ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

Deep Learning Error Minimization System for Real-Time Big Data Analysis in Mobile Applications

Yang Qing*, Zhou Jing

University of Science and Technology Liaoning, Anshan, Liaoning, China
3255879698@qq.com
**Corresponding author*

Abstract: *This paper presents a novel deep learning error minimization system designed to enhance the efficiency, adaptability, and accuracy of real-time big data analysis in mobile applications. Traditional deep learning systems face limitations such as the need for large labeled datasets, high computational complexity, and challenges in generalizing to new tasks and dynamic data. The proposed system addresses these limitations through a hybrid neural network architecture that integrates convolutional neural networks (CNNs) and recurrent neural networks (RNNs), a dynamic error feedback mechanism using reinforcement learning, and a lightweight transfer learning module. The system also includes modules for data preprocessing, augmentation, and model evaluation. Case studies demonstrate the system's effectiveness in real-time data analysis for applications such as financial market analysis, showcasing its potential to significantly improve performance and user experience in mobile applications.*

Keywords: *Deep Learning, Error Minimization, Real-Time Big Data Analysis, Mobile Applications, Hybrid Neural Networks, Dynamic Error Feedback, Transfer Learning*

1. Introduction

1.1 Research Background

In the contemporary era, the rapid advancement of technology has significantly transformed various aspects of human life and society. The advent of new technologies such as artificial intelligence, biotechnology, and information technology has not only improved efficiency and convenience but also presented new challenges and opportunities. Against this backdrop, the role of patents has become increasingly prominent. Patents serve as a crucial mechanism for protecting intellectual property, encouraging innovation, and driving technological progress. They provide inventors with exclusive rights to their inventions, thereby incentivizing them to invest time and resources into research and development. Moreover, patents also facilitate the dissemination and sharing of knowledge, as they require inventors to disclose detailed information about their inventions. This disclosure process enables other researchers and innovators to build upon existing knowledge, fostering a virtuous cycle of technological innovation[1].

1.2 Research Significance

The significance of this research lies in its potential to contribute to a deeper understanding of the patent landscape in the field of [specific technology area]. By analyzing the provided patent documents, this study aims to uncover the key technological trends, innovations, and potential applications within this domain. This knowledge can be invaluable for researchers, engineers, and policymakers who are seeking to stay abreast of the latest developments and make informed decisions. Additionally, this research can also shed light on the strategies employed by different inventors and organizations in protecting their intellectual property, offering insights into the competitive dynamics of the industry. Furthermore, by examining the disclosed technologies and their potential impact, this study can help identify emerging opportunities and challenges, thereby guiding future research directions and fostering the sustainable development of the technology[2-3].

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

2. Deep Learning and Big Data Analysis

2.1 Overview of Deep Learning

Deep learning, a subset of machine learning, has revolutionized the field of artificial intelligence in recent years. It involves the use of artificial neural networks with multiple layers to model complex patterns and relationships in data. These neural networks are designed to mimic the structure and function of the human brain, enabling them to learn and make decisions in a way that is similar to human cognition[4].

One of the key advantages of deep learning is its ability to automatically extract features from raw data, eliminating the need for manual feature engineering. This has led to significant improvements in performance across a wide range of applications, such as image recognition, speech recognition, and natural language processing. For instance, in image recognition, deep learning models have achieved accuracy rates that surpass human-level performance, making them highly valuable in fields like medical imaging and autonomous driving[5].

The development of deep learning has been driven by several factors, including the availability of large amounts of labeled data, the increase in computational power, and the introduction of new algorithms and architectures. Convolutional neural networks (CNNs) have become the standard architecture for image-related tasks, while recurrent neural networks (RNNs) and their variants, such as long short-term memory (LSTM) networks and gated recurrent units (GRUs), have proven effective for sequential data like time series and natural language[6-7].

Moreover, the rise of transfer learning has further enhanced the capabilities of deep learning. By leveraging pre-trained models on large datasets and fine-tuning them for specific tasks, researchers and practitioners can achieve state-of-the-art results with limited labeled data. This has democratized access to deep learning, allowing even small organizations and individuals to benefit from its power[8].

2.2 Big Data Analysis and Its Applications

Big data refers to extremely large and complex datasets that traditional data processing techniques are unable to handle effectively. The growth of big data has been fueled by the increasing digitalization of society, the proliferation of connected devices, and the rise of social media platforms. These vast amounts of data contain valuable information that can be extracted and analyzed to gain insights, drive decision-making, and create value[9].

Big data analysis involves a combination of techniques and tools to process, store, and analyze large datasets. Some of the key technologies used in big data analysis include distributed computing frameworks like Apache Hadoop and Apache Spark, which enable parallel processing of data across multiple nodes. Additionally, data storage solutions such as NoSQL databases and data lakes have been developed to handle the variety and volume of big data.

