Scholar Publishing Group
International Journal of Big Data Intelligent Technology
https://doi.org/10.38007/IJBDIT.2026.070111
ISSN 2790-0932 Vol. 7, Issue 1: 87-95

Research on End-To-End Reliability Modeling and Optimization of Service Grid

Xiao Ma

Cloud Data Technologies, eBay, San Jose, 95125, California, United States

Keywords: Service grid, Latency, Jitter, Machine learning, Heterogeneous neural networks

Abstract: This study focuses on the challenges of end-to-end reliability modeling and optimization in service grids. In response to the shortcomings of traditional methods in adapting to heterogeneous link characteristics, data collection limitations, and dynamic environment adaptability, an innovative solution based on machine learning is proposed. The research background points out that latency and jitter, as key performance indicators, directly affect service quality, while traditional models have significant deficiencies in capturing nonlinear relationships, data collection costs, and dynamic optimization capabilities. The research method adopts machine learning techniques such as deep neural networks, random forests, and LSTM, combined with high-precision time synchronization (accuracy $\leqslant$ 300 nanoseconds) and large capacity data collection (single link data volume $\geqslant$ 50GB, lasting for 3 months), to construct a flexible and adjustable real dataset; Propose a heterogeneous neural network latency model, which independently models a single link through multiple models and introduces a weight learning mechanism to intelligently integrate the results of each link to adapt to heterogeneous characteristics; Introducing transfer learning to supplement target domain data, reducing annotation dependencies, and expanding jitter modeling application scenarios. Research has found that LSTM performs the best in modeling single link temporal sequences, but the entire path modeling requires heterogeneous neural networks to adapt to the differences in each link; When using transfer learning combined with feedforward neural networks and LSTM to model jitter, the R^2 index exceeded 0.99, verifying the efficient utilization of data and the model's generalization ability; Heterogeneous neural network has better loss and fitting degree than single model in path delay prediction, which can effectively capture complex network behavior characteristics. Research contributions include the construction of high-quality datasets, new sample approximation methods based on path and link relationships, innovation in heterogeneous neural network models, and the application of transfer learning in supplementing lost traffic data. In the future, we will explore potential impact relationships between multiple links, consistency of transfer sample relationships, and better jitter modeling models to continuously optimize network performance and management support.

1. Introduction

The research on end-to-end reliability modeling and optimization of service grids focuses on deeply abstracting network features, behaviors, and performance, gaining a deep understanding of topology structure, and predicting development trends to address the challenges brought about by the expansion of network scale and the surge in service verification demand. Time delay and jitter, as key performance indicators, directly affect user experience and service quality - time delay reflects data transmission delay and determines service response speed; Jitter reflects transmission stability, and its abnormalities may cause problems such as data loss and service interruption. However, traditional modeling methods face multiple challenges: a single model is difficult to adapt to the heterogeneous characteristics of different links, resulting in insufficient capture of nonlinear relationships; Data collection is often limited by device rules, making it difficult to obtain a wide range of operating parameters, and large-scale traffic measurement is costly; Traditional optimization problems are mostly NP hard and lack self-learning ability, making it difficult to adapt to dynamic network environments; Empirical models have shortcomings in processing high-speed and large amounts of data, such as poor convergence and high misjudgment rates. The motivation for this study stems from solving the above problems: using machine learning techniques such as deep neural networks, random forests, and LSTM to construct a real network environment dataset, and enhancing model generalization ability through fine-grained, high-precision time synchronization and large capacity data (over 10000 data points under a single traffic value); Propose a delay model based on heterogeneous neural networks, which independently models a single link through multiple models and introduces a weight learning mechanism to intelligently integrate the results of each link to adapt to characteristic differences; Introducing transfer learning to supplement target domain data, reducing dependence on annotated data, and solving the problem of data scarcity. Clear objective: To construct a structurally flexible and adjustable real dataset to verify the accuracy and feasibility of modeling latency and jitter; Prove the superiority of heterogeneous neural network models in delay prediction through performance comparison; Expand the application scenarios of jitter modeling using transfer learning. Contributions include: proposing a heterogeneous neural network latency model that adapts to the heterogeneous characteristics of links, improving modeling accuracy; Efficiently utilizing data and reducing measurement costs through transfer learning; Building high-value datasets promotes the widespread application of machine learning in network feature modeling.

