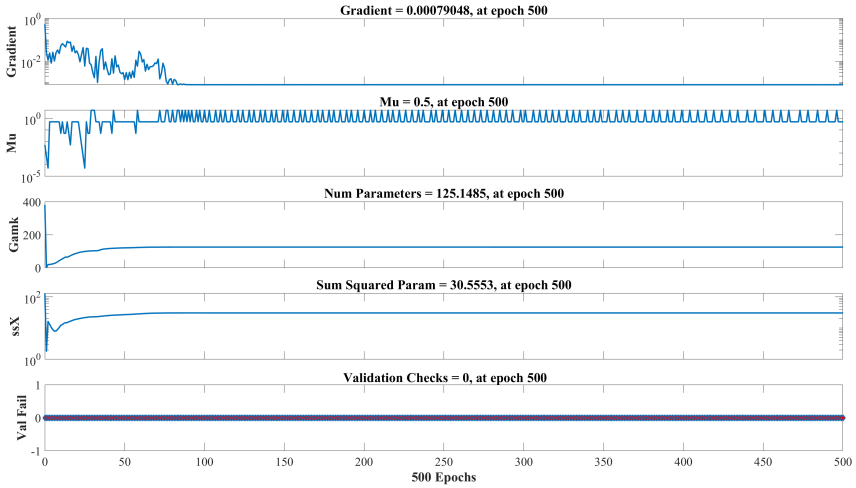


Fig. 4.85 Training States

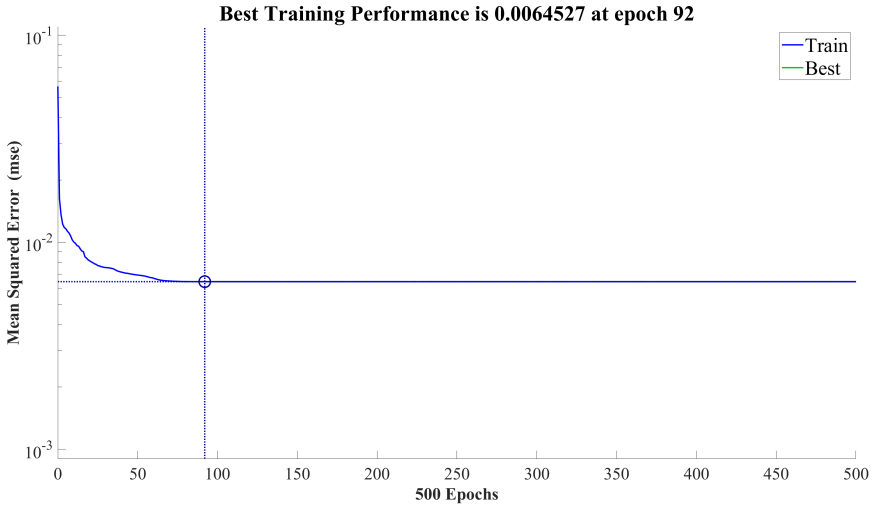


Fig. 4.86 Performance

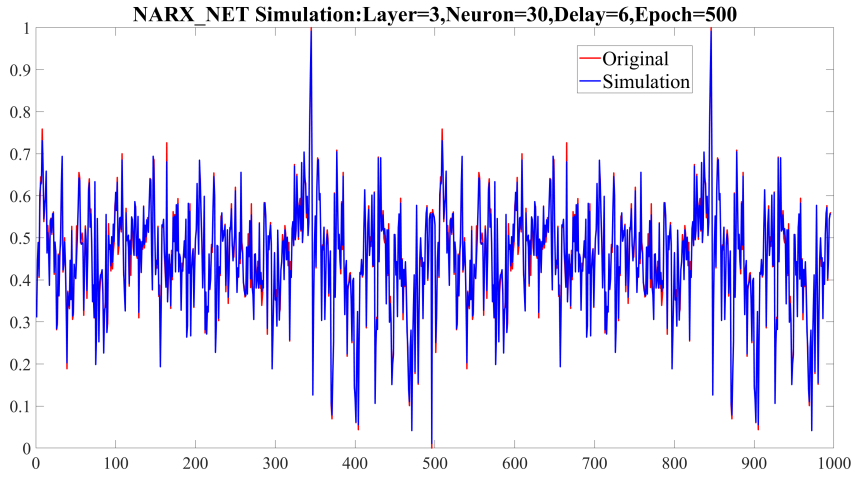


Fig. 4.87 Simulation

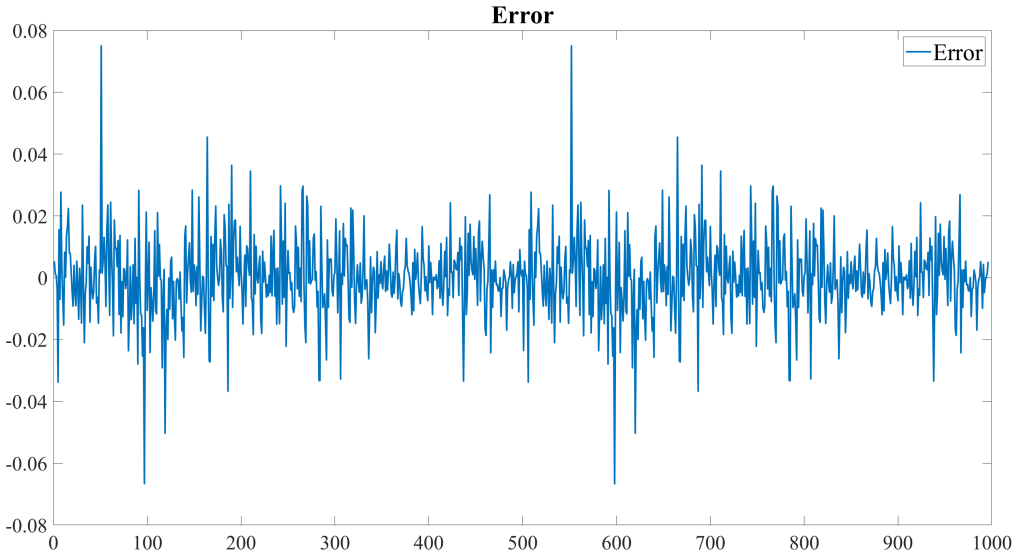


Fig. 4.88 Error

The performance index calculation results of the NARX model for this result are:

narx_net: layer=[10 15 5], Delay=6, Epoch=500, $e_{mean} = 5.6715 \times 10^{-8}$, $e_{max} = 0.0750$, $e_{sum} = 0.1624$, $e_{fit} = 89.93\%$

Further analysis of the simulation results error are as follows Fig. 4.89, Fig. 4.90, Fig. 4.91, Fig. 4.92 and Fig. 4.93:

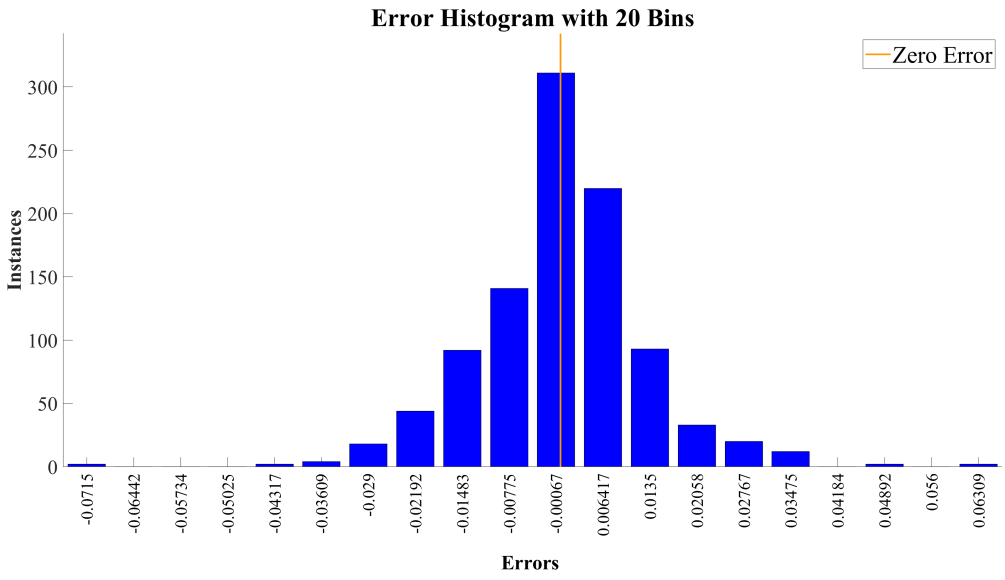


Fig. 4.89 Histogram

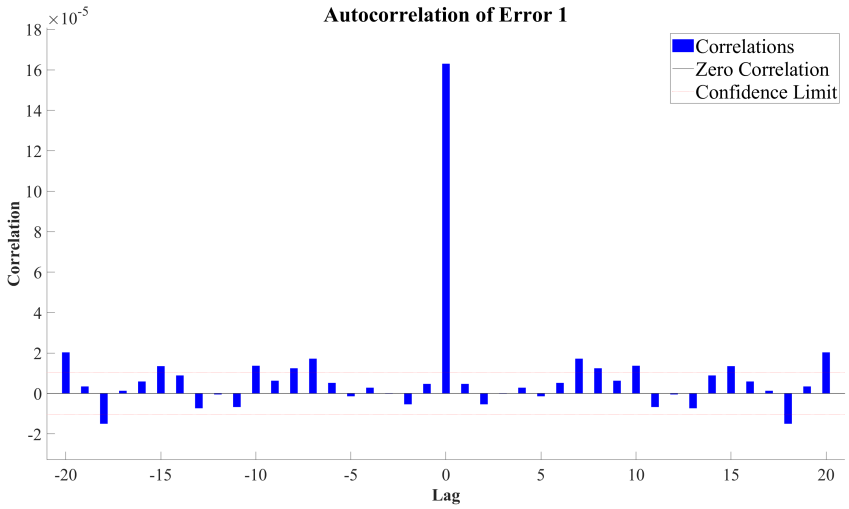


Fig. 4.90 Error Correction

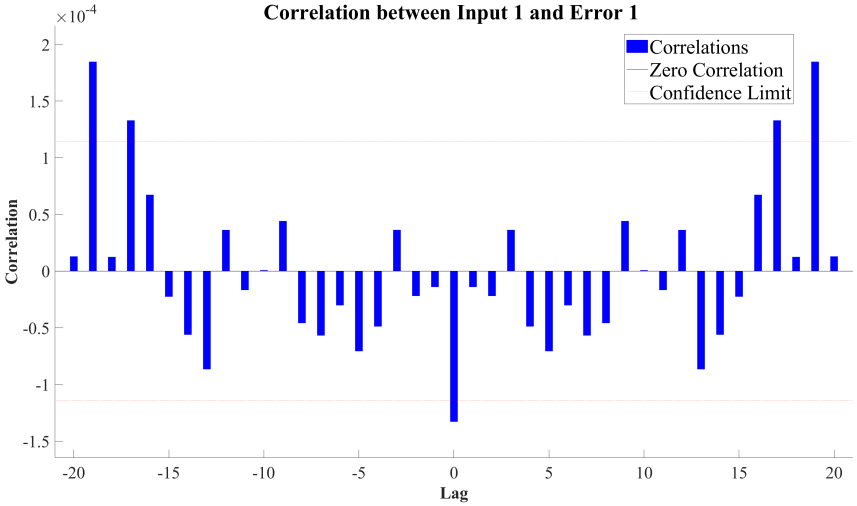


Fig. 4.91 Error Input

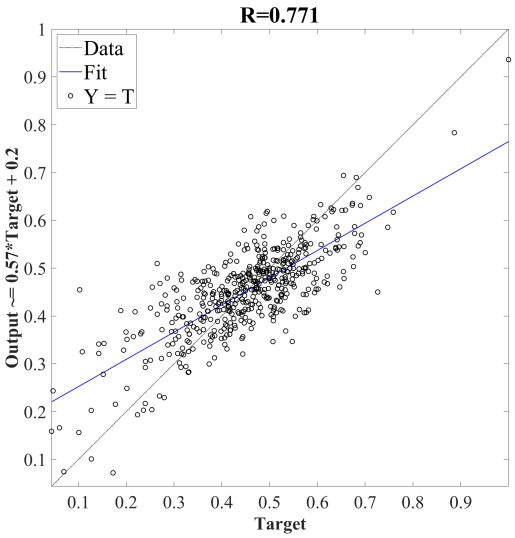


Fig. 4.92 Regression

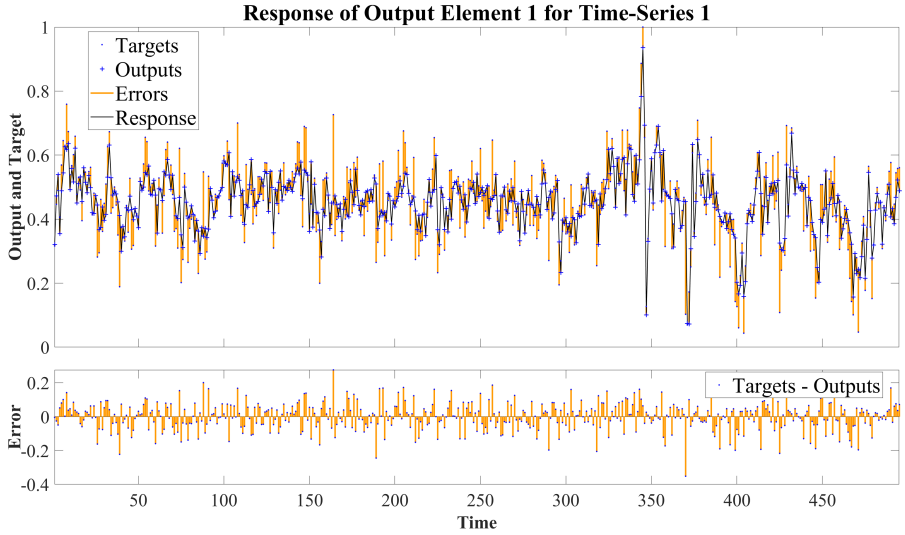


Fig. 4.93 Response

The error histogram shows that the vast majority of instances are concentrated within the error interval [-0.0290, 0.03475], indicating that the model training accuracy is very good. The error autocorrelation coefficients are all within 1.8×10^{-4}, indicating no correlation. The correlation coefficients between the input and error exhibit a certain periodicity, indicating that the augmented dataset has increased the periodicity of UAI-VMAF sequence changes, which is consistent with the experimental design. The regression graph shows a certain amount of regression error, with the output points being relatively concentrated and R=0.7710. The response graph also reflects the poor regression error and responsiveness. However, compared to the previous single-level game results, the simulation effect has greatly improved, indicating that the NARX model has a good predictive effect on UAI-VMAF data with obvious periodicity.

4.6.6 NARX Case 5 - Music Game

Finally, we re-selected a music game and simulated it. Similarly, we first conducted a single-level experiment and then a double-level experiment for comparative analysis.