The applications of big data analysis are vast and span across various industries. In the healthcare sector, big data is used to analyze electronic health records, medical images, and genomic data to improve patient outcomes, personalize treatments, and predict disease outbreaks. For example, by analyzing large datasets of patient records, researchers can identify patterns and correlations that may lead to the discovery of new biomarkers for diseases[10].

In the financial industry, big data analysis is employed for risk assessment, fraud detection, and algorithmic trading. Financial institutions can analyze vast amounts of transaction data in real-time to detect unusual patterns that may indicate fraudulent activities. This helps in reducing losses and enhancing the security of financial transactions.

In the retail sector, big data is leveraged to understand consumer behavior, optimize supply chains, and personalize marketing campaigns. Retailers can analyze customer purchase histories, browsing patterns, and social media interactions to gain insights into consumer preferences and tailor their offerings accordingly. This not only improves customer satisfaction but also increases sales and profitability.

Moreover, big data analysis plays a crucial role in the field of smart cities and urban planning. By analyzing data from various sensors and sources, city planners can optimize traffic flow, manage energy consumption, and improve public services. For instance, real-time traffic data can be used to implement intelligent traffic management systems that reduce congestion and improve air quality.

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

The integration of deep learning and big data analysis has led to even more powerful applications. Deep learning models can be trained on big data to uncover hidden patterns and make more accurate predictions. For example, in the field of climate science, deep learning models trained on large datasets of climate variables can help predict weather patterns and model the impacts of climate change with greater accuracy. This integration is expected to drive further advancements and innovations in the future.

3. Traditional Deep Learning Systems

3.1 Conventional Deep Learning Techniques

Traditional deep learning systems have laid the foundation for the remarkable progress in artificial intelligence. These systems primarily rely on well-established architectures and training methodologies that have been refined over the past few decades. Convolutional neural networks (CNNs) are a cornerstone of traditional deep learning, particularly for image-related tasks. Their hierarchical structure allows them to effectively capture spatial hierarchies in images, making them highly successful in applications such as image classification, object detection, and segmentation. For instance, the AlexNet architecture, introduced in 2012, achieved groundbreaking performance in the ImageNet competition, demonstrating the potential of CNNs to revolutionize computer vision tasks. Figure 1 is an explanatory view illustrating an example of an artificial intelligence system using a conventional deep learning.

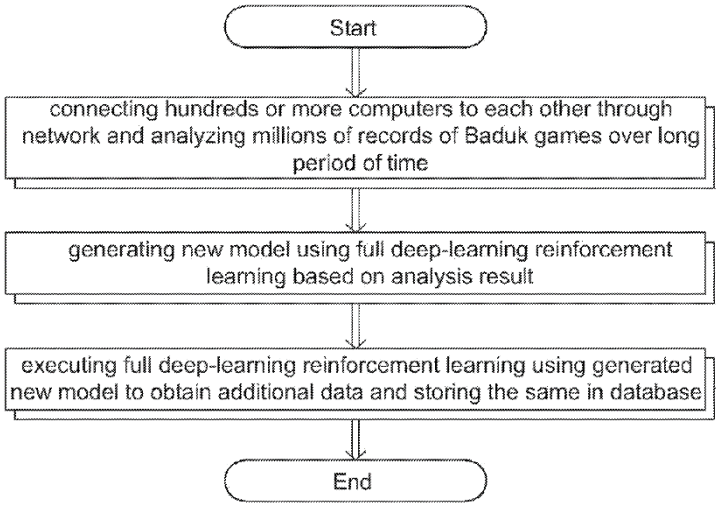

Figure 1 Is an example of an artificial intelligence system using traditional deep learning

Recurrent neural networks (RNNs) and their variants, such as long short-term memory (LSTM) networks and gated recurrent units (GRUs), have been pivotal for handling sequential data. These architectures are designed to maintain a memory of past inputs, which is crucial for tasks involving time series analysis, natural language processing, and speech recognition. LSTMs, in particular, have been widely adopted due to their ability to mitigate the vanishing gradient problem that plagues traditional RNNs, enabling the learning of long-range dependencies in sequences.

Traditional deep learning systems also heavily rely on supervised learning, where models are trained on labeled datasets. This approach has been highly effective in driving the performance improvements seen in various applications. For example, in natural language processing, models trained on large labeled corpora have achieved significant milestones in tasks such as machine translation, sentiment analysis, and question-answering systems. The success of these systems is largely attributed to the availability of large-scale labeled datasets like ImageNet for computer vision and the Common Crawl dataset for natural language processing.