2. Correlation theory

2.1 Overview and Architecture Analysis of Software Defined Networking

Software defined networks [1] were first proposed by a research team at Stanford University, aiming to address the rigidity and complexity issues caused by the close coupling of control and data forwarding functions in traditional networks. The core innovation lies in separating the control plane from the data forwarding plane, and implementing global dynamic management and programming control of the network through a centralized controller, thereby enhancing the flexibility, programmability, and intelligence level of the network. The SDN architecture mainly consists of three parts: the controller serves as the core component, communicates with network devices through open interfaces (such as OpenFlow), centrally issues flow table rules to guide packet forwarding paths, and supports customizing network behavior through applications or scripts; Data plane devices (such as switches and routers) are only responsible for performing forwarding, filtering, and processing operations, and operate according to controller instructions; The application module is built on top of the controller and implements specific network functions (such

as traffic engineering, security policies, network virtualization, etc.) through programming interfaces. Compared to traditional networks, SDN achieves centralized management by separating control and forwarding, simplifying network configuration complexity and enhancing maintainability; Open interfaces and standardized protocols promote interoperability between devices and applications, providing a foundation for network innovation such as traffic optimization and dynamic adjustment of security policies. However, the centralized control feature also introduces security risks (such as susceptibility to packet injection attacks and potential bottleneck of controllers), and OpenFlow [2] protocol vulnerabilities, controller performance limitations, and architecture flaws (such as single point of failure risk) remain key challenges that need to be optimized. Overall, SDN has driven the development of network flexibility and intelligence through architectural innovation, but continuous improvement is needed in security, performance optimization, and protocol refinement to meet the needs of large-scale network scenarios.

2.2 Machine Learning and Its Application in Network Feature Modeling

Machine learning[3], as a core subfield of artificial intelligence, is committed to enabling computers to autonomously learn and optimize performance through data-driven approaches, achieving tasks such as prediction, classification, and clustering. Its learning paradigms include supervised learning (training models using known input-output pairs), unsupervised learning (mining potential structures under unlabeled data), semi supervised learning (mixing labeled and unlabeled data), transfer learning (cross domain knowledge transfer), deep learning (learning complex feature representations through multi-layer neural networks), and reinforcement learning (optimizing strategies through trial and error). Since the 1950s and 1960s, machine learning has been building intelligent systems by simulating human thinking processes. Through breakthroughs in neural networks, support vector machines, deep learning, and other fields, it has become a key technology in image recognition, natural language processing, medical diagnosis, and other fields due to data explosion and increased computing power. Transfer learning focuses on improving the performance of target tasks by utilizing knowledge from the source domain. By pre training models from the source task[4] (such as deep learning models), features or parameters are transferred to the target domain to solve the problem of data scarcity. Its process includes source task learning (acquiring general knowledge) and knowledge transfer (applying to target tasks), with methods covering model parameter transfer, feature matching, and adversarial adaptation, widely used in natural language understanding and computer vision (as shown in Figure 1).

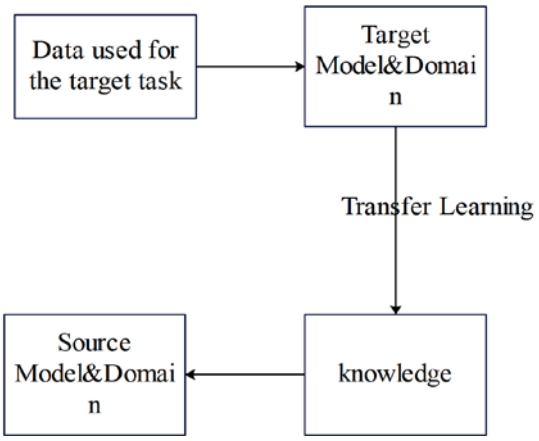

Figure 1 Transfer Learning Knowledge Transfer Flow Chart

LSTM (Long Short Term Memory Network), as a variant of RNN, is adept at handling long sequence time dependency problems. Its structure includes memory cells, input gates, output gates, and forget gates: forget gates