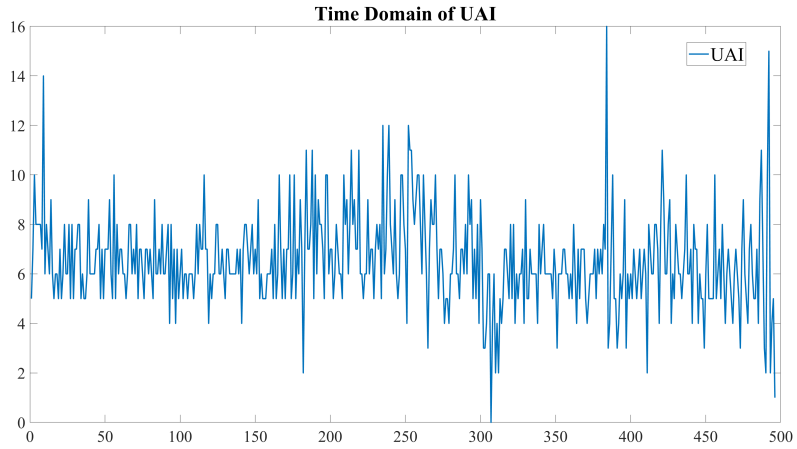


Fig. 4.94 Time Domain of UAI

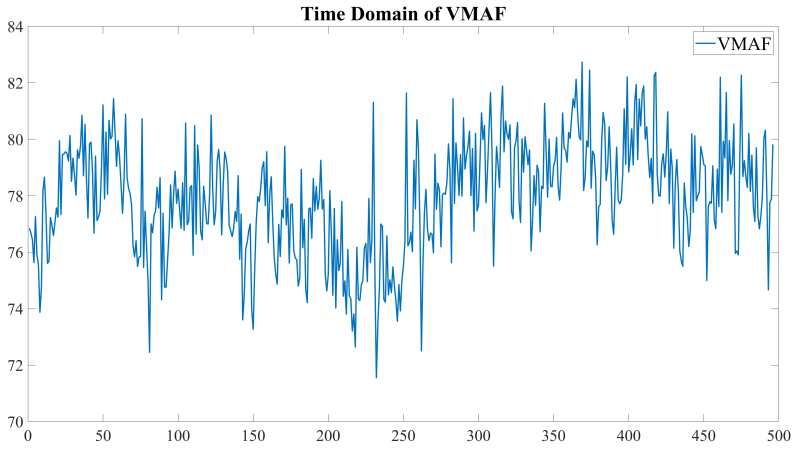


Fig. 4.95 Time Domain of VMAF

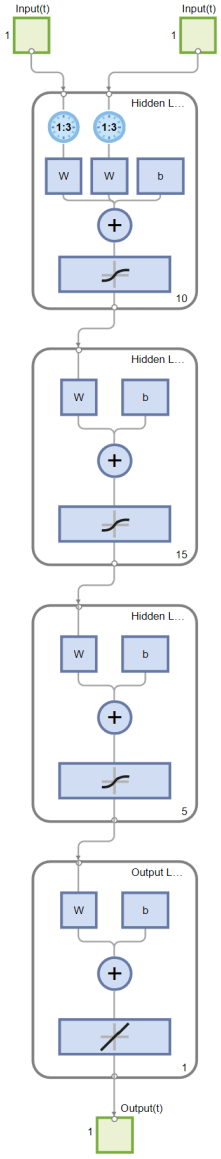


Fig. 4.96 Network for Game 5

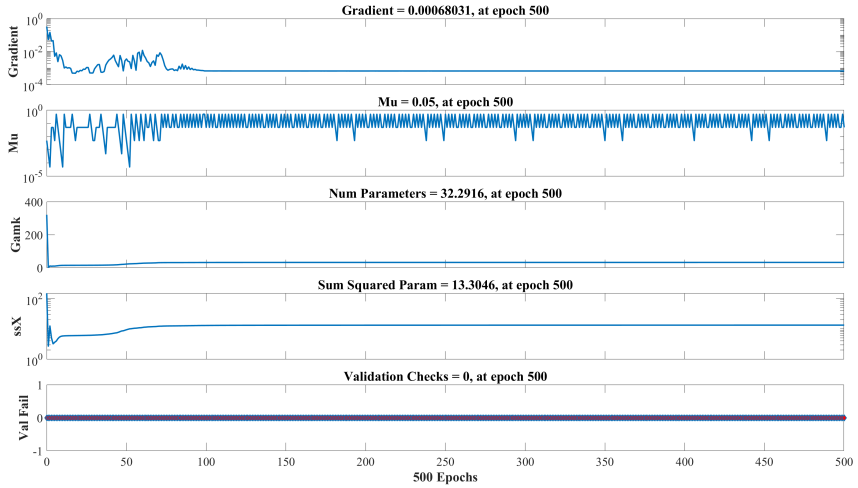


Fig. 4.97 Training States

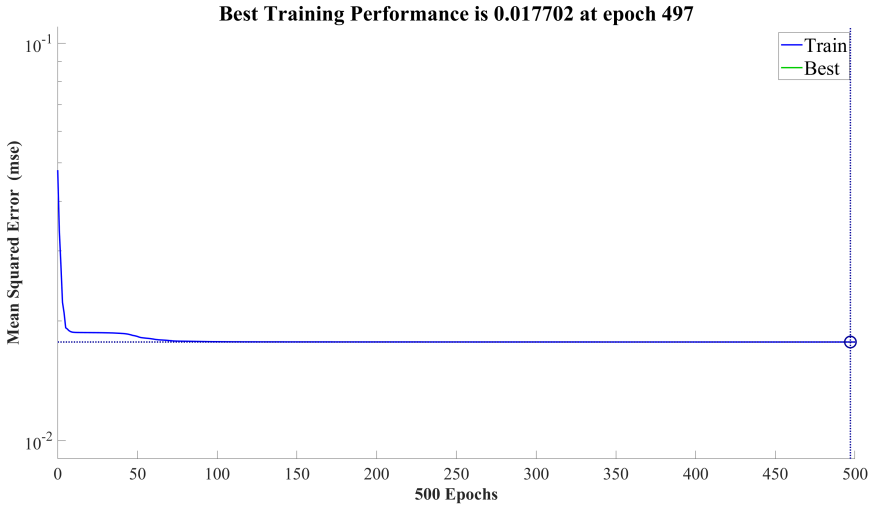


Fig. 4.98 Performance

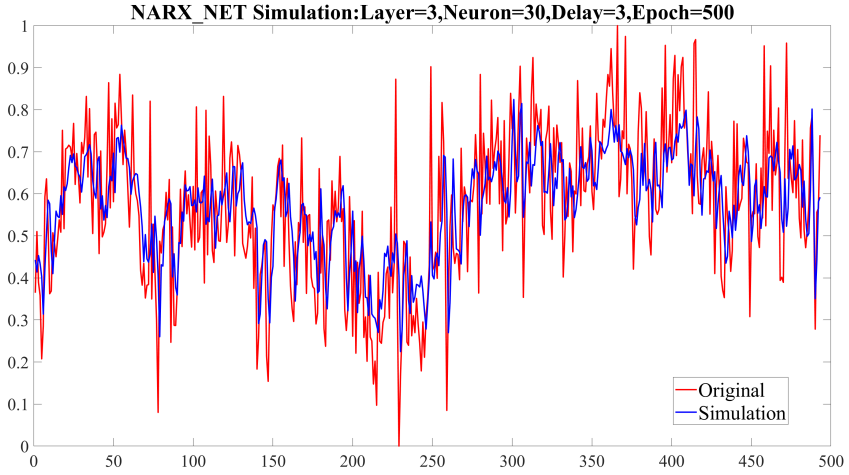


Fig. 4.99 Simulation

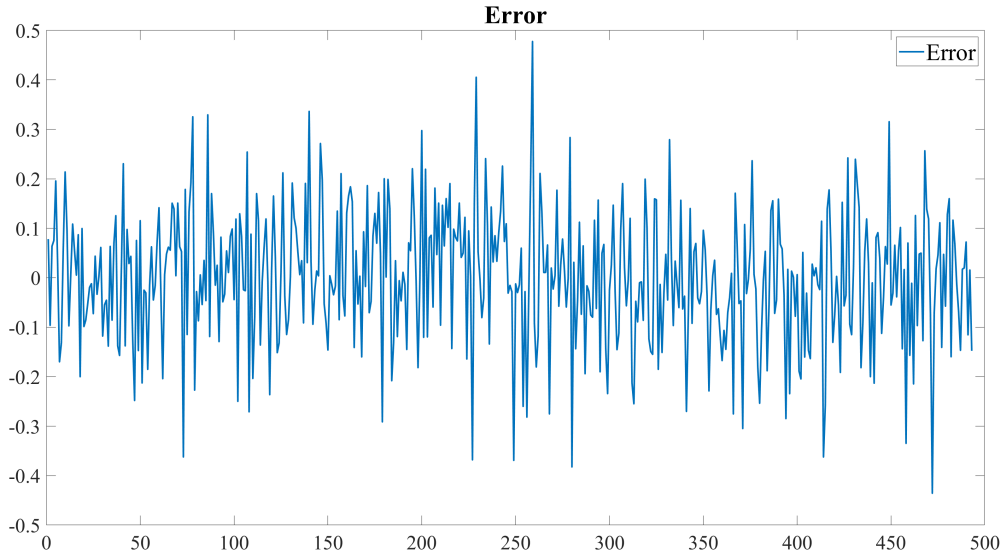


Fig. 4.100 Error

The performance index calculation results of the NARX model are:

narx_net: layer=[10 15 5], Delay=6, Epoch=500, $e_{mean} = -3.7960 \times 10^{-6}$, $e_{max} = 0.4778$, $e_{sum} = 8.7270$, $e_{fit} = 25.78\%$

Further analysis of the simulation results error are as follows Fig. 4.101, Fig. 4.102, Fig. 4.103, Fig. 4.104 and Fig. 4.105:

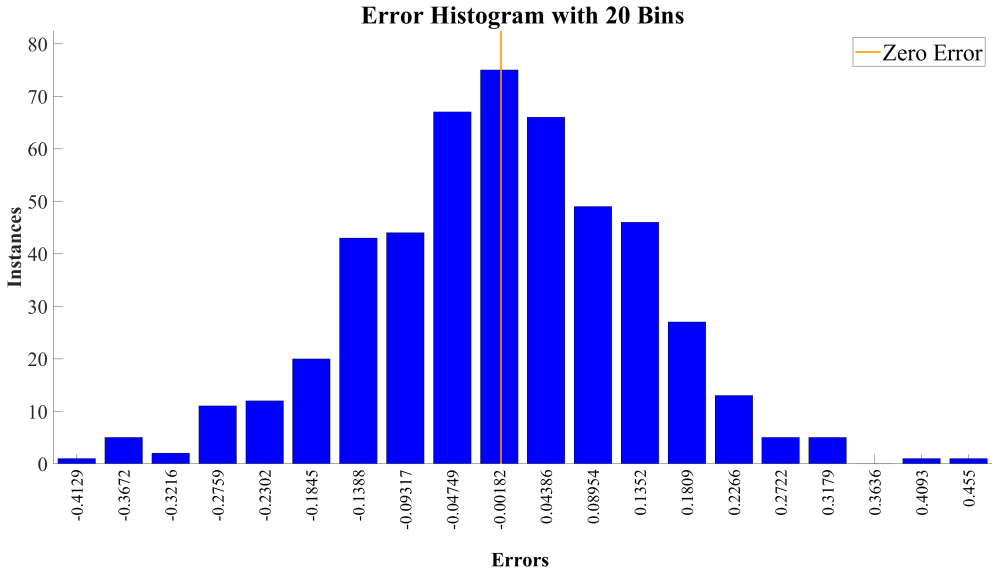


Fig. 4.101 Histogram

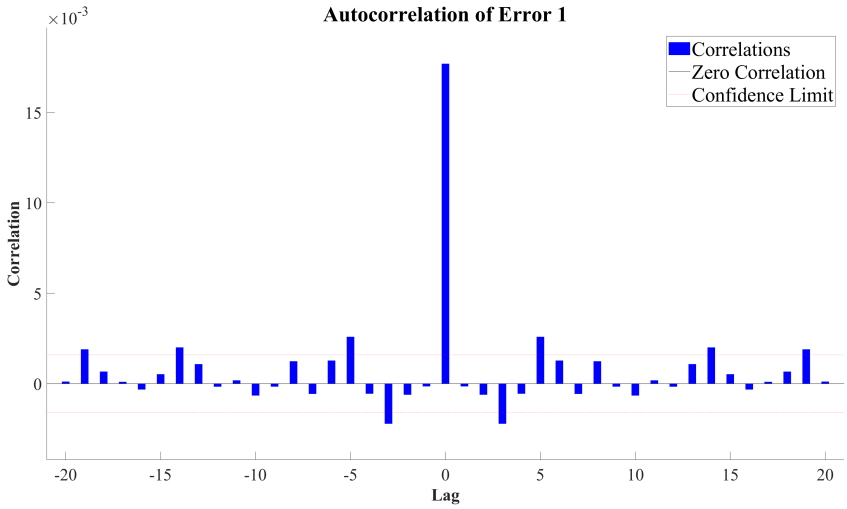


Fig. 4.102 Error Correction

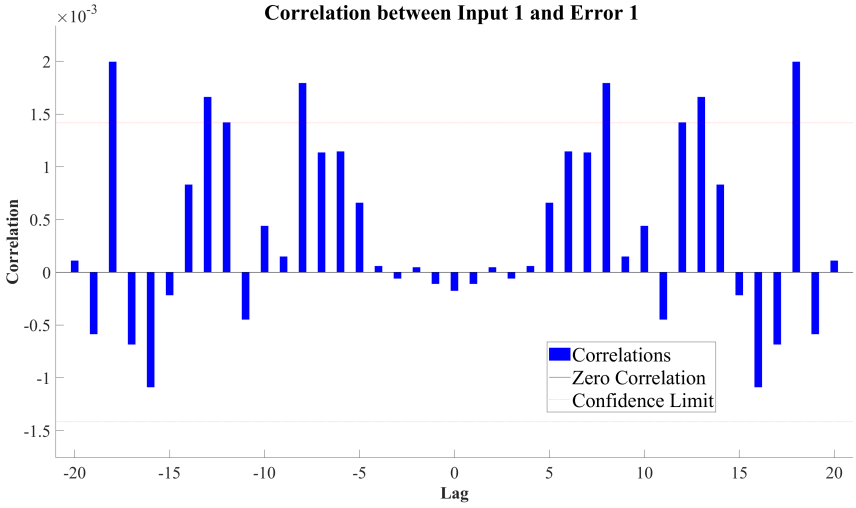


Fig. 4.103 Error Input

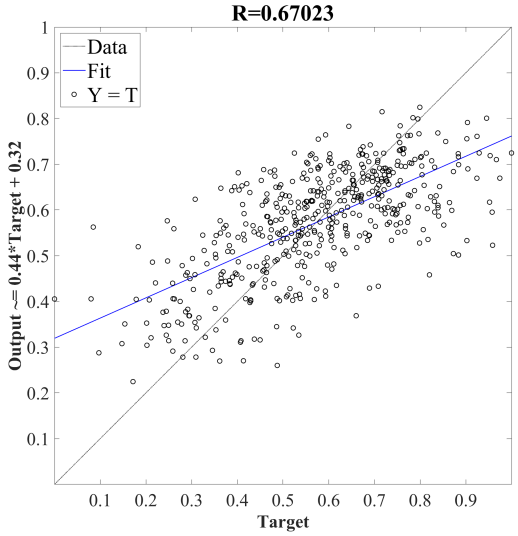


Fig. 4.104 Regression

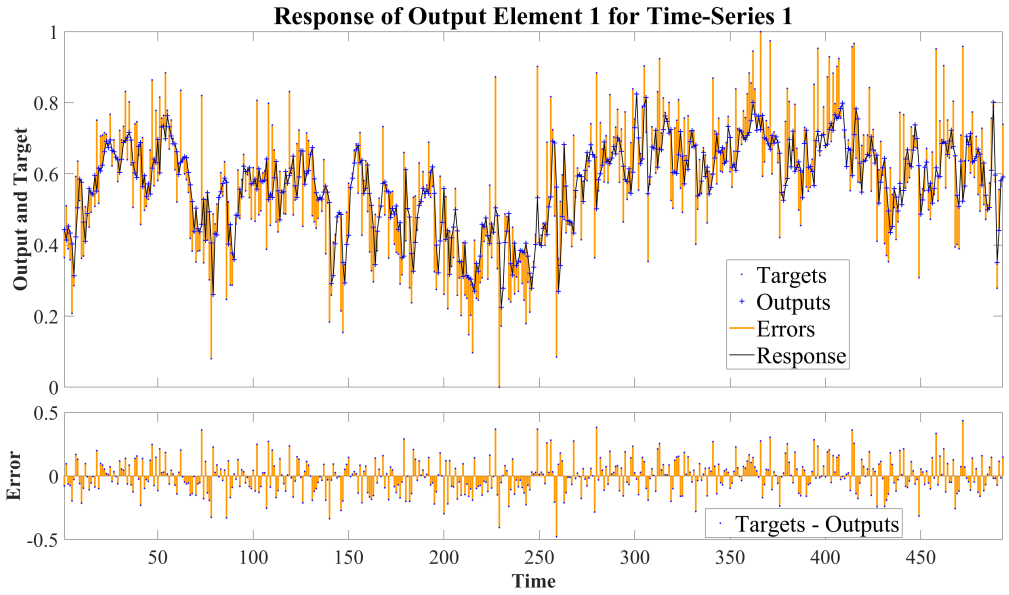


Fig. 4.105 Response

The error histogram shows that the proportion of instances with errors above -0.1388 and 0.1352 is very large, indicating that the model training accuracy is not high. The error autocorrelation coefficients exhibit periodic patterns, and the correlation coefficients between the output and error also show regularity, indicating that the model's simulation effect is not good, which is consistent with the performance metric calculation results. In the regression graph, there is a large error between Y=T and Fit, and the outputs are scattered over a wide range, with R=0.6702. The response graph shows that the response error interval is [-0.5, 0.5], and the response of outputs to targets is very poor, indicating that the model is basically unusable.