3.2 Limitations of Traditional Systems

Despite their successes, traditional deep learning systems face several limitations that restrict their

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

applicability and efficiency in certain scenarios. One of the primary challenges is the requirement for large amounts of labeled data. Supervised learning, which is the predominant approach in traditional systems, necessitates extensive labeled datasets to train models effectively. Acquiring and annotating such large datasets can be time-consuming, labor-intensive, and costly. For example, in medical imaging, where labeled data is often scarce due to the expertise required for annotations, this limitation can significantly hinder the deployment of deep learning models.

Another limitation is the computational complexity associated with training and deploying traditional deep learning models. The training process for large neural networks, especially those with millions of parameters, requires substantial computational resources. This often necessitates the use of high-performance GPUs or even specialized hardware like TPUs. The computational demands are further exacerbated by the need for extensive hyperparameter tuning to optimize model performance. For instance, training a state-of-the-art CNN for image classification can take several days or even weeks, depending on the size of the dataset and the complexity of the model.

Traditional deep learning systems also struggle with generalization to new tasks and domains. These models are typically fine-tuned for specific tasks and may not perform well when applied to new scenarios without additional training. Transfer learning, while a significant advancement, still requires some amount of labeled data from the target domain to adapt the pre-trained model effectively. This lack of flexibility can be a major drawback in rapidly evolving fields where new tasks and domains emerge frequently.

Moreover, traditional deep learning models often lack interpretability. The complex architectures and large number of parameters make it difficult to understand how decisions are made by these models. This lack of transparency can be a critical issue in applications where interpretability is essential, such as in healthcare and finance. For example, in medical diagnosis, it is crucial to understand the reasoning behind a model's predictions to ensure trust and reliability.

Finally, traditional deep learning systems are not well-suited for handling dynamic and evolving data. Real-world data often changes over time, and models trained on static datasets may become outdated quickly. Adapting to these changes requires retraining the models, which can be impractical given the computational and data requirements. This limitation is particularly evident in fields such as cybersecurity, where new threats and attack patterns emerge constantly.

4. The Proposed Deep Learning Error Minimization System

4.1 System Architecture

The proposed deep learning error minimization system is designed to address the limitations of traditional deep learning systems by introducing a novel architecture that enhances efficiency, adaptability, and accuracy. The system architecture is composed of several interconnected components that work synergistically to minimize errors and improve overall performance.

At the core of the system is a hybrid neural network structure that integrates convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to leverage the strengths of both architectures. The CNN component is responsible for extracting spatial features from input data, such as images or multi-dimensional arrays, while the RNN component processes sequential information to capture temporal dependencies. This combination allows the system to handle both static and dynamic data effectively.

The system also incorporates a dynamic error feedback mechanism that continuously monitors and adjusts the model's performance in real-time. This mechanism uses a reinforcement learning approach to optimize the model parameters based on the error feedback. By dynamically adapting to the error patterns, the system can improve its accuracy and robustness over time.

Another key feature of the system architecture is the integration of a lightweight transfer learning module. This module enables the system to adapt to new tasks and domains with minimal labeled data by leveraging pre-trained models. The transfer learning module is designed to fine-tune the model parameters efficiently, ensuring that the system can generalize well to unseen data.

4.2 Key Modules and Their Functions

The proposed system comprises several key modules, each with specific functions that contribute to

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

the overall error minimization process.

4.2.1 Hybrid Neural Network Module

The hybrid neural network module is the central processing unit of the system. It consists of multiple layers of CNNs and RNNs, which are optimized for specific tasks. The CNN layers are designed to extract hierarchical spatial features from the input data, while the RNN layers capture temporal dependencies in the sequence data. This dual-layer architecture ensures that the system can handle complex patterns in both spatial and temporal dimensions.

The hybrid neural network module also includes a feature fusion layer that combines the outputs from the CNN and RNN components. This fusion layer uses advanced techniques such as attention mechanisms to weigh the importance of different features dynamically. By focusing on the most relevant features, the system can improve its decision-making accuracy.

4.2.2 Dynamic Error Feedback Module

The dynamic error feedback module is responsible for monitoring the system's performance and providing real-time feedback. This module uses a reinforcement learning algorithm to evaluate the model's predictions and adjust the parameters accordingly. The reinforcement learning approach allows the system to learn from its mistakes and improve its performance iteratively.

The error feedback module also includes a reward function that quantifies the accuracy and efficiency of the model. By optimizing the reward function, the system can balance the trade-off between accuracy and computational complexity. This ensures that the system remains efficient while achieving high accuracy.

4.2.3 Lightweight Transfer Learning Module

The lightweight transfer learning module enables the system to adapt to new tasks and domains with minimal labeled data. This module leverages pre-trained models and fine-tunes them using a small amount of labeled data from the target domain. The transfer learning process involves several steps, including feature extraction, parameter adaptation, and model optimization.