$$ft = \sigma(Wf \cdot [ht - 1, xt] + bf) \quad (1)$$

Control the retention of historical information; The input gate $it = \sigma(Wi \cdot [ht - 1, xt] + bi)$and the unit state update value $\tilde{C}t = \tanh(Wc \cdot [ht - 1, xt] + bC)$ jointly determine the writing of new information; The output gate $ot = \sigma(Wo \cdot [ht - 1, xt] + bo)$ controls the final output $ht = ot \otimes \tanh(Ct)$, where $Ct = ft \otimes Ct - 1 + it \otimes \tilde{C}t$ The use of gating mechanisms to solve gradient vanishing/exploding problems has been widely applied in natural language processing and speech recognition, but there are challenges in overfitting and training efficiency.

CNN (Convolutional Neural Network)[5]processes grid structured data through convolutional layers, pooling layers, and fully connected layers. The convolutional layer utilizes learnable convolution kernels for local feature extraction, and the output feature map size formula is$N = (M - k)/s + 1$is the input size, k is the convolution kernel size, and s is the stride), preserving edge information through padding. The pooling layer (such as max pooling) reduces the feature dimension, enhances translation invariance, and alleviates overfitting by using $ajl = f(bjl + \beta ji \cdot down(ajl - 1, Ml))$. The fully connected layer integrates features to complete classification/regression tasks. CNN training relies on backpropagation and optimization algorithms (such as gradient descent), which are widely used in fields such as image recognition, video analysis, and traffic monitoring. This study focuses on using machine learning methods such as transfer learning, LSTM, and CNN to construct end-to-end feature (such as latency and jitter) models for networks. The performance of the models is verified through real datasets, promoting the development of network feature modeling and optimization.

3. Research method

3.1 Overview of Network Measurement and Feature Modeling Techniques

Network measurement and network feature modeling are the core components of network data analysis. Measurement provides basic data support, while modeling achieves network behavior prediction and performance optimization through data abstraction. Network measurement needs to follow a code of conduct and provide input for applications such as traffic engineering and security analysis. The methods can be divided into three categories: active measurement obtains latency, connectivity, and other information by sending probe packets (such as ICMP Echo requests), but may introduce additional loads that affect measurement accuracy. In practical applications, host reachability needs to be detected first, and latency data needs to be collected by continuously sending data packets; Passive measurement analyzes traffic status by capturing packet headers without increasing network load. It is suitable for micro level statistical analysis of individual packet characteristics or macro level grouping of data streams according to aggregation rules. It is commonly used for jitter estimation (analyzing the timestamp interval of consecutive packets), but there are challenges in data processing in high-speed link scenarios; Other measurement methods are divided into single point and multi-point measurement based on the number of measurement points (multi-point can integrate cross routing information), and end-to-end measurement based on support level (only requiring edge hosts to participate, without router cooperation, inferring internal QoS performance based on packet delay and lost information). Based on router measurement and router collaborative measurement, end-to-end measurement research originated from multicast tree packet loss analysis and later expanded to unicast network performance research. Network feature

modeling relies on high-quality data and requires the selection of appropriate measurement methods. This study is based on software defined network architecture, combined with machine learning models such as transfer learning (solving data scarcity problems through cross domain knowledge transfer), LSTM (processing long sequence time dependencies through gating mechanisms, suitable for time series prediction), and CNN (extracting grid data features through convolutional and pooling layers, and completing classification/regression tasks through fully connected layers) to construct end-to-end feature (such as latency and jitter) models. Subsequently, we will elaborate in detail on latency modeling based on heterogeneous neural networks and transfer learning driven jitter modeling methods to verify model performance and promote technological optimization development.