From the above, we can see that music games are similar to RPG games in that content changes are strongly related to levels, and the simulation effect of a single level is not good, so the following Fig. 4.106, Fig. 4.107, Fig. 4.108, Fig. 4.109, Fig. 4.110, Fig. 4.111 and Fig. 4.112 are the data for two levels:

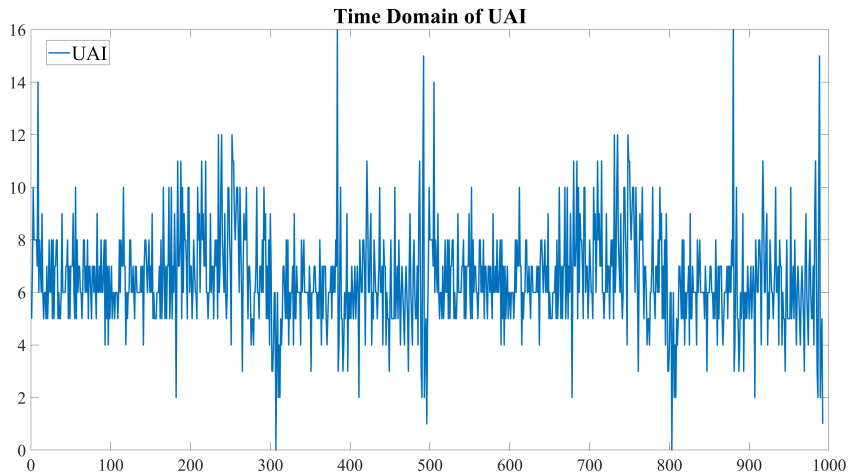


Fig. 4.106 Time Domain of UAI

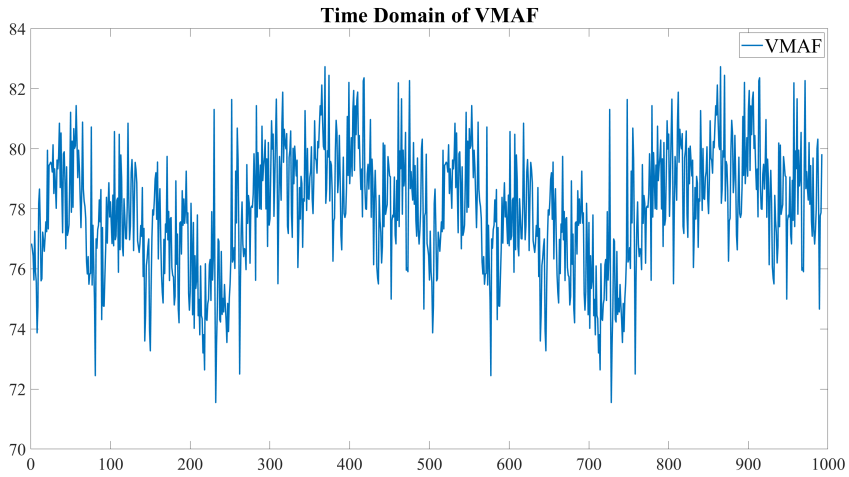


Fig. 4.107 Time Domain of VMAF

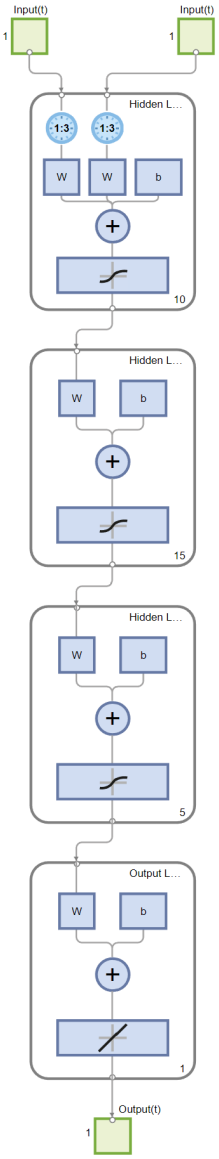


Fig. 4.108 Network for Game 5 Repeat Level

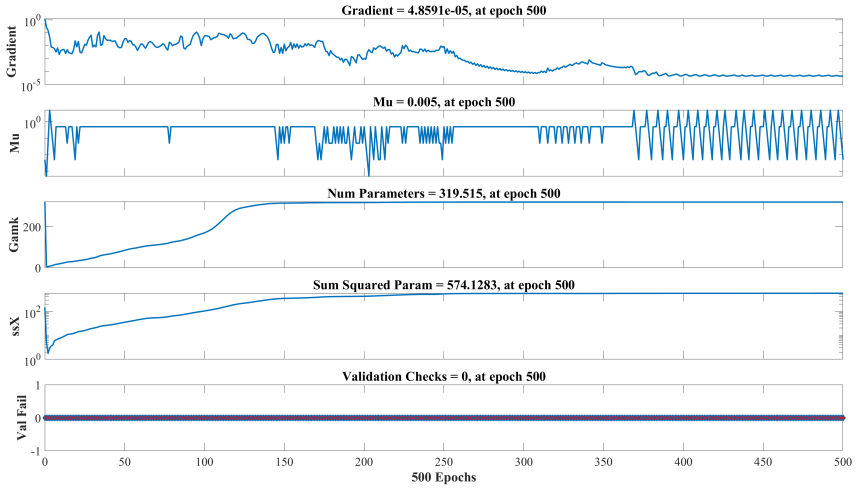


Fig. 4.109 Training States

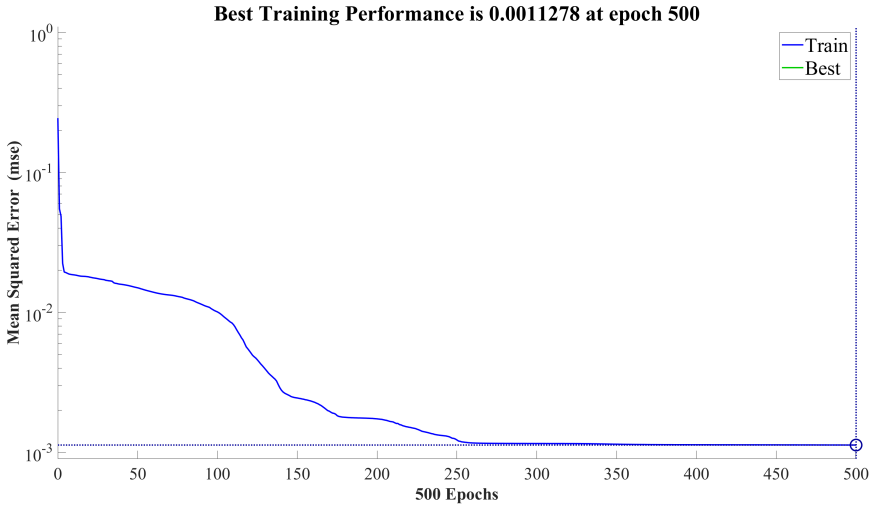


Fig. 4.110 Performance

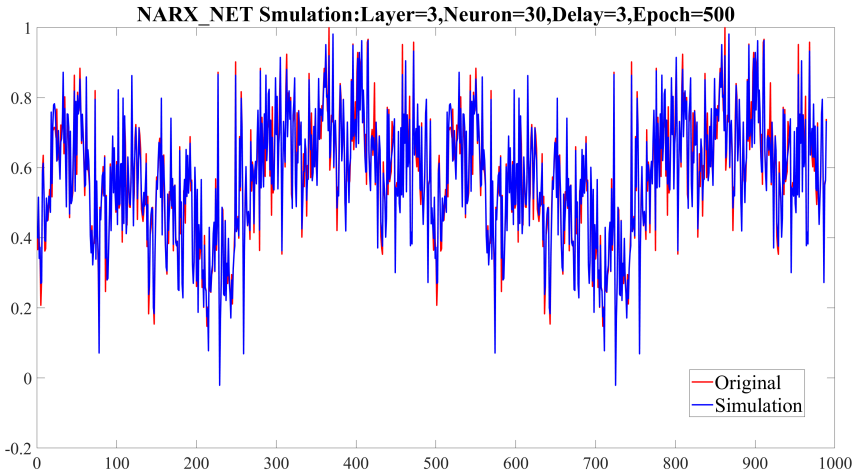


Fig. 4.111 Simulation

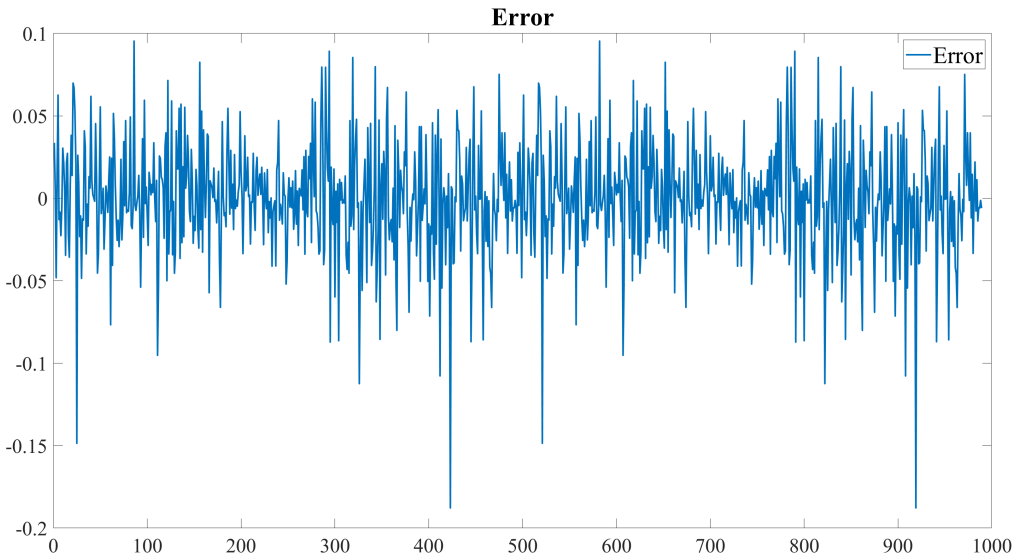


Fig. 4.112 Error

The performance index calculation results of the NARX model are:

narx_net: layer=[10 15 5], Delay=6, Epoch=500, $e_{mean} = 1.5659 \times 10^{-6}$, $e_{max} = 0.1878$, $e_{sum} = 1.1154$, $e_{fit} = 81.25\%$

Compared with the previous section, it is obvious that the performance indicators have been greatly improved. Further analysis of the simulation results shows that the error is as follows Fig. 4.113, Fig. 4.114, Fig. 4.115, Fig. 4.116 and Fig. 4.117:

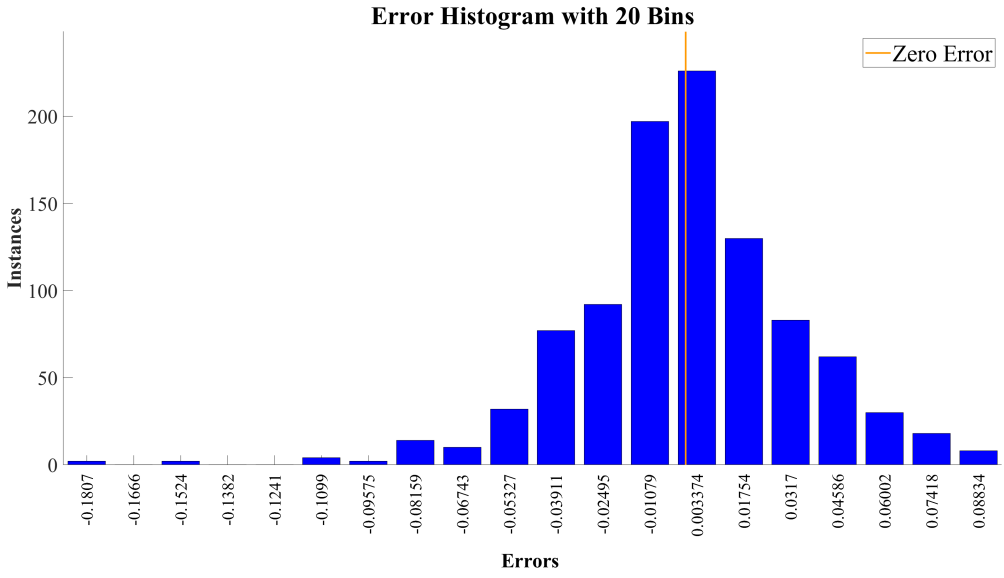


Fig. 4.113 Histogram

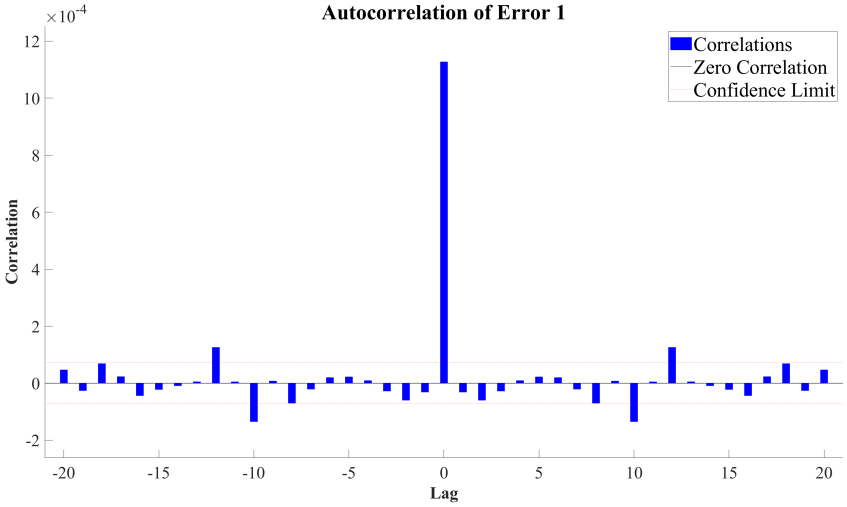


Fig. 4.114 Error Correction

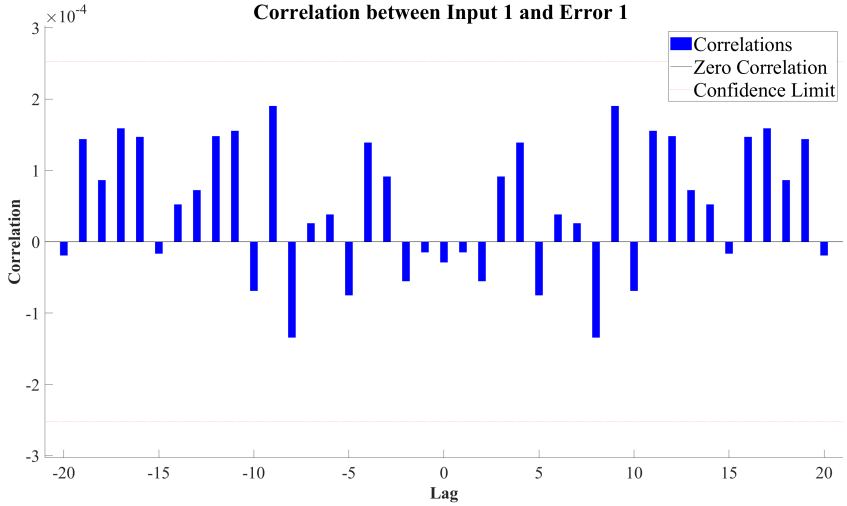


Fig. 4.115 Error Input

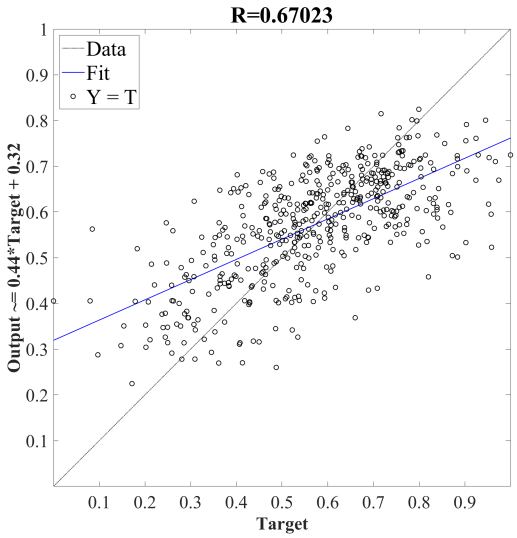


Fig. 4.116 Regression

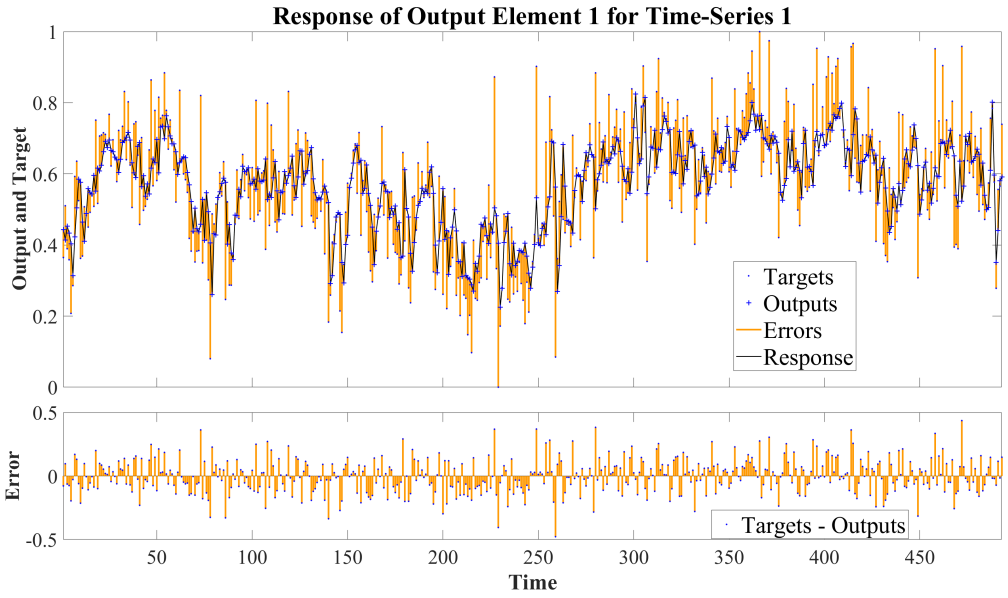


Fig. 4.117 Response

The error histogram, error autocorrelation coefficients, and correlation coefficients between Input and Error all show that the model training effect has significantly improved after enhancement. The regression graph and response graph also confirm this result.

4.7 Summary

Through the content of this chapter, we discovered that UAI and VMAF have a strong non-linear relation. It is impossible to use the ARX/ARMAX model based on linear theory to predict VMAF by UAI. However, the simulation results of the ARX/ARMAX model fully reveal the auto-regressive and dynamic delay features of the UAI/VMAF series. This discovery is undoubtedly valuable for future study. In multiple sets of experimental results, we found that as UAI changes in a complex manner, VMAF also exhibits the same trend of change. In this study, we successfully performed NARX modelling on the UAI and VMAF of multiple datasets and effectively evaluated the model results. Using the obtained model, we substituted the experimental data for verification and conducted simple predictions of VMAF based on UAI data, successfully verifying the possibility of using UAI to predict VMAF. However, because different cloud games result in complex UAI/VMAF relations, some case simulations were suboptimal; we employed data augmentation techniques to expand the dataset and introduced more diverse training samples to enhance the prediction effectiveness. In summary, in this chapter, we successfully achieved the main objectives of this study, successfully analysed the correlation between UAI and VMAF, obtained the corresponding model, and provided a solid theoretical and experimental foundation for future research.

Chapter 5

Conclusions

5.1 Contributions

Cloud gaming is a promising network gaming technology based on cloud computing that allows devices with limited graphic processing and computational power to run high-quality games. In cloud gaming, games are operated on cloud servers, and the rendered video and audio streams are transmitted over the network to the player's gaming terminal. The core of cloud gaming services is built on streaming video platforms, and dynamic video compression technology is critical to reducing costs and enhancing the gaming experience. However, cloud gaming services differ from traditional streaming video platforms in terms of the content transmitted and the interaction with users. Cloud game streaming video is rendered by cloud game servers, and its quality can be predicted based on game content to dynamically adjust compression rates. Additionally, cloud game services are interactive, and both streaming video content and game data processing depend on user behavioural data.

Based on the above characteristics, this study conducted a study on cloud game streaming video quality and user behaviour data, and in order to address the shortcomings of current cloud game services, this study finally establishes a few outcomes as below:

5.1.1 Propose and Development of the User Activity Index (UAI)

This study introduces the User Activity Index (UAI), a novel metric designed to capture and quantify the interactive characteristics of cloud game streaming videos. The UAI takes into account both game content and user behaviour, providing a comprehensive measure of user activity in cloud gaming. An in-depth analysis of user inputs via keyboard, mouse, and controller was conducted across various game content types to understand the relationship between user actions and game dynamics. The analysis involved examining the frequency, intensity, and patterns of user inputs in different gaming scenarios, such as action, strategy, and puzzle games. Based on this comprehensive analysis, a robust method to calculate the UAI was established, considering the specific characteristics of each input device and the impact of different game genres on user activity.

During the experimental phase, several cloud game streaming videos were carefully selected for analysis, ensuring a diverse representation of game types and user interaction styles. A specially designed program was developed to collect and process user input data from these videos. The program efficiently captured and recorded the relevant input events, such as keystrokes, mouse clicks, and controller actions, along with their corresponding timestamps. The collected data were then processed to calculate the UAI values for each video segment, taking into account the specific input devices used and the game content being played. The resulting UAI values provided a quantitative measure of user activity throughout the gaming sessions, reflecting the interactive nature of cloud gaming services and the trends of user engagement for specific game content. This innovative approach to quantifying user activity in cloud gaming lays the foundation for further analysis and optimisation of cloud gaming services based on user behaviour patterns.

5.1.2 Modelling the UAI-VMAF Relationship using Linear Model Theory

In this study, we employed linear model theory to investigate the relationship between the User Activity Index (UAI) and the Video Multimethod Assessment Fusion (VMAF), a comprehensive indicator of video quality. Given that VMAF is influenced by multiple factors, we hypothesised that its variation within a game session follows a stationary stochastic process. To test this hypothesis and establish a predictive model, we analysed the auto-correlation properties of VMAF values over time.

By examining the auto-correlation function (ACF) and partial auto-correlation function (PACF) of the VMAF time series, we determined the appropriate order and parameters for the linear model. The ACF measures the correlation between VMAF values at different time lags, while the PACF helps identify the direct relationship between VMAF values at specific lags, excluding the influence of intermediate values. Based on these analyses, we estimated the coefficients of the linear model using techniques such as least squares estimation or maximum likelihood estimation.

The resulting linear model establishes a mathematical relationship between UAI and VMAF, allowing for the prediction of VMAF values based on the observed UAI. However, experimental results revealed that the relationship between UAI and VMAF exhibits strong non-linearity, indicating that linear model prediction methods may not adequately capture the complex dynamics between user activity and video quality in cloud gaming.

Despite the limitations of linear models in this context, the analysis of auto-correlation properties and the establishment of a linear model provide valuable insights into the temporal dependencies and overall trends in the UAI-VMAF relationship. These findings contribute to a better understanding of the factors influencing video quality in cloud gaming and highlight the need for more advanced modelling techniques to accurately predict VMAF based on user activity patterns.

5.1.3 Justification for using Recurrent Neural Networks

The objective of this study is to establish and elucidate the relationship between the User Activity Index (UAI) during cloud gaming interactions and the VMAF (Video Multimethod Assessment Fusion) score. This relationship provides a foundation for analysing and optimising cloud gaming services based on user behaviour patterns during interactions. Consequently, establishing a quantitative relationship between these two indicators, UAI and VMAF, is the focal point of this thesis.

It is well-known that the simplest and most direct tool for investigating the relationship between two univariate variables is the linear regression method. This was the initial rationale for employing linear model theory to model the UAI/VMAF relationship in this study. However, we discovered that the relationship between the UAI and VMAF sequences is highly complex and varies significantly among different game types (refer to the UAI/VMAF time-domain graphs). This complexity renders the linear model method ineffective in adequately addressing the problem.

The NARX (Non-linear AutoRegressive with eXogenous inputs) model is a type of recurrent neural network (RNN) based on linear model theory, particularly suited for handling time series data. Through learning and training, it can achieve highly complex non-linear mappings and is widely applied in fields such as control, communications, image processing, finance, and healthcare. A search of the Science Citation Index Expanded revealed that over the past five years, more than 800 research articles have utilised the NARX model, with over 200 focusing on time series analysis, providing abundant reference resources for this research.

Moreover, researchers such as Nagabhushan Eswara, Soumen Chakraborty, and Hemanth P. Sethuram have employed NARX to predict the subjective Quality of Experience (QoE) associated with streamed videos, yielding valuable results. By utilising NARX to establish a video quality-aware resource allocation algorithm in cellular networks to enhance users' perceptual QoE, they demonstrated that NARX holds significant potential in the prediction of video streaming quality. [37]

In summary, the primary reason for using the Non-linear AutoRegressive with eXogenous inputs model is its exceptional ability to model dynamic non-linear systems where the current state is influenced by past states and external inputs. The advantages of the NARX model include capturing non-linear relationships, considering historical dependencies, integrating external inputs, and delivering high predictive performance in time-series forecasting tasks. Specifically for analysing streamed video quality, the NARX model excels because video quality is affected by multiple external factors such as network bandwidth, latency, and packet loss rate, which may have complex non-linear relationships. [5]

Additionally, video quality changes dynamically over time, influenced not only by current network conditions but also by previous network states and video encoding methods. At the same time, due to the impact of user activity interference in cloud gaming, we need to model different game scenarios for different players separately to make the model more targeted. Compared with other models, NARX has lower resource consumption, which aligns with our ultimate goal of

reducing cloud gaming operating costs and can also better improve the quality of cloud gaming services for players.

Furthermore, the integration of NARX and other models is one of our future research directions, aiming to provide an enhanced experience for cloud gaming service providers and gamers. By leveraging historical data and external inputs, the NARX model can accurately predict trends in video quality, helping to anticipate potential declines or improvements. This precise modelling enables service providers to optimise streaming strategies, such as adaptive bitrate adjustments, thereby enhancing the overall user viewing experience and improving the quality of cloud gaming services.

5.1.4 Modelling the UAI-VMAF Relationship using Recurrent Neural Networks

To address the limitations of linear models in capturing the complex relationship between the User Activity Index (UAI) and the Video Multimethod Assessment Fusion (VMAF), we explored the use of recurrent neural networks (RNNs) for modelling and prediction. Specifically, we employed the Non-linear AutoRegressive with eXogenous inputs (NARX) model, a type of RNN architecture that incorporates external input variables alongside the autoregressive terms. The NARX model assumes that the variation of VMAF within a game session follows a stationary stochastic process, allowing for the modelling of temporal dependencies and non-linear relationships.

To establish the NARX models, we collected data from different types of games, ensuring a diverse representation of user activity patterns and video quality variations. The data included UAI values as exogenous inputs and corresponding VMAF values as the target variable. One crucial aspect of NARX modelling is the selection of appropriate delay parameter settings. Due to the dynamic nature of UAI and VMAF sequence data, the choice of delay parameters significantly impacts the model's ability to capture relevant temporal dependencies. We conducted a thorough analysis of the design principles for time series delay selection, considering factors such as the autocorrelation structure, cross-correlation between UAI and VMAF, and the trade-off between model complexity and predictive performance. Based on these principles, we developed guidelines for selecting optimal delay settings in NARX models for UAI-VMAF prediction. Experimental results demonstrated that the NARX model, when configured with appropriate delay parameters, can effectively address the UAI-VMAF prediction problem.

The model's ability to capture non-linear relationships and account for the temporal dynamics of user activity and video quality led to improved prediction accuracy compared to linear models. The NARX model successfully learned the underlying patterns and dependencies in the data, enabling accurate forecasting of VMAF values based on the observed UAI. The successful application of recurrent neural networks, particularly the NARX model, in modelling the UAI-VMAF relationship highlights the potential of advanced machine learning techniques in the context of cloud gaming. By leveraging the power of RNNs to capture complex temporal dependencies and non-linear relationships, we can develop more accurate and reliable models

for predicting video quality based on user activity patterns. These findings pave the way for the development of intelligent cloud gaming systems that can dynamically adapt to user behaviour and optimise video quality in real-time.

5.1.5 Experimental Evaluation and Analysis of the Model and Result

In the experimental stage of this study, we conducted a comprehensive evaluation and analysis of the proposed models to predict the relationship between the User Activity Index (UAI) and the Video Multimethod Assessment Fusion (VMAF) in cloud gaming. To begin, we defined a set of metrics to assess the prediction performance of the models, including the mean of errors, the maximum absolute error, the sum of accumulated error, and the percentage of agreement between the model's calculated outputs and actual outputs. These metrics show not only the model's usability but also some details of individual abnormal samples, as well as the confidence when it is used to make predictions.

To ensure a diverse and representative dataset, we selected multiple cloud gaming streaming videos for analysis, covering different game genres such as action, strategy, and card games. These videos were carefully chosen to capture a wide range of user activity patterns and video quality variations. We developed a user input collection program based on Windows interrupt capture technology to accurately record user interactions, including keyboard, mouse, and controller inputs. The collected input data were then processed and converted into UAI values, providing a quantitative measure of user activity throughout the gaming sessions.

To simulate the actual network environment of cloud gaming services, we subjected the video samples to different compression bitrates. By comparing the compressed video data with the original video samples, we calculated the corresponding VMAF values, which served as the ground truth for evaluating the prediction models. This step allowed us to assess the impact of compression on video quality and examine the relationship between UAI and VMAF under realistic network conditions. By combining the calculated VMAF values with the corresponding UAI data for each game segment, we successfully verified the significant impact of user behaviour on the quality of cloud gaming streaming videos. This analysis revealed that variations in user activity patterns, as captured by UAI, have a direct influence on the perceived video quality, as measured by VMAF.

To evaluate the effectiveness of the proposed models, we employed the NARX model for UAI-VMAF prediction, considering different game types and compression rates. We carefully selected appropriate NARX parameters, such as the number of hidden layers, neurons per layer, and delay settings, based on the specific characteristics of each game genre and compression scenario. The NARX models were trained using the collected UAI-VMAF data, and their prediction performance was assessed using the defined evaluation metrics.

The experimental results highlighted the challenges associated with predicting VMAF in action games, where the dense control inputs led to relatively gradual changes in UAI. Consequently, the periodicity of the corresponding VMAF variation pattern was less significant, making it more difficult for the NARX model to capture the underlying relationships accurately.

As a result, the initial training and simulation effects of the NARX model for action games were suboptimal. To address this issue, we employed data augmentation techniques to expand the dataset, introduce more diverse training samples, and enhance the variability and robustness of the training data. The augmented dataset allowed the NARX model to learn more generalisable patterns and improve its prediction performance for action games. After incorporating data augmentation, the NARX model's performance improved considerably, demonstrating its ability to capture the complex relationships between UAI and VMAF across different game genres and compression scenarios. The experimental evaluation and analysis conducted in this study underscore the importance of considering game-specific characteristics and video data quality when developing predictive models for cloud gaming. The findings highlight the potential of advanced machine learning techniques, such as recurrent neural networks, in capturing the intricate dynamics between user behaviour and video quality. By leveraging these insights, cloud gaming service providers can optimise their systems to deliver high-quality gaming experiences while efficiently managing network resources.

Furthermore, the successful application of data augmentation techniques in improving the prediction performance for action games demonstrates the significance of diverse and representative training data. This finding emphasises the need for continuous data collection and refinement to enhance the robustness and generalisability of predictive models in cloud gaming. Overall, the experimental evaluation and analysis conducted in this study provide valuable insights into the relationship between UAI and VMAF in cloud gaming and lay the foundation for the development of intelligent and adaptive cloud gaming systems. By accurately predicting video quality based on user activity patterns, service providers can proactively optimise their streaming strategies, ensuring a seamless and immersive gaming experience for users across various game genres and network conditions.