3.2 Construction Method and Experimental Environment Design of Network Feature Dataset

The modeling of network features relies on high-quality datasets. The dataset constructed in this paper follows the principles of granularity, large data volume, and precise time synchronization. Detailed measurements were taken on 20 links, with each link having raw data of over 50GB. The collection process took 3 months to complete, and after rigorous cleaning and preprocessing, it provided rich input samples for model training, helping the model adapt to diverse real-world environments. The experimental environment is built on a local area network, using two servers, three switches, and fiber optic connections. It is equipped with gigabit network ports and ensures time synchronization accuracy within 300ns. Accurate time synchronization is achieved through a timing server. The data transmission adopts both TCP and UDP protocols to comprehensively evaluate network performance, compare protocol differences, and meet different application requirements (such as TCP for reliable transmission scenarios and UDP for scenarios with high real-time requirements). The data collection process includes: verifying connectivity using the ping command, sending probe streams at a specified rate through a custom packet sending program, capturing packets using Wireshark, and finally processing and analyzing data using Python. In the experiment, 9 randomly selected link data were used to verify the effectiveness of the method. This dataset makes up for the shortcomings of small scale, high cost, and limited privacy of publicly available datasets in the field of networks, providing important support for network feature modeling research.

3.3 Construction process and processing method of network feature dataset

The construction of the network feature dataset is based on the principles of granularity, large data volume, and precise time synchronization, and diversified network data is collected through a local area network experimental environment. The experimental environment is equipped with gigabit Ethernet servers, switches, and fiber optic connections, with time synchronization accuracy controlled within 300ns. TCP and UDP dual protocol transmission is used to comprehensively evaluate network performance differences. The data collection process includes: verifying connectivity using the ping command, sending a probe stream at a specified rate through a custom packet sending program (TCP probe stream sends 500MB of data in a loop, UDP probe stream randomly fills 256B packets), capturing packets using Wireshark [6], and finally processing the data through Python. The data generation steps are divided into four categories: the raw latency data is converted from pcapng to csv to calculate inaccurate latency, and the link latency is corrected by combining the sending/receiving ping packet latency file; The first type of sample is based on the statistical average, maximum, and 95th percentile values of the original delay file; The second type

of sample selects traffic values through parameter n and randomly samples to generate target link training samples; The third type of sample generates source link training samples based on the formula for continuous traffic values; The fourth type of sample generates path delay samples by combining random traffic values from multiple links, ultimately forming a high-quality dataset with 20 links and a single link sample size of over 50GB. It takes 3 months to complete the collection and preprocessing, providing reliable data support for end-to-end delay and transmission jitter modeling in the network.

4. Results and discussion

4.1 Network latency modeling method and experimental analysis based on heterogeneous neural networks

In end-to-end feature modeling of networks, latency is a key factor that requires special attention to the measurement and modeling of path and link latency. Traditional path delay measurement requires a large amount of test data flow, which can easily lead to high network load or even failures; Software defined networks dynamically obtain link latency through controllers, which can effectively reduce testing costs and achieve global optimization. A link refers to the physical/logical connection between nodes, while a path is composed of multiple links forming a data transmission path. Its delay modeling needs to be combined with the characteristics of link state changes. This article proposes a delay modeling method based on heterogeneous neural networks: establishing traffic segmentation mapping relationships through data preprocessing, supplementing uncollected samples through sample migration, constructing heterogeneous models including LSTM, CNN, and feedforward neural networks - each link is independently modeled to adapt to characteristic differences, introducing a weight learning mechanism to intelligently fuse the delay results of each link, and ultimately deriving overall performance through the sum of path delays. The model adopts MSE loss function [7]and Adam optimizer, with 300 iterations and a learning rate of 0.001. The number of traffic segments is set to 1000 to ensure experimental generality. The system model includes a controller (monitoring link delay), a learner (training link model and processing sample transfer), and a path delay detector (obtaining running network path data). Through traffic delay, it constructs data associations between the source domain (experimental network), target domain (running network), and joint domain (path delay). Experimental verification shows that this method improves the accuracy and reliability of path delay modeling while reducing measurement costs, providing effective support for network performance optimization.

4.2 Model experiment

This chapter focuses on solving the problems of high cost of obtaining path delay data and high measurement overhead in network feature modeling. The effectiveness of the method is verified by comparing the performance of heterogeneous neural networks and single neural network models. The experiment uses transfer learning to supplement sample data and constructs a heterogeneous model including LSTM, CNN, and feedforward neural network. Each link is independently modeled to adapt to characteristic differences, and a weight learning mechanism is introduced to fuse the delay results of each link.In the single model test (as shown in Table 1), the LSTM model performs best under random links and background traffic - its average MSE loss is significantly lower than that of feedforward neural networks and CNN, especially the ultra low loss of 0.0000008899 under Link9 (100 background traffic), which benefits from LSTM's ability to capture long-term dependence on time series data. In contrast, the overall average loss of feedforward neural networks is 0.0015489399, and CNN is 0.0016264037, both of which have outlier fitting

bias, reflecting their insufficient adaptability to complex nonlinear time-delay data.

Table 1 Comparison of Performance Test Results of Neural Network Models

Neural Network Model	test link	Background traffic	MSE average loss
Feedforward neural network	Link2	20	0.0003424944
	Link4	40	0.0013989635
	Link5	40	0.0028523205
	Link9	100	0.0002713662
Convolutional Neural Network	Link2	20	0.0003714390
	Link4	40	0.0014346647
	Link5	40	0.0029531518
	Link9	100	0.0002942836
Long Short Term Memory Network	Link2	20	0.0000470408
	Link4	40	0.0001984667
	Link5	40	0.0001483116
	Link9	100	0.0000008899

The heterogeneous neural network model[8]integrates the contributions of each link through a weight learning mechanism. The weight of Link2 is as high as 0.71118353, corresponding to its low loss value (0.0001449720), while the weight of high loss links (such as Link8) is significantly reduced. The total loss of the final model is 0.0012522057, which is better than the loss value of any single model, and the overall fitting trend is better, which verifies the robustness and generalization ability of heterogeneous models in complex network environment. The experimental results show that the model effectively improves the accuracy and stability of path delay modeling by integrating the advantages of multiple models, providing reliable technical support for network performance optimization.

4.3 Effect analysis

Network transmission jitter refers to the instability of packet arrival time at the receiver, caused by factors such as network congestion, routing failures, and changes in link quality, which significantly affects the performance of real-time applications such as voice calls and video conferences. Traditional modeling methods rely on large amounts of real-time data, but are limited by the availability of data in specific environments and cross environment adaptability. Transfer learning improves the generalization ability of the target domain by utilizing source domain knowledge, effectively solving the problem of insufficient samples. Its core lies in combining existing jitter data (source domain) with target network data to construct a traffic delay mapping relationship, thereby enhancing the accuracy and stability of the model in the target environment. In software defined networks, controllers can assist in jitter monitoring, but face dual challenges of sample scarcity and resource consumption: a small number of samples are difficult to support a complete link feature model, and frequent queries lead to excessive controller load. Transfer learning generates source/target link traffic delay samples through traffic segmentation mechanism (evenly dividing link traffic into n segments according to bandwidth, with each segment

corresponding to a reference traffic value), and introduces data saturation index to evaluate dataset quality (based on the number of unmeasured traffic segments and sample distribution density). The experiment used a feedforward neural network and a long short-term memory network (LSTM) to construct a model: the feedforward network processed jitter data through a four layer fully connected structure, and used MMSE loss function[9] and L1 regularization; LSTM utilizes temporal dependency characteristics to capture long-term fluctuation patterns, with a hidden layer set to 64 and batch_2 set to 1 to avoid inter sample interference. The experimental results show that the LSTM model significantly outperforms the feedforward network in MMSE (as low as 0.0000094759) and R^2 (as high as 0.9999945617) metrics, and the fitted curve highly matches real jitter data, verifying its advantages in temporal data processing. However, LSTM is slightly inferior to feedforward networks in handling outliers (abnormal events), which may be sensitive to outliers due to overfitting simple one-dimensional data. The data saturation index and segmentation deviation rate validate the key impact of dataset quality on model performance, and the optimal dataset performs well in average jitter, variance, and distribution uniformity.In summary, transfer learning effectively improves the generalization ability of network transmission jitter modeling by supplementing sample data. Combined with the temporal characteristics of LSTM and rigorous data quality evaluation, it provides reliable technical support for jitter prediction in complex network environments, while revealing the balance between outlier processing and model selection requirements.