5.2 Limitations and Further Improvements

In this comprehensive study, we have successfully established and demonstrated the intricate relationship between the User Activity Index (UAI) and the Video Multimethod Assessment Fusion (VMAF) in the context of cloud gaming. Moreover, we have conducted preliminary predictions of VMAF using UAI, showcasing the potential of this approach. However, considering the inherent complexity and unique characteristics of cloud gaming video content, there are several key aspects that warrant further research and exploration to enhance the accuracy and effectiveness of our findings.

Firstly, it is crucial to acknowledge that VMAF, a metric developed by Netflix for mainstream streaming video, may lack the necessary specificity to accurately assess the quality of cloud gaming video content. In cloud game streaming services, we have the ability to capture a vast amount of game video rendering data generated by the server in real-time. This rendering data can provide a more precise reflection of the changes in-game video content and the associated variations in video quality. By leveraging the characteristics of the original game engine rendering

data, we can develop a new type of video quality parameter tailored specifically for gaming, which could potentially replace VMAF. This customised metric would more effectively and accurately represent the changes in video content quality, taking into account the unique dynamics and requirements of cloud gaming.

Secondly, in line with the previous point, the processing of compressed videos in our study still relies on common codecs used by mainstream streaming video platforms. However, given the distinct and regular characteristics of cloud gaming content, there is an opportunity to develop a new compression format designed specifically for cloud gaming video content. By integrating a large amount of game-rendering original data into the compression process, we can further optimise and enhance the video compression quality. This targeted approach would allow for more efficient and effective compression, ultimately improving the overall user experience and reducing the bandwidth requirements for cloud gaming services.

Thirdly, this thesis represents a pioneering effort in the field of cloud gaming service research, as it introduces the concept and definition of UAI. While our study provides a preliminary definition and analysis of UAI, it is important to recognise that different game contents exhibit various characteristics that can influence user activity patterns. To further refine and improve the calculation methods for UAI, future research should focus on conducting more in-depth analyses based on a wider range of game characteristics. By examining user activity across diverse game genres, gameplay mechanics, and player demographics, we can develop more robust and reliable methods for quantifying UAI. This expanded analysis will enhance the accuracy and validity of UAI data, enabling more precise predictions and optimisations in cloud gaming services.

Lastly, it is essential to acknowledge that different game video contents exhibit distinct patterns of change, which can pose challenges for the Non-linear AutoRegressive with eXogenous inputs (NARX) model in accurately reflecting the complex content changes across various game types. To address this issue, future studies should explore the application of diverse modelling methods tailored to specific game characteristics. By employing targeted analysis techniques and segmenting game sessions based on length and content, we can conduct detailed analyses of streaming video for different game types. This refined approach will allow for a more nuanced understanding of the relationship between UAI and VMAF, capturing the unique dynamics and patterns of change within each game category. By adapting our modelling methods to the specific requirements of different game types, we can enhance the accuracy and effectiveness of our predictions, ultimately leading to more efficient and optimised cloud gaming services.

In conclusion, while our study has made significant strides in establishing the relationship between UAI and VMAF in cloud gaming, there remain several avenues for further research and refinement. By developing gaming-specific video quality metrics, optimising compression formats, expanding UAI analysis across diverse game characteristics, and employing targeted modelling methods, we can continue to push the boundaries of cloud gaming service optimisation. Through ongoing research and innovation in these areas, we can unlock the full potential of cloud gaming, delivering seamless, high-quality experiences to players worldwide.

5.3 Summary

This study explores the dynamic landscape of cloud gaming technology and introduces a novel predictive model for cloud game video quality. The model aims to forecast future video quality trends by analysing data from simulated interactions of cloud game players and video quality parameters from cloud game streaming. The research methodology involved the collection and analysis of multiple data sets, leading to the establishment of the Linear Time Series Analysis Model and the Recurrent Neural Networks Model. These models were compared and tailored to suit the specific demands of different game genres based on varied user behaviours across different gaming scenarios.

The effectiveness of these models has been confirmed through rigorous simulation experiments across diverse gaming environments, demonstrating their capability to accurately predict fluctuations in cloud game streaming video quality. The results substantiate the applicability and precision of the model for predicting the quality of cloud game video streams, offering strategies to dynamically optimise video quality in response to user behaviour. The potential applications of this predictive model extend beyond cloud gaming to encompass streaming media services, providing a theoretical foundation for the future evolution and enhancement of cloud gaming services.

The findings of this research have practical implications for enhancing the quality and efficiency of cloud gaming services and hold significant theoretical value, contributing to the broader discourse on cloud gaming technology advancement.

References

[1] Aaron, A., Li, Z., Manohara, M., Lin, J. Y., Wu, E. C.-H. and Kuo, C.-C. J. [2015], 'Challenges in cloud based ingest and encoding for high quality streaming media', *2015 IEEE International Conference on Image Processing (ICIP)* **1**.

[2] Alhilal, A., Braud, T., Han, B. and Hui, P. [2022], 'Nebula: Reliable low-latency video transmission for mobile cloud gaming'.

[3] Avramova, Z., Vleeschauwer, D., Laevens, K., Wittevrongel, S. and Bruneel, H. [2008], 'Modelling h.264/avc vbr video traffic: comparison of a markov and a self-similar source model', *Telecommunication systems/Telecommunications systems* **39**, 91–102.

[4] Bampis, C. G., Li, Z. and Bovik, A. C. [2019], 'Spatiotemporal feature integration and model fusion for full reference video quality assessment', *IEEE Transactions on Circuits and Systems for Video Technology* **29**(8), 2256–2270.

[5] Bampis, C. G., Li, Z., Katsavounidis, I. and Bovik, A. C. [2018], 'Recurrent and dynamic models for predicting streaming video quality of experience', *IEEE Transactions on Image Processing* **27**(7), 3316–3331.

[6] Bampis, C. G., Li, Z., Katsavounidis, I., Huang, T.-Y., Ekanadham, C. and Bovik, A. C. [2018], 'Towards perceptually optimized end-to-end adaptive video streaming', *ArXiv (Cornell University)* .

[7] Bampis, C. G., Li, Z., Moorthy, A. K., Katsavounidis, I., Aaron, A. and Bovik, A. C. [2017], 'Study of temporal effects on subjective video quality of experience', *IEEE Transactions on Image Processing* **26**(11), 5217–5231.

[8] Barman, N. and Martini, M. G. [2019], 'Qoe modeling for http adaptive video streaming–a survey and open challenges', *IEEE Access* **7**, 30831–30859.

[9] Barman, N., Schmidt, S., Zadtootaghaj, S., Martini, M. G. and Möller, S. [2018], 'An evaluation of video quality assessment metrics for passive gaming video streaming', *Proceedings of the 23rd Packet Video Workshop* **1**.

[10] Bentaleb, A., Taani, B., Begen, A. C., Timmerer, C. and Zimmermann, R. [2019], 'A survey on bitrate adaptation schemes for streaming media over http', *IEEE Communications Surveys and Tutorials* **21**, 562–585.

[11] Bestagini, P., Battaglia, S., Milani, S., Tagliasacchi, M. and Tubaro, S. [2013], 'Detection of temporal interpolation in video sequences', *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on* pp. 3033–3037.

[12] Biernacki, A. [2016], 'Analysis and modelling of traffic produced by adaptive http-based video', *Multimedia tools and applications* **76**, 12347–12368.

[13] Billings, S. A. [2013], *Nonlinear System Identification*, John Wiley and Sons.

[14] Bishnoi, A. [2020], *All About Cloud Gaming*, Anirudh Bishnoi.

[15] Blazakis, K., Katsigiannis, Y. A. and Stavrakakis, G. [2022], 'One-day-ahead solar irradiation and windspeed forecasting with advanced deep learning techniques', *Energies* **15**, 4361–4361.

[16] Blog, N. T. [2017], 'Toward a practical perceptual video quality metric'.
URL: *https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652*

[17] Bovik, A. C. [2005], *Handbook of Image and Video Processing*, Academic Press.

[18] Bowman, B., Elmqvist, N. and Jankun-Kelly, T. [2012], 'Toward visualization for games: Theory, design space, and patterns', *IEEE Transactions on Visualization and Computer Graphics* **18**(11), 1956–1968.

[19] Cai, W., Shea, R., Huang, C.-Y., Chen, K.-T., Liu, J., Leung, V. C. M. and Hsu, C.-H. [2016*a*], 'A survey on cloud gaming: Future of computer games', *IEEE Access* **4**, 7605–7620.

[20] Cai, W., Shea, R., Huang, C.-Y., Chen, K.-T., Liu, J., Leung, V. C. M. and Hsu, C.-H. [2016*b*], 'A survey on cloud gaming: Future of computer games', *IEEE Access* **4**, 7605–7620.

[21] Capel, M. I. [2022], 'Artificial neuron-based model for a hybrid real-time system: Induction motor case study', *Mathematics* **10**(18).
URL: *https://www.mdpi.com/2227-7390/10/18/3410*

[22] Cassidy, G. G. and Paisley, A. M. J. [2013], 'Music-games: a case study of their impact', *Research Studies in Music Education* **35**, 119–138.

[23] Chan, K. L., Ichikawa, K., Watashiba, Y., Putchong, U. and Iida, H. [2017], 'A hybrid-streaming method for cloud gaming: To improve the graphics quality delivered on highly accessible game contents', *International Journal of Serious Games* **4**.

[24] Chen, K.-T., Chang, Y.-C., Hsu, H.-J., Chen, D.-Y., Huang, C.-Y. and Hsu, C.-H. [2014], 'On the quality of service of cloud gaming systems', *IEEE Transactions on Multimedia* **16**(2), 480–495.

[25] Chen, K.-T., Chang, Y.-C., Tseng, P.-H., Huang, C.-Y. and Lei, C.-L. [2011], 'Measuring the latency of cloud gaming systems', *Proceedings of the 19th ACM international conference on Multimedia - MM '11* **1**.

[26] Chen, Y., Wang, B. and Liu, K. [2009], 'Multiuser rate allocation games for multimedia communications', *IEEE Transactions on Multimedia* **1**.

[27] Cheng, X., Wang, C. and Chen, S. [1995], *Neural Network Principles and Applications*, National Defence Industry Publishing House.
URL: *https://books.google.co.uk/books?id=mD1sAAAACAAJ*

[28] Chikkerur, S., Sundaram, V., Reisslein, M. and Karam, L. J. [2011], 'Objective video quality assessment methods: A classification, review, and performance comparison', *IEEE Transactions on Broadcasting* **57**(2), 165–182.

[29] Claypool, K. T. and Claypool, M. [2007], 'On frame rate and player performance in first person shooter games', *Multimedia Systems* **13**, 3–17.

[30] Claypool, M. and Claypool, K. [2006], 'Latency and player actions in online games', *Commun. ACM* **49**(11), 40–45.
URL: *https://doi.org/10.1145/1167838.1167860*

[31] Claypool, M. and Finkel, D. [2014], The effects of latency on player performance in cloud-based games, *in* '2014 13th Annual Workshop on Network and Systems Support for Games', pp. 1–6.

[32] Conroy, E., Toth, A. J. and Campbell, M. J. [2022], 'The effect of computer mouse mass on target acquisition performance among action video gamers', *Applied Ergonomics* **99**, 103637.

[33] Deng, S., Han, J. and Xu, Y. [2020], 'Vmaf based rate-distortion optimization for video coding', *1* **1**.

[34] Dobrian, F., Sekar, V., Awan, A., Stoica, I., Joseph, D., Ganjam, A., Zhan, J. and Zhang, H. [2011], 'Understanding the impact of video quality on user engagement', *Proceedings of the ACM SIGCOMM 2011 conference on SIGCOMM - SIGCOMM '11* **1**.

[35] Doggen, J. and Filip, V. d. S. [2008], *Design and Simulation of a H.264 AVC Video Streaming Model*, European Conference on the Use of Modern Information and Communication Technologies.

[36] Duanmu, Z., Zeng, K., Ma, K., Rehman, A. and Wang, Z. [2017], 'A quality-of-experience index for streaming video', *IEEE Journal of Selected Topics in Signal Processing* **11**(1), 154–166.

[37] Eswara, N., Chakraborty, S., Sethuram, H. P., Kuchi, K., Kumar, A. and Channappayya, S. S. [2020], 'Perceptual qoe-optimal resource allocation for adaptive video streaming', *IEEE Transactions on Broadcasting* **66**(2), 346–358.

[38] Feit, A., Weir, D. and Oulasvirta, A. [2016], 'How we type: Movement strategies and performance in everyday typing', *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* **16**.

[39] Forlines, C., Wigdor, D., Shen, C. and Balakrishnan, R. [2007], 'Direct-touch vs. mouse input for tabletop displays', *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '07* **1**.
URL: *https://dl.acm.org/citation.cfm?id=1240726*

[40] Gao, M., Wu, Q., Li, J., Wang, B., Zhou, Z., Liu, C. and Wang, D. [2023], 'Temperature prediction of solar greenhouse based on narx regression neural network', *Scientific Reports* **13**, 1563.
URL: *https://www.nature.com/articles/s41598-022-24072-1*

[41] García, B., López-Fernández, L., Gortázar, F. and Gallego, M. [2019], 'Practical evaluation of vmaf perceptual video quality for webrtc applications', *Electronics* **8**, 854.

[42] Gerling, K. M., Klauser, M. and Niesenhaus, J. [2011], 'Measuring the impact of game controllers on player experience in fps games', *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11* **1**.

[43] Green, C. and Bavelier, D. [2007], 'Action-video-game experience alters the spatial resolution of vision', *Psychological Science* **18**, 88–94.
URL: *https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2896830/*

[44] Grois, D., Marpe, D., Mulayoff, A., Itzhaky, B. and Hadar, O. [2013], Performance comparison of h.265/mpeg-hevc, vp9, and h.264/mpeg-avc encoders, *in* '2013 Picture Coding Symposium (PCS)', pp. 394–397.

[45] Gunawan, I. P. and Ghanbari, M. [2008], 'Reduced-reference video quality assessment using discriminative local harmonic strength with motion consideration', *IEEE Transactions on Circuits and Systems for Video Technology* **18**(1), 71–83.

[46] Göring, S., Rao, R. R. R., Feiten, B. and Raake, A. [2021], 'Modular framework and instances of pixel-based video quality models for uhd-1/4k', *IEEE Access* **9**, 31842–31864.

[47] Hamilton, J. D. [1994], *Time series analysis*, Princeton University Press.

[48] He, Z. and Chen, C. W. [2002], End-to-end video quality analysis and modeling for video streaming over ip network, *in* 'Proceedings. IEEE International Conference on Multimedia and Expo', Vol. 1, pp. 853–856 vol.1.

[49] Hegazy, M., Diab, K., Saeedi, M., Ivanovic, B., Amer, I., Liu, Y., Sines, G. and Hefeeda, M. [2019], 'Content-aware video encoding for cloud gaming', *Proceedings of the 10th ACM Multimedia Systems Conference* **1**.

[50] Hermansson, P. [2011], 'Optimizing an h.264 video encoder for real-time hd-video encoding', *Optimizing an H.264 video encoder for real-time HD-video encoding* **1**.

[51] Holbert, B. and Huber, M. [2008], 'Design and evaluation of haptic effects for use in a computer desktop for the physically disabled', *Universal Access in the Information Society* **10**, 165–178.

[52] Hopfield, J. J. [1982], 'Neural networks and physical systems with emergent collective computational abilities.', *Proceedings of the National Academy of Sciences* **79**, 2554–2558.
URL: *https://www.pnas.org/content/79/8/2554*

[53] Huang, C.-Y., Hsu, C.-H., Chang, Y.-C. and Chen, K.-T. [2013], 'Gaminganywhere', *Proceedings of the 4th ACM Multimedia Systems Conference on - MMSys '13* **1**.
URL: *https://dl.acm.org/citation.cfm?id=2483981*

[54] Huynh-Thu, Q. and Ghanbari, M. [2008*a*], 'Scope of validity of psnr in image/video quality assessment', *Electronics Letters* **44**(13), 1–2. - Copyright The Institution of Engineering and Technology Jun 19, 2008; CODEN - ELLEAK.
URL: *https://www.proquest.com/scholarly-journals/scope-validity-psnr-image-video-quality/docview/1625957339/se-2*

[55] Huynh-Thu, Q. and Ghanbari, M. [2008*b*], 'Scope of validity of psnr in image/video quality assessment', *Electronics Letters* **44**, 800.

[56] Jarschel, M., Schlosser, D., Scheuring, S. and Hoßfeld, T. [2011], An evaluation of qoe in cloud gaming based on subjective tests, *in* '2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing', pp. 330–335.

[57] Jarschel, M., Schlosser, D., Scheuring, S. and Hoßfeld, T. [2013], 'Gaming in the clouds: Qoe and the users' perspective', *Mathematical and Computer Modelling* **57**, 2883–2894.

[58] Jiang, J., Sekar, V. and Zhang, H. [2012], 'Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive', *Proceedings of the 8th international conference on Emerging networking experiments and technologies - CoNEXT '12* **1**.

[59] Khan, A., Sun, L. and Ifeachor, E. [2009], 'Content clustering based video quality prediction model for mpeg4 video streaming over wireless networks', *2009 IEEE International Conference on Communications* **1**.

[60] Kougioumtzidis, G., Poulkov, V., Zaharis, Z. D. and Lazaridis, P. I. [2022], 'A survey on multimedia services qoe assessment and machine learning-based prediction', *IEEE Access* **10**, 19507–19538.

[61] Krishnan, S. S. and Sitaraman, R. K. [2013], 'Video stream quality impacts viewer behavior: Inferring causality using quasi-experimental designs', *IEEE/ACM Transactions on Networking* **21**, 2001–2014.

[62] Laghari, A. A., He, H., Memon, K. A., Laghari, R. A., Halepoto, I. A. and Khan, A. [2019], 'Quality of experience (qoe) in cloud gaming models: A review', *Multiagent and Grid Systems* **15**, 289–304.

[63] Lampe, U., Wu, Q., Dargutev, S., Hans, R., Miede, A. and Steinmetz, R. [n.d.], 'Assessing latency in cloud gaming'.

[64] Liao, X., Lin, L., Tan, G., Jin, H., Yang, X., Zhang, W. and Li, B. [2016], 'Liverender: A cloud gaming system based on compressed graphics streaming', *IEEE/ACM Transactions on Networking* **24**(4), 2128–2139.

[65] Lin, Y. and Shen, H. [2017], 'Cloudfog: Leveraging fog to extend cloud gaming for thin-client mmog with high quality of service', *IEEE Transactions on Parallel and Distributed Systems* **28**(2), 431–445.

[66] Liu, L., Liu, B., Huang, H. and Bovik, A. C. [2014], 'No-reference image quality assessment based on spatial and spectral entropies', *Signal Processing: Image Communication* **29**, 856–863.

[67] Liu, Y., Li, Z. and Soh, Y. C. [2008], 'Region-of-interest based resource allocation for conversational video communication of h.264/avc', *IEEE Transactions on Circuits and Systems for Video Technology* **18**, 134–139.

[68] Liu, Y., Wang, S. and Dey, S. [2012], 'Modeling, characterizing, and enhancing user experience in cloud mobile rendering', *2012 International Conference on Computing, Networking and Communications (ICNC)* **1**.

[69] Luo, Z., Zhu, C., Huang, Y., Xie, R., Song, L. and Kuo, C.-C. J. [2021], 'Vmaf oriented perceptual coding based on piecewise metric coupling', *IEEE transactions on image processing* **30**, 5109–5121.

[70] Ma, L., Li, S., Zhang, F. and Ngan, K. N. [2011], 'Reduced-reference image quality assessment using reorganized dct-based image representation', *IEEE Transactions on Multimedia* **13**, 824–829.

[71] Mahmou, A. [2020], 'Single image super-resolution algorithm using psnr in the wavelet domain', *Journal of Advanced Research in Dynamical and Control Systems* **12**, 677–691.

[72] Manzano, M., Urueña, M., Sužnjević, M., Calle, E., Hernández, J. A. and Matijasevic, M. [2014], 'Dissecting the protocol and network traffic of the onlive cloud gaming platform', *Multimedia Systems* **20**, 451–470.

[73] Mao, H., Netravali, R. and Alizadeh, M. [2017], Neural adaptive video streaming with pensieve, *in* 'Proceedings of the Conference of the ACM Special Interest Group on Data Communication', SIGCOMM '17, Association for Computing Machinery, New York, NY, USA, p. 197–210.
URL: *https://doi.org/10.1145/3098822.3098843*

[74] Marpe, D., Wiegand, T. and Gordon, S. [2005], H.264/mpeg4-avc fidelity range extensions: tools, profiles, performance, and application areas, *in* 'IEEE International Conference on Image Processing 2005', Vol. 1, pp. I–593.

[75] MathWorks [2024], 'Home - matlab central'.
URL: *https://www.mathworks.cn/matlabcentral/*

[76] Matla, Y., Yannamaneni, R. R. and Pappas, G. [2024], 'Globalizing food items based on ingredient consumption', *Sustainability* **16**(17).
URL: *https://www.mdpi.com/2071-1050/16/17/7524*

[77] Mendi, E. [2015], 'Image quality assessment metrics combining structural similarity and image fidelity with visual attention', *Journal of Intelligent and Fuzzy Systems* **28**, 1039–1046.

[78] Min, X., Duan, H., Sun, W., Zhu, Y. and Zhai, G. [2024], 'Perceptual video quality assessment: a survey', *Science China Information Sciences* **67**.

[79] Mittal, A., Moorthy, A. K. and Bovik, A. C. [2012*a*], 'No-reference image quality assessment in the spatial domain', *IEEE Transactions on Image Processing* **21**, 4695–4708.
URL: *https://dl.acm.org/citation.cfm?id=2711832*

[80] Mittal, A., Moorthy, A. K. and Bovik, A. C. [2012*b*], 'No-reference image quality assessment in the spatial domain', *IEEE Transactions on Image Processing* **21**(12), 4695–4708.

[81] Moorthy, A. K. and Bovik, A. C. [2011], 'Blind image quality assessment: From natural scene statistics to perceptual quality', *IEEE Transactions on Image Processing* **20**(12), 3350–3364.

[82] Moorthy, A. K., Choi, L. K., Bovik, A. C. and de Veciana, G. [2012], 'Video quality assessment on mobile devices: Subjective, behavioral and objective studies', *IEEE Journal of Selected Topics in Signal Processing* **6**(6), 652–671.

[83] Mozhaeva, A., Streeter, L., Vlasuyk, I. and Potashnikov, A. [2021], 'Full reference video quality assessment metric on base human visual system consistent with psnr', *DOAJ (DOAJ: Directory of Open Access Journals)* **1**.

[84] Naima, S., Bourenane, M. and Douga, Y. [2023], 'Deep reinforcement learning-based approach for video streaming: Dynamic adaptive video streaming over http', *Applied Sciences* **13**, 11697.

[85] Napoli, R. and Piroddi, L. [2010], 'Nonlinear active noise control with narx models', *IEEE Transactions on Audio, Speech, and Language Processing* **18**(2), 286–295.

[86] Natapov, D. and MacKenzie, I. S. [2010], 'The trackball controller', *Proceedings of the International Academic Conference on the Future of Game Design and Technology - Futureplay '10* **1**.

[87] Nelles, A. O. [2001], *Nonlinear system identification : from classical approaches to neural networks and fuzzy models*, Springer.

[88] Nogueira, P. A., Teófilo, L. F. and Silva, P. B. [2015], 'Multi-modal natural interaction in game design: a comparative analysis of player experience in a large scale role-playing game', *Journal on Multimodal User Interfaces* **9**, 105–119.

[89] Normoyle, A., Guerrero, G. and Jörg, S. [2014], 'Player perception of delays and jitter in character responsiveness', *CiteSeer X (The Pennsylvania State University)* **1**.

[90] Park, S.-J., Myung, J.-S., Sim, D.-G. and Oh, S.-J. [2008], 'A fast macroblock mode decision method using psnr prediction for h.264/avc', *Journal of Broadcast Engineering* **13**, 137–151.

[91] Peñaherrera-Pulla, O. S., Baena, C., Fortes, S., Baena, E. and Barco, R. [2021], 'Measuring key quality indicators in cloud gaming: Framework and assessment over wireless networks', *Sensors* **21**, 1387.

[92] Pinson, M. and Wolf, S. [2004], 'A new standardized method for objectively measuring video quality', *IEEE Transactions on Broadcasting* **50**(3), 312–322.

[93] Pistonesi, S., Martinez, J., Ojeda, S. M. and Vallejos, R. [2018], 'Structural similarity metrics for quality image fusion assessment: Algorithms', *Image Processing On Line* **8**, 345–368.

[94] Piórkowski, R., Mantiuk, R. and Siekawa, A. [2017], 'Automatic detection of game engine artifacts using full reference image quality metrics', *ACM Transactions on Applied Perception* **14**, 1–17.

[95] Prevratil, M. J., Harwell, K. W., Boot, W. R. and Towne, T. J. [2022], 'Investigating the behavioral mechanisms of action video game effects in a complex transfer task', *Acta Psychologica* **230**, 103718.
URL: *https://www.sciencedirect.com/science/article/pii/S0001691822002335*

[96] Pro [2007], *Methods for evaluating games: how to measure usability and user experience in games?*, Association for Computing Machinery.
URL: *https://doi.org/10.1145/1255047.1255142*

[97] Pro [2008], *Motivations for play in computer role-playing games*, Association for Computing Machinery.
URL: *https://doi.org/10.1145/1496984.1496995*

[98] QIAN, F., GUO, J., SUN, T. and WANG, T.-f. [2015], 'Multi-scale ssim algorithm based on wavelet transforms', *Chinese Journal of Liquid Crystals and Displays* **30**, 317–325.

[99] Qiu, F. and Cui, Y. [2010], 'An analysis of user behavior in online video streaming', *Proceedings of the international workshop on Very-large-scale multimedia corpus, mining and retrieval - VLS-MCMR '10* **1**.

[100] Rassool, R. [2017], 'Vmaf reproducibility: Validating a perceptual practical video quality metric'.
URL: *https://ieeexplore.ieee.org/document/7986143*

[101] Rehman, A., Rostami, M., Wang, Z., Brunet, D. and Vrscay, E. R. [2012], 'Ssim-inspired image restoration using sparse representation', *EURASIP Journal on Advances in Signal Processing* **2012**.

[102] Rehman, A. and Wang, Z. [2012], 'Reduced-reference image quality assessment by structural similarity estimation', *IEEE Transactions on Image Processing* **21**, 3378–3389.

[103] Rosenblatt, F. [1958], 'The perceptron: A probabilistic model for information storage and organization in the brain.', *Psychological Review* **65**, 386–408.

[104] Sabet, S. S., Schmidt, S., Zadtootaghaj, S., Naderi, B., Griwodz, C. and Möller, S. [2020], 'A latency compensation technique based on game characteristics to mitigate the influence of delay on cloud gaming quality of experience', *Proceedings of the 11th ACM Multimedia Systems Conference* **1**.

[105] Schmidt, S. [2021], *Assessing the quality of experience of cloud gaming services*, Springer.

[106] Schwarz, H., Marpe, D. and Wiegand, T. [2007], 'Overview of the scalable video coding extension of the h.264/avc standard', *IEEE Transactions on Circuits and Systems for Video Technology* **17**(9), 1103–1120.

[107] Seo, J.-D. and Sohn, K.-H. [2011], 'Psnr estimation of h.264/avc bitstream for hierarchical-b picture structure', *Journal of Broadcast Engineering* **16**, 996–1008.

[108] Seshadrinathan, K., Soundararajan, R., Bovik, A. C. and Cormack, L. K. [2010*a*], 'Study of subjective and objective quality assessment of video', *IEEE Transactions on Image Processing* **19**(6), 1427–1441.

[109] Seshadrinathan, K., Soundararajan, R., Bovik, A. C. and Cormack, L. K. [2010*b*], 'A subjective study to evaluate video quality assessment algorithms', *Proceedings of SPIE* .

[110] Seufert, M., Egger, S., Slanina, M., Zinner, T., Hoßfeld, T. and Tran-Gia, P. [2015], 'A survey on quality of experience of http adaptive streaming', *IEEE Communications Surveys and Tutorials* **17**(1), 469–492.

[111] Shafiee Sabet, S., Hashemi, M. R., Shirmohammadi, S. and Ghanbari, M. [2018], A novel objective quality assessment method for perceptually-coded cloud gaming video, *in* '2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)', pp. 75–79.

[112] Shaker, N., Yannakakis, G. N. and Togelius, J. [2011], Feature analysis for modeling game content quality, *in* '2011 IEEE Conference on Computational Intelligence and Games (CIG'11)', pp. 126–133.

[113] Shea R., Jiangchuan Liu, N., C.-H, E. and Cui, Y. [2013], 'Cloud gaming: architecture and performance', *IEEE Network* **27**, 16–21.
URL: *https://www.sfu.ca/ rws1/papers/Cloud-Gaming-Architecture-and-Performance.pdf*

[114] Shea, R., Liu, J., Ngai, E. C.-H. and Cui, Y. [2013], 'Cloud gaming: architecture and performance', *IEEE Network* **27**(4), 16–21.

[115] Shi, L., Zhou, W., Chen, Z. and Zhang, J. [2020], 'No-reference light field image quality assessment based on spatial-angular measurement', *IEEE Transactions on Circuits and Systems for Video Technology* **30**, 4114–4128.

[116] Silva, R. C. C., de Menezes Júnior, J. M. P. and de Araújo Júnior, J. M. [2021], 'Optimization of narx neural models using particle swarm optimization and genetic algorithms applied to identification of photovoltaic systems', *Journal of Solar Energy Engineering* **143**(5), 051001.
URL: *https://doi.org/10.1115/1.4049718*

[117] Sinno, Z. and Bovik, A. C. [2019], 'Large-scale study of perceptual video quality', *IEEE Transactions on Image Processing* **28**(2), 612–627.

[118] Slivar, I., Suznjevic, M., Skorin-Kapov, L. and Matijasevic, M. [2014], Empirical qoe study of in-home streaming of online games, *in* '2014 13th Annual Workshop on Network and Systems Support for Games', pp. 1–6.

[119] Soundararajan, R. and Bovik, A. C. [2012], 'Rred indices: Reduced reference entropic differencing for image quality assessment', *IEEE Transactions on Image Processing* **21**, 517–526.

[120] Soundararajan, R. and Bovik, A. C. [2013], 'Video quality assessment by reduced reference spatio-temporal entropic differencing', *IEEE Transactions on Circuits and Systems for Video Technology* **23**(4), 684–694.

[121] SU, Y.-y. and SANG, Q.-b. [2013], 'Image quality assessment based on phase congruency structural similarity', *Journal of Computer Applications* **32**, 2283–2287.

[122] Sun, K. and Wu, D. [2015], 'Video rate control strategies for cloud gaming', *Journal of Visual Communication and Image Representation* **30**, 234–241.

[123] Suznjevic, M., Slivar, I. and Skorin-Kapov, L. [2016], Analysis and qoe evaluation of cloud gaming service adaptation under different network conditions: The case of nvidia geforce now, *in* '2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)', pp. 1–6.

[124] Sánchez, J. L. G., Vela, F. L. G., Simarro, F. M. and Padilla-Zea, N. [2012], 'Playability: analysing user experience in video games', *Behaviour and Information Technology* **31**, 1033–1054.

[125] Ting, A. and Hedge, A. [2001], 'An ergonomic evaluation of a hybrid keyboard and game controller', *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* **45**, 677–681.

[126] Toni, L., Aparicio-Pardo, R., Pires, K., Simon, G., Blanc, A. and Frossard, P. [2015], 'Optimal selection of adaptive streaming representations', *ACM Transactions on Multimedia Computing, Communications, and Applications* **11**, 1–26.

[127] Topiwala, P., Dai, W., Pian, J., Biondi, K. and Krovvidi, A. [2021], 'Vmaf and variants: towards a unified vqa', *arXiv (Cornell University)* **1**.

[128] Utke, M., Zadtootaghaj, S., Schmidt, S., Bosse, S. and Möller, S. [2020], 'Ndnetgaming - development of a no-reference deep cnn for gaming video quality prediction', *Multimedia Tools and Applications* **81**, 3181–3203.

[129] Valencia, C. H., Vellasco, M. M. and Figueiredo, K. [2023], 'Echo state networks: Novel reservoir selection and hyperparameter optimization model for time series forecasting', *Neurocomputing* **545**, 126317.
URL: *https://www.sciencedirect.com/science/article/pii/S092523122300440X*

[130] Van Damme, S., Torres Vega, M., Heyse, J., De Backere, F. and De Turck, F. [2020], 'A low-complexity psychometric curve-fitting approach for the objective quality assessment of streamed game videos', *Signal Processing: Image Communication* **88**, 115954.