5. Conclusion

Software defined networking provides a new solution for managing commonly used resources in society, such as water and electricity. By monitoring critical equipment, faults can be detected and addressed in a timely manner, reducing the likelihood of failure. With the increasingly complex network environment, traditional end-to-end feature modeling methods can no longer meet the requirements of network service quality, and the development of machine learning has brought new possibilities for this. Network features [10], such as latency and transmission jitter, are crucial for measuring network quality and operational status. In response to the difficulties faced by end-to-end feature modeling and the challenge of insufficient data samples, a machine learning based end-to-end feature modeling method has been proposed. Detailed experimental research has been conducted on latency and transmission jitter to optimize modern network performance and provide basic support for network management and optimization. Prior to the experiment, an experimental environment based on running software defined networks and an independent experimental network were established. Data was obtained from both the running and experimental networks, and the quality of the dataset was evaluated using data saturation indicators. In a real environment, data is collected using switches, servers, network timing servers, and data sending/receiving programs. Wireshark is used to capture packets and Python is used to process the data. An experimental model is built in PyTorch environment to obtain latency and network characteristic data. The experiment uses transfer learning to process the dataset and models different neural networks. In the first experiment, modeling a single link using long short-term memory networks, feedforward neural networks, and convolutional neural networks showed that long short-term memory networks performed better in processing time-series data; However, when applied to modeling each link of the entire path, the results are opposite to those of a single link, which verifies the necessity of the delay model based on heterogeneous neural networks. Heterogeneous neural network models each link of the path separately, which can better simulate the complex state of the real network and learn the characteristics of complex behavior. $R^2>0.99$ validates the excellent performance of the modeling method. Innovation focus: 1) Building high-quality datasets (self built real network

collection, time synchronization accuracy ⩽ 300ns, single link data volume exceeding 50GB, lasting for 3 months); 2) Propose a new sample approximation method driven by path link relationships (integrating all link samples under a specific path, combining transfer learning to generate diverse path samples, and approximating the true distribution); 3) Adopting link level heterogeneous neural network modeling to adapt to complex network behaviors; 4) A modeling scheme for feature missing scenarios under continuous background traffic is proposed, which uses transfer learning to complete traffic data (constructing a dataset based on the relationship between latency and transmission jitter, and modeling transmission jitter using LSTM). In the future, we will explore the undiscovered impact relationships of multiple links in path modeling to improve performance, study the potential consistency of the relationship between transfer samples and source samples, and try different neural network models to find better representations of network transmission jitter.

References

[1] *PRC-5333 VHF/UHF Software Defined Networking Radio (SDNR).C4ISR & Mission Systems: Land, 2025, 000(000).*

[2] *Riggs H, Khalid A, Sarwat A I .An Overview of SDN Issues—A Case Study and Performance Evaluation of a Secure Open Flow Protocol Implementation. Electronics (2079-9292), 2025, 14(16).*

[3] *Sterling T, Anderson M, Brodowicz M .Machine Learning. High Performance Computing, 2025:383-393.DOI:10.1016/b978-0-12-823035-0.00019-5.*

[4] *Wang J, Xu L, Sun C .Learning general multi-agent decision model through multi-task pre-training. Neurocomputing, 2025, 627(000).*

[5] *Tripathi S K, Singh S P, Sharma D, et al. Weed Detection using Convolutional Neural Network. 2025.*

[6] *Hui, X. (2026). Research on the Design and Optimization of Automated Data Collection and Visual Dashboard in the Medical Industry. Journal of Computer, Signal, and System Research, 3(1), 27-34.*

[7] *Shen, D. (2026). Application of Large Language Model in Mental Health Clinical Decision Support System. International Journal of Engineering Advances, 3(1), 23-30.*

[8] *Wang, Y. (2026). Research on Optimization of Neuromuscular Rehabilitation Program Based on Physiological Assessment. European Journal of AI, Computing & Informatics, 2(1), 21-30.*

[9] *Ding, J. (2026). Optimization Strategies for Supply Chain Management and Quality Control in the Automotive Manufacturing Industry. Strategic Management Insights, 3(1), 17-23.*

[10] *Zhang, Q. (2026). How to Improve Marketing Efficiency and Precision through AI-Driven Innovative Products. Strategic Management Insights, 3(1), 1-8.*