[131] Vu, T. H., Cong, H. P., Sisouvong, T., HoangVan, X., NguyenQuang, S. and DoNgoc, M. [2022], 'Vmaf based quantization parameter prediction model for low resolution video coding', *2022 International Conference on Advanced Technologies for Communications* **1**.

[132] Wahab, A., Ahmad, N., Martini, M. G. and Schormans, J. [2021], 'Subjective quality assessment for cloud gaming', *J* **4**, 404–419.

[133] Wang, R. [1987], *Stochastic processes*, Xi'an Jiaotong University Press.
URL: *https://books.google.co.uk/books?id=jBFInQAACAAJ*

[134] Wang, S. and Dey, S. [2009], Modeling and characterizing user experience in a cloud server based mobile gaming approach, *in* 'GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference', pp. 1–7.

[135] Wang, Y. [2010], 'Analysis application for h.264 video encoding', *1* **1**.

[136] Wang, Y., Wang, H., Shang, J. and Tuo, H. [2019], Resa: A real-time evaluation system for abr, *in* '2019 IEEE International Conference on Multimedia and Expo (ICME)', pp. 1846–1851.

[137] Wang, Z., Bovik, A., Sheikh, H. and Simoncelli, E. [2004], 'Image quality assessment: from error visibility to structural similarity', *IEEE Transactions on Image Processing* **13**(4), 600–612.

[138] Wang, Z. and Simoncelli, E. P. [2005], 'Reduced-reference image quality assessment using a wavelet-domain natural image statistic model', *Human Vision and Electronic Imaging X* .

[139] Winkler, S. [2009], Video quality measurement standards — current status and trends, *in* '2009 7th International Conference on Information, Communications and Signal Processing (ICICS)', pp. 1–5.

[140] Winkler, S. [2012], 'Analysis of public image and video databases for quality assessment', *IEEE Journal of Selected Topics in Signal Processing* **6**(6), 616–625.

[141] Winkler, S. [2013], *Digital Video Quality*, John Wiley and Sons.

[142] Winkler, S. and Mohandas, P. [2008], 'The evolution of video quality measurement: From psnr to hybrid metrics', *IEEE Transactions on Broadcasting* **54**(3), 660–668.

[143] Wong, J.-J. and Liu, J. [2023], 'How the gaming mouse connects the virtual world and the body', *Art and Design Review* **11**, 55–72.

[144] Xiao, C., Zhang, Y., Huang, X., Huang, Q., Chen, J. and Sun, c. [2023], Mastering strategy card game (hearthstone) with improved techniques, *in* '2023 IEEE Conference on Games (CoG)', pp. 1–8.

[145] Xu, D., Lou, S., Shi, Y. and Wu, Z. [1998], *Systems Analysis and Design with MATLAB: Neural Networks*, Xidian University Press.
URL: *https://books.google.co.uk/books?id=9zbsAAAACAAJ*

[146] Xue, J., Zhang, D.-Q., Yu, H. and Chen, C. W. [2014], Assessing quality of experience for adaptive http video streaming, *in* '2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)', pp. 1–6.

[147] Yan, P. and Zhang, C. [2005], *Artificial Neural Networks and Simulated Evolutionary Computation*, Tsinghua University Press.

[148] Yan, T. [2019], 'Bit allocation algorithm based on ssim for 3d video coding', *International Journal of Performability Engineering* **1**.

[149] yo, j. [2023], 'What is cloud gaming and how does it work'.
URL: *https://ukyojana.com/web-stories/what-is-cloud-gaming-and-how-does-it-work/*

[150] Zadtootaghaj, S. [2022], *Quality of experience modeling for cloud gaming services*, Springer.

[151] Zadtootaghaj, S., Barman, N., Rao, R. R. R., Göring, S., Martini, M. G., Raake, A. and Möller, S. [2020], 'Demi: Deep video quality estimation model using perceptual video quality dimensions'.
URL: *https://ieeexplore.ieee.org/document/9287080/*

[152] Zadtootaghaj, S., Barman, N., Schmidt, S., Martini, M. G. and Moller, S. [2018], 'Nr-gvqm: A no reference gaming video quality metric', *2018 IEEE International Symposium on Multimedia (ISM)* **1**.

[153] Zadtootaghaj, S., Schmidt, S., Barman, N., Möller, S. and Martini, M. G. [2018], A classification of video games based on game characteristics linked to video coding complexity, *in* '2018 16th Annual Workshop on Network and Systems Support for Games (NetGames)', pp. 1–6.

[154] Zadtootaghaj, S., Schmidt, S. and Möller, S. [2018], Modeling gaming qoe: Towards the impact of frame rate and bit rate on cloud gaming, *in* '2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)', pp. 1–6.

[155] Zagal, J., Fernández-Vara, C. and Mateas, M. [2008], 'Rounds, levels, and waves the early evolution of gameplay segmentation', *Games and Culture* **3**, 175–198.

[156] Zhang, H., Dong, L., Gao, G., Hu, H., Wen, Y. and Guan, K. [2020], 'Deepqoe: A multimodal learning framework for video quality of experience (qoe) prediction', *IEEE Transactions on Multimedia* **22**(12), 3210–3223.

[157] Zhang, Q., Cheng, L. and Boutaba, R. [2010], 'Cloud computing: state-of-the-art and research challenges', *Journal of Internet Services and Applications* **1**, 7–18.

[158] ZHAO, X., XIE, D. and PAN, K. [2013], 'Color image quality assessment algorithm based on color structural similarity', *Journal of Computer Applications* **33**, 1715–1718.

[159] Zhu, C., Huang, Y., Xie, R. and Song, L. [2021], 'Hevc vmaf-oriented perceptual rate distortion optimization using cnn', *2021 Picture Coding Symposium (PCS)* **1**.

[160] Çalı, M. and Özbek, N. [2020], 'Ssim-based adaptation for dash with svc in mobile networks', *Signal, Image and Video Processing* **1**.

Appendix A

Raw Data Simple

Frame	adm2	adms0	adms1	adms2	adms3	motion2	motion	vif0	vif1	vif2	vif3	vmaf
0	0.964424	0.982432	0.979413	0.975339	0.945086	0	0	0.532862	0.90631	0.953995	0.971514	85.543752
1	0.957975	0.982384	0.977795	0.969053	0.934246	0.19496	0.19496	0.517635	0.884973	0.935865	0.957343	82.568222
2	0.955703	0.981908	0.975976	0.964589	0.93252	0.211753	0.211753	0.504791	0.869623	0.923485	0.947622	80.882464
3	0.956296	0.982133	0.975751	0.966618	0.932495	0.226124	0.226124	0.498943	0.863859	0.919759	0.945382	80.720104
4	0.954583	0.980774	0.97432	0.965403	0.930095	0.164031	0.243228	0.488592	0.857876	0.917589	0.945199	80.217327
5	0.950488	0.980745	0.972853	0.960315	0.92386	0.164031	0.164031	0.476651	0.84293	0.904644	0.934298	78.06052
6	0.948871	0.980589	0.972374	0.958507	0.921085	0.170859	0.170859	0.465481	0.830629	0.894922	0.926791	76.837382
7	0.949331	0.981849	0.972253	0.958182	0.921917	0.28994	0.28994	0.463621	0.824055	0.889462	0.922156	76.533308
8	0.944066	0.978728	0.968754	0.951449	0.915918	0.297438	0.297438	0.442082	0.804741	0.875174	0.911485	74.230915
9	0.950497	0.981392	0.973132	0.959484	0.923037	0.208537	0.438798	0.460006	0.819253	0.885688	0.919915	76.389421
10	0.946222	0.981365	0.971494	0.955812	0.915674	0.208537	0.208537	0.456714	0.811244	0.875669	0.908238	74.277946

Table A.1 continued from previous page

Frame	adm2	adms0	adms1	adms2	adms3	motion2	motion	vif0	vif1	vif2	vif3	vmaf
11	0.950247	0.981097	0.972593	0.962176	0.920602	0.331366	0.331366	0.45435	0.81097	0.877129	0.911071	75.534096
12	0.952585	0.980364	0.972529	0.965172	0.924586	0.358017	0.358017	0.456204	0.812802	0.879412	0.914319	76.370259
13	0.952191	0.979173	0.97109	0.962541	0.926487	0.365291	0.365291	0.437314	0.7981	0.870701	0.910571	75.796864
14	0.953215	0.980151	0.973688	0.965577	0.925456	0.20807	0.432417	0.456114	0.812806	0.880183	0.916344	76.505612
15	0.951843	0.979983	0.971889	0.963342	0.924468	0.20807	0.20807	0.456028	0.811816	0.878069	0.913161	75.904143
16	0.951004	0.979533	0.971288	0.962827	0.923261	0.338349	0.338349	0.452776	0.807461	0.873733	0.909031	75.434462
17	0.953653	0.982859	0.972875	0.964251	0.926096	4.24288	13.156518	0.463456	0.809893	0.871324	0.903994	80.005807
18	0.954275	0.981635	0.969486	0.963708	0.930459	3.010549	4.24288	0.453404	0.80377	0.871469	0.907638	79.045888
19	0.955754	0.98261	0.973322	0.965916	0.93083	3.010549	3.010549	0.460517	0.795188	0.857764	0.893176	77.774136
20	0.954606	0.982491	0.973373	0.965159	0.928632	5.870943	6.118345	0.445037	0.770678	0.840124	0.881884	79.400103
21	0.953718	0.982763	0.972192	0.96297	0.928601	5.870943	5.870943	0.431374	0.750501	0.821902	0.865942	77.525606
22	0.949301	0.982037	0.96876	0.95782	0.923329	7.895338	7.895338	0.451148	0.749295	0.828477	0.878509	79.91455
23	0.945565	0.982874	0.97118	0.950529	0.917457	8.384489	9.365352	0.468996	0.731563	0.80506	0.853221	77.1092
24	0.952107	0.983709	0.974333	0.958549	0.926885	8.161454	8.384489	0.457468	0.724583	0.809532	0.865025	79.267663
25	0.948827	0.984264	0.976657	0.955438	0.919068	7.881208	8.161454	0.47739	0.724216	0.79905	0.848442	76.738213
26	0.951359	0.983556	0.974346	0.960246	0.92312	7.881208	7.881208	0.455716	0.714034	0.800503	0.858548	78.10916
27	0.948239	0.985439	0.973127	0.959091	0.915438	8.019093	8.019093	0.504725	0.741571	0.810258	0.854195	77.435371
28	0.951769	0.983934	0.975028	0.961246	0.923003	8.082476	8.082476	0.482794	0.726956	0.802711	0.853372	78.01371
29	0.951268	0.985442	0.974195	0.965446	0.917534	8.223019	8.223019	0.506494	0.738512	0.80401	0.845028	77.43388
30	0.953899	0.984988	0.975945	0.966472	0.923299	8.594374	8.594374	0.458733	0.711815	0.793893	0.850046	78.573877
31	0.954357	0.984589	0.973783	0.969213	0.923875	8.591462	8.683195	0.513499	0.749066	0.815599	0.857762	79.721202
32	0.95426	0.984809	0.977819	0.965196	0.924596	8.499941	8.591462	0.489103	0.729824	0.803415	0.852794	78.937975

Table A.1 continued from previous page

Frame	adm2	adms0	adms1	adms2	adms3	motion2	motion	vif0	vif1	vif2	vif3	vmaf
33	0.952777	0.983753	0.973408	0.967651	0.92148	8.299361	8.499941	0.517212	0.744081	0.808708	0.849937	78.303449
34	0.954002	0.984563	0.976311	0.968091	0.922244	8.195808	8.299361	0.469507	0.723522	0.803724	0.858078	78.995769
35	0.952963	0.985134	0.979106	0.965625	0.919961	8.195808	8.195808	0.512406	0.741546	0.807197	0.848123	78.064794
36	0.949689	0.985652	0.975017	0.961683	0.916295	8.196445	8.196445	0.484944	0.721076	0.794712	0.843063	76.741621
37	0.954872	0.985366	0.976742	0.970202	0.922472	8.363992	8.363992	0.466396	0.719373	0.801538	0.856392	79.187049
38	0.953226	0.984616	0.975739	0.968109	0.920125	8.637307	8.764254	0.502456	0.736278	0.807151	0.852389	78.928677
39	0.953828	0.984708	0.976145	0.968623	0.921559	8.423096	8.637307	0.465924	0.705638	0.784568	0.838748	77.308539
40	0.948999	0.984395	0.971218	0.964452	0.914404	8.423096	8.423096	0.491815	0.724354	0.797868	0.8439	76.982408
41	0.950223	0.984556	0.972651	0.960415	0.919401	8.113691	8.470571	0.467184	0.716286	0.801903	0.860588	78.28913
42	0.946734	0.984172	0.972921	0.956659	0.913118	7.957616	8.113691	0.49412	0.730594	0.806412	0.85535	77.076156
43	0.950523	0.985637	0.973775	0.959575	0.920462	7.719499	7.957616	0.460775	0.702651	0.787134	0.844181	76.37253
44	0.951123	0.984889	0.978002	0.957514	0.921147	7.719499	7.719499	0.493664	0.733645	0.810942	0.86152	78.305657
45	0.952799	0.985105	0.974687	0.963752	0.923299	7.768516	7.768516	0.453325	0.711571	0.801724	0.863019	78.662744
46	0.956076	0.98462	0.975218	0.968391	0.928531	5.889782	8.052037	0.464577	0.723211	0.81105	0.870359	78.063141
47	0.946677	0.985385	0.972682	0.953455	0.916387	3.560068	5.889782	0.399583	0.663406	0.763116	0.830806	69.835626
48	0.946312	0.982528	0.966814	0.955269	0.919193	1.787566	3.560068	0.389645	0.663901	0.765111	0.832675	68.032798
49	0.948663	0.982829	0.969491	0.955427	0.923418	1.787566	1.787566	0.389772	0.66949	0.772671	0.841166	69.273428
50	0.948971	0.982515	0.970389	0.959674	0.921372	3.220599	3.935761	0.405131	0.693499	0.795155	0.861072	72.793106
51	0.954629	0.982262	0.970603	0.962841	0.933546	2.810525	3.220599	0.392839	0.708673	0.818054	0.8841	75.845046
52	0.950589	0.982615	0.966565	0.95548	0.930526	2.621593	2.810525	0.405852	0.70955	0.811563	0.873294	73.760771
53	0.948329	0.980291	0.962794	0.956161	0.927093	2.621593	2.621593	0.397386	0.698031	0.799466	0.862512	72.235371
54	0.951438	0.980951	0.9635	0.955345	0.934652	2.692677	2.692677	0.395501	0.701371	0.804179	0.867096	73.412786

Table A.1 continued from previous page

Frame	adm2	adms0	adms1	adms2	adms3	motion2	motion	vif0	vif1	vif2	vif3	vmaf
55	0.951959	0.981502	0.96564	0.958686	0.932309	2.723873	2.723873	0.405301	0.714035	0.815738	0.877129	74.545478
56	0.953508	0.980592	0.965255	0.963069	0.933362	2.767574	2.804415	0.381326	0.702702	0.811393	0.878309	75.001165
57	0.952423	0.981229	0.964734	0.964161	0.929702	2.52237	2.767574	0.39711	0.705923	0.807677	0.869864	73.701513
58	0.947748	0.981698	0.964999	0.956918	0.923626	2.52237	2.52237	0.387839	0.692089	0.794263	0.857735	71.576425
59	0.9501	0.982134	0.966113	0.954719	0.930368	2.765926	2.765926	0.39151	0.693587	0.796018	0.860467	72.550073
60	0.949526	0.983511	0.96736	0.95968	0.924132	2.980676	2.980676	0.409902	0.708312	0.807515	0.868771	73.479859
61	0.949471	0.982263	0.964865	0.960517	0.924598	3.144214	3.144214	0.399477	0.709173	0.813994	0.879019	74.631024
62	0.948143	0.983141	0.967651	0.9628	0.918152	3.72293	3.893158	0.421105	0.713775	0.810434	0.868021	74.000021
63	0.94557	0.982217	0.967659	0.960896	0.913397	3.72293	3.72293	0.416943	0.702383	0.799245	0.858481	72.509951
64	0.944591	0.982993	0.968314	0.96204	0.909335	3.813367	3.924005	0.414248	0.698801	0.796896	0.859485	72.435947
65	0.947245	0.983	0.966593	0.96551	0.913817	3.753579	3.813367	0.417533	0.707838	0.806402	0.867798	73.746509
66	0.949361	0.982535	0.966573	0.964627	0.919416	2.899407	3.753579	0.399639	0.714338	0.819409	0.88206	74.707233
67	0.949856	0.983968	0.968007	0.96431	0.919885	2.73619	2.899407	0.415403	0.722129	0.82142	0.879788	74.432907
68	0.950301	0.981794	0.967455	0.966034	0.920674	2.566448	2.73619	0.404216	0.716739	0.818732	0.879927	74.319959
69	0.950722	0.982429	0.967327	0.96379	0.923123	2.566448	2.566448	0.41246	0.727896	0.82794	0.886226	75.086816
70	0.953043	0.982605	0.967572	0.966705	0.926329	2.446867	2.792176	0.387718	0.713272	0.819699	0.883855	75.141214
71	0.951984	0.982921	0.966479	0.964306	0.925814	2.222401	2.446867	0.40728	0.727478	0.82815	0.886783	75.024782
72	0.95252	0.981396	0.96792	0.966204	0.925868	2.222401	2.222401	0.402191	0.723624	0.825626	0.885457	74.990464
73	0.954863	0.983316	0.970536	0.968623	0.92833	2.279338	2.353958	0.404742	0.727508	0.828685	0.886988	75.722585
74	0.953207	0.983404	0.966981	0.967429	0.926297	2.279338	2.279338	0.390616	0.716691	0.82263	0.88515	75.150904
75	0.951719	0.983871	0.970744	0.966549	0.921461	2.676473	3.012559	0.413692	0.725857	0.825174	0.883152	75.104459
76	0.949243	0.983867	0.967206	0.961215	0.920847	2.676473	2.676473	0.408324	0.713227	0.812443	0.871396	73.409409

Table A.1 continued from previous page

Frame	adm2	adms0	adms1	adms2	adms3	motion2	motion	vif0	vif1	vif2	vif3	vmaf
77	0.950788	0.982513	0.966623	0.960563	0.925948	3.024075	3.024075	0.410003	0.716127	0.816031	0.876544	74.581508
78	0.952295	0.982264	0.968376	0.960909	0.928941	3.035003	3.035003	0.416792	0.727807	0.826897	0.885957	75.868414
79	0.954427	0.98276	0.967242	0.965241	0.931616	2.963532	3.048972	0.403513	0.729189	0.833066	0.89331	77.011749
80	0.952861	0.984414	0.968773	0.965436	0.926246	2.486733	2.963532	0.417123	0.731945	0.829557	0.885975	75.437247
81	0.950581	0.984875	0.967198	0.962198	0.923701	2.486733	2.486733	0.40472	0.714047	0.81246	0.870574	73.418825
82	0.949859	0.984356	0.967631	0.961569	0.922413	2.625171	2.681995	0.405466	0.719603	0.819165	0.87847	74.183582
83	0.952959	0.985013	0.972848	0.964811	0.925319	2.625171	2.625171	0.417237	0.737439	0.835921	0.893198	76.333931
84	0.953971	0.983601	0.969863	0.965362	0.929134	2.628248	2.628248	0.400141	0.73331	0.837648	0.897923	77.02853
85	0.953458	0.983762	0.969317	0.963383	0.929504	2.319629	2.678449	0.415399	0.736715	0.835249	0.892749	76.043666
86	0.951118	0.982734	0.965801	0.963788	0.925245	2.319629	2.319629	0.398989	0.714252	0.814089	0.874211	73.674484
87	0.951045	0.980928	0.968185	0.963006	0.925153	2.24725	2.436275	0.39803	0.719256	0.819855	0.879152	74.10245
88	0.952578	0.982577	0.968172	0.962119	0.929169	2.214635	2.24725	0.412874	0.741557	0.839746	0.895179	76.063865
89	0.954825	0.981989	0.967986	0.964431	0.933191	2.178047	2.214635	0.39414	0.738466	0.841993	0.89969	76.974442
90	0.95585	0.98372	0.968816	0.967936	0.932201	1.984589	2.178047	0.404075	0.741864	0.840829	0.896119	76.61205
91	0.950397	0.983384	0.967037	0.96397	0.922573	1.575879	1.984589	0.393287	0.722105	0.822393	0.880518	73.399068
92	0.953199	0.983246	0.965979	0.963911	0.929988	1.575879	1.575879	0.394068	0.73069	0.832498	0.890251	74.974925
93	0.953818	0.983064	0.966575	0.964723	0.930476	2.024162	2.28862	0.403618	0.743649	0.843174	0.898437	76.484689
94	0.954783	0.981088	0.966652	0.966089	0.932541	2.024162	2.024162	0.393355	0.745265	0.847461	0.90347	77.234496
95	0.955153	0.983054	0.967222	0.96639	0.932118	1.799005	2.228549	0.407067	0.751909	0.849903	0.903353	77.056162
96	0.951455	0.98261	0.964826	0.965252	0.924857	1.799005	1.799005	0.398334	0.736797	0.836164	0.892499	75.113225
97	0.952164	0.98196	0.965994	0.962899	0.927871	1.982917	1.982917	0.397479	0.737546	0.837261	0.894697	75.664962
98	0.953009	0.983679	0.968019	0.964973	0.926893	2.080407	2.080407	0.410379	0.749698	0.847356	0.900505	76.631868

Table A.1 continued from previous page

Frame	adm2	adms0	adms1	adms2	adms3	motion2	motion	vif0	vif1	vif2	vif3	vmaf
99	0.954896	0.982734	0.9693	0.965965	0.93066	2.278307	2.278307	0.413188	0.752776	0.851746	0.905664	77.762739
100	0.954732	0.983185	0.971351	0.966402	0.928851	2.910857	2.910857	0.442261	0.758027	0.851859	0.903909	78.215437
101	0.95369	0.982624	0.968907	0.966277	0.927587	3.052805	3.052805	0.444459	0.754467	0.848386	0.901961	77.899953
102	0.956142	0.983037	0.970097	0.969016	0.931394	3.395188	3.395188	0.464203	0.769538	0.858666	0.907878	79.504757
103	0.957068	0.982062	0.968781	0.965872	0.937197	3.733223	3.741925	0.445551	0.769829	0.864187	0.916222	80.952865
104	0.95546	0.983732	0.968374	0.965245	0.933157	3.66765	3.733223	0.459864	0.772354	0.861987	0.909541	79.920847
105	0.955718	0.98431	0.971434	0.964585	0.93263	3.605395	3.66765	0.457267	0.769915	0.860018	0.908985	79.803823
106	0.955563	0.983598	0.972222	0.966053	0.931076	3.605395	3.605395	0.463013	0.775825	0.863692	0.912123	80.107129
107	0.958906	0.983734	0.972288	0.968814	0.937525	3.622407	3.676155	0.452535	0.777735	0.869554	0.917977	81.50068
108	0.956935	0.983807	0.970223	0.966962	0.935125	3.525953	3.622407	0.462098	0.777406	0.864591	0.910622	80.221558
109	0.955902	0.9829	0.971055	0.966404	0.932706	3.496526	3.525953	0.45835	0.776283	0.865025	0.913177	80.191211
110	0.955899	0.983	0.970193	0.965318	0.933982	3.496526	3.496526	0.46267	0.778387	0.865774	0.91286	80.177801
111	0.958797	0.98267	0.973539	0.96812	0.93779	3.41733	3.51653	0.450508	0.780538	0.871346	0.920058	81.462735
112	0.95689	0.984558	0.973423	0.967846	0.93258	3.241902	3.41733	0.454608	0.774359	0.86293	0.910231	79.824244
113	0.955874	0.983755	0.97196	0.964649	0.93327	3.241902	3.241902	0.442726	0.76002	0.852031	0.903563	78.800668
114	0.956945	0.984036	0.970555	0.967248	0.934704	3.056525	3.359099	0.445977	0.769811	0.858619	0.907323	79.285419
115	0.95905	0.982704	0.972204	0.968261	0.938932	2.708916	3.056525	0.433235	0.775761	0.867356	0.915821	80.297167
116	0.958689	0.98299	0.974568	0.970602	0.935288	2.510062	2.708916	0.434405	0.774799	0.863646	0.911008	79.496262
117	0.957249	0.982759	0.97153	0.96672	0.93586	2.504372	2.510062	0.423414	0.766919	0.859185	0.909808	78.996698
118	0.957189	0.983454	0.972859	0.965236	0.935867	2.504372	2.504372	0.419113	0.75789	0.850675	0.902401	78.173467
119	0.960137	0.983325	0.972583	0.970643	0.939545	2.934718	2.934718	0.446081	0.765911	0.857697	0.908459	79.875731
120	0.958046	0.983619	0.973358	0.968648	0.935251	4.703449	4.703449	0.450737	0.725066	0.81542	0.874129	77.594502

Appendix B

NARX Simulation Setting and Results

	Stop	Time	Num Epoch	Best Perf	Mu	Gradient	e-Mean	e-Max	e-Sum	e-Fit
1	Max_mu	1.1872	317	7.7179E-03	5.0000E+10	1.2463E-04	-3.4754E-06	4.0841E-01	3.4499E+00	63.77%
2	Max_mu	0.7893	178	7.4400E-03	5.0000E+10	1.6743E-04	1.1489E-05	3.9689E-01	3.3182E+00	64.43%
3	Max_mu	1.7960	499	2.4696E-03	5.0000E+10	2.2975E-04	2.9390E-06	1.8238E-01	1.0990E+00	79.51%
4	Max_mu	1.3694	319	1.9788E-03	5.0000E+10	2.4699E-04	-5.6526E-06	1.4566E-01	8.7861E-01	81.67%
5	Max_ epochs	2.1383	500	1.6922E-03	5.0000E+00	2.6674E-04	-2.5070E-06	1.3403E-01	7.4965E-01	83.05%
6	Max_ epochs	2.2397	500	3.6917E-03	5.0000E+00	4.5902E-04	-7.4864E-05	2.1986E-01	1.6317E+00	74.98%
7	Max_ epochs	2.3604	500	3.3315E-03	5.0000E+00	4.9047E-04	-8.3266E-06	2.2296E-01	1.4692E+00	76.24%
8	Max_ epochs	2.5662	500	2.8448E-03	5.0000E+00	5.2291E-04	3.1386E-05	1.9716E-01	1.2517E+00	78.06%
9	Max_ epochs	2.7679	500	2.9089E-03	5.0000E+00	5.9558E-04	8.6684E-06	1.7703E-01	1.2770E+00	77.83%
10	Max_ epochs	3.0845	500	1.8900E-03	5.0000E+00	4.8599E-04	5.8188E-06	1.9699E-01	8.2781E-01	82.15%
11	Max_ epochs	3.2903	500	1.2296E-03	5.0000E+00	3.9051E-04	-1.3485E-05	1.5099E-01	5.3732E-01	85.62%

12	Max_ epochs	4.6004	500	9.6489E-04	5.0000E+01	4.1173E-04	4.4926E-04	1.1299E-01	4.2069E-01	87.27%
13	Max_ epochs	5.2347	500	1.3365E-03	5.0000E+00	5.3435E-04	1.2762E-05	1.5709E-01	5.8137E-01	85.04%
14	Max_ epochs	6.1038	500	9.4551E-04	5.0000E+01	4.5870E-04	3.8489E-06	1.1514E-01	4.1035E-01	87.43%
15	Max_ epochs	7.1129	500	7.4859E-04	5.0000E+00	4.4765E-04	1.3870E-04	1.2553E-01	3.2414E-01	88.82%
16	Max_mu	1.1252	13	6.0068E-02	5.0000E+10	1.1368E-03	1.1368E-03	6.2146E-01	2.5949E+01	0.00%
17	Max_ epochs	7.7955	500	3.5452E-04	5.0000E+00	3.1315E-04	2.1781E-06	9.2710E-02	1.5280E-01	92.33%
18	Max_ epochs	8.6166	500	2.5843E-04	5.0000E+00	2.8721E-04	4.7098E-06	8.6263E-02	1.1112E-01	93.45%
19	Max_ epochs	9.5403	500	1.9997E-04	5.0000E+01	2.7899E-04	8.2776E-07	9.9164E-02	8.5787E-02	94.24%
20	Max_ epochs	9.7729	500	9.4872E-06	5.0000E+01	5.8709E-05	4.3734E-05	1.4197E-02	4.0605E-03	98.75%
21	Max_ epochs	10.8111	500	9.4811E-16	5.0000E+00	2.3394E-07	-1.0706E-08	1.6025E-07	4.0484E-13	100.00%
22	Min_gradient	11.0542	474	1.1069E-15	5.0000E+01	9.9195E-08	-6.9990E-10	1.2735E-07	4.7152E-13	100.00%
23	Max_mu	0.7370	16	5.9795E-02	5.0000E+10	1.1901E-03	1.1901E-03	6.1731E-01	2.5413E+01	0.00%
24	Max_mu	0.6484	10	5.9593E-02	5.0000E+10	1.1980E-03	1.1980E-03	6.1640E-01	2.5268E+01	0.00%
25	Min_gradient	8.5258	303	9.8549E-16	5.0000E+01	9.9830E-08	4.9033E-10	1.5526E-07	4.1686E-13	100.00%
26	Max_mu	0.7048	11	5.9483E-02	5.0000E+10	1.2147E-03	1.2147E-03	6.1510E-01	2.5102E+01	0.00%
27	Min_gradient	13.7349	428	8.8313E-15	5.0000E+01	9.7766E-08	-2.9299E-09	4.3318E-07	3.7180E-12	100.00%
28	Max_mu	0.7468	12	5.9450E-02	5.0000E+10	1.2326E-03	1.2326E-03	6.1387E-01	2.4969E+01	0.00%
29	Max_mu	0.8397	13	5.9493E-02	5.0000E+10	1.1981E-03	1.1980E-03	6.1342E-01	2.4928E+01	0.00%
30	Max_mu	0.8711	14	5.9441E-02	5.0000E+10	1.2461E-03	1.2461E-03	6.1269E-01	2.4846E+01	0.00%
31	Max_ epochs	18.5785	500	2.9912E-14	5.0000E+01	1.8441E-07	4.2324E-09	1.4965E-06	1.2473E-11	100.00%
32	Max_ epochs	20.1298	500	3.9886E-16	5.0000E+01	1.9507E-07	1.1453E-08	6.8302E-08	1.6592E-13	100.00%

Appendix C

Simulation Code

Algorithm 2 NARX Network Training and Simulation

function NARX_NET($data, dly$)
  $y \leftarrow$ Transpose($data.outputdata$)
  $u \leftarrow$ Transpose($data.inputdata$)
  $y \leftarrow$ Convert to sequence(y)
  $u \leftarrow$ Convert to sequence(u)
  $p \leftarrow$ Get sequence elements from u starting at index $(dly+1)$ to end
  $t \leftarrow$ Get sequence elements from y starting at index $(dly+1)$ to end
  $d1 \leftarrow$ Set as integers from 1 to dly
  $d2 \leftarrow$ Set as integers from 1 to dly
  $narx\_net \leftarrow$ Initialize NARX network($p, t, d1, d2, [10, 10]$)
  narx_net.trainFcn $\leftarrow' trainbr'$
  narx_net.divideMode $\leftarrow' none'$
  narx_net.trainParam.show $\leftarrow$ 50
  narx_net.trainParam.epochs $\leftarrow$ 500
  $Pi \leftarrow$ Combine sequences from the start to dly of both u and y
  Start Timer
  $(narx\_net, tr) \leftarrow$ Train network($[p;t], t, Pi$)
  $last\_time \leftarrow$ **Stop Timer**
  $netx \leftarrow narx\_net$
  $yp \leftarrow$ Simulate network($narx\_net, [p;t], Pi$)
  $(a, b, c, d) \leftarrow$ Fit ARX model(Convert to matrix(t), Convert to matrix(yp))
  $e \leftarrow$ Calculate error(Convert to matrix(yp) – Convert to matrix(t))
  Plot error(e)
end function

Algorithm 3 Standardization VMAF

```
function VMAFOutPut(file_path)
    Import pandas as pd
    Import ElementTree from xml.etree
    rows ← empty list
    Open file_path for reading as file
    for each line in file do
        if line starts with "<frame " then
            element ← parse line as XML
            Append element.attributes to rows
        end if
    end for
    df ← convert rows to DataFrame
    return df
end function
vmaf_output_path ← "PATH"
df ← VMAFOutPut(vmaf_output_path)
excel_file_path ← "PATH"
Write df to Excel at excel_file_path without index
Print "VMAF data written to excel_file_path"
```