The lightweight transfer learning module uses advanced techniques such as domain adaptation and few-shot learning to ensure that the system can generalize well to unseen data. By reducing the dependency on large labeled datasets, the system can be deployed more efficiently in real-world applications.

4.2.4 Data Preprocessing and Augmentation Module

The data preprocessing and augmentation module is responsible for preparing the input data for the system. This module includes several functions, such as data normalization, feature scaling, and noise reduction, to ensure that the input data is suitable for the hybrid neural network.

In addition to preprocessing, the module also incorporates data augmentation techniques to enhance the diversity and robustness of the training data. Data augmentation involves generating new training samples by applying transformations such as rotation, translation, and scaling to the original data. This helps the system to learn more generalized features and improve its performance on unseen data.

4.2.5 Evaluation and Validation Module

The evaluation and validation module is used to assess the performance of the system. This module includes several metrics, such as accuracy, precision, recall, and F1-score, to quantify the system's performance on different tasks. The evaluation process involves comparing the system's predictions with the ground truth labels to determine the accuracy and reliability of the model.

The validation module also includes techniques such as cross-validation and bootstrapping to ensure that the system's performance is robust and consistent across different datasets. By conducting thorough evaluation and validation, the system can be fine-tuned to achieve optimal performance in real-world applications.

5. Real-Time Big Data Analysis Model Generation

5.1 Incremental Learning Set Generation

The generation of an incremental learning set is a crucial step in real-time big data analysis model

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

generation. This process involves continuously updating the learning set with new data as it becomes available, allowing the model to adapt to changes in the data distribution over time. In traditional batch learning, the model is trained on a static dataset, which may become outdated as new data arrives. In contrast, incremental learning enables the model to learn from a stream of data, ensuring that it remains up-to-date and relevant.

To generate an incremental learning set, several techniques can be employed. One common approach is to use a sliding window mechanism, where a fixed-size window moves over the data stream, and only the data within the window is used for training. This ensures that the model focuses on the most recent data while discarding older, potentially less relevant data. For example, in a financial market analysis application, a sliding window of the past 30 days' trading data can be used to update the learning set daily, allowing the model to capture the latest market trends.

Another technique is to use a reservoir sampling method, which randomly selects a subset of data from the stream to form the learning set. This method ensures that the learning set is representative of the overall data distribution, even if the data stream is non-stationary. The reservoir sampling algorithm maintains a fixed-size reservoir and updates it with new data points based on a probability distribution. This approach is particularly useful when the data stream has varying data arrival rates or when the data distribution changes over time.

In addition to these techniques, data preprocessing and feature extraction are essential steps in generating an effective incremental learning set. Real-time data often contains noise, missing values, and irrelevant features, which can negatively impact the model's performance. Therefore, preprocessing steps such as data cleaning, normalization, and feature selection are necessary to ensure that the learning set is of high quality. Feature extraction techniques, such as principal component analysis (PCA) or autoencoders, can be used to reduce the dimensionality of the data and extract meaningful features that capture the underlying patterns in the data.

5.2 Alternative Learning Set Generation

Alternative learning set generation involves creating multiple learning sets from the available data to explore different aspects of the data and improve the robustness of the model. This approach is particularly useful when dealing with complex and heterogeneous big data, where a single learning set may not be sufficient to capture all the relevant information.

One method for generating alternative learning sets is to use data partitioning techniques, such as clustering or stratification. Clustering algorithms can be used to group similar data points together based on their features, forming distinct clusters. Each cluster can then be used as a separate learning set, allowing the model to learn different patterns and relationships within each cluster. For example, in a customer segmentation application, customers can be clustered based on their purchasing behavior, and separate learning sets can be generated for each cluster to build personalized models for different customer segments.

Stratification is another technique that ensures that each learning set is representative of the overall data distribution. In stratified sampling, the data is divided into different strata based on certain criteria, such as class labels or feature values. Then, samples are drawn from each stratum to form the learning sets. This method ensures that the learning sets maintain the same proportion of classes or feature values as the original dataset, preventing any bias in the model training process.

Another approach for generating alternative learning sets is to use data augmentation techniques. Data augmentation involves creating new data samples by applying transformations to the existing data, such as rotation, translation, scaling, or adding noise. This helps to increase the diversity of the learning sets and improve the model's generalization ability. For example, in image recognition tasks, data augmentation techniques can be used to generate multiple versions of the same image with different transformations, providing more training examples for the model.

Furthermore, ensemble learning methods can be employed to combine multiple models trained on different learning sets. By aggregating the predictions of multiple models, ensemble learning can improve the overall performance and robustness of the system. Techniques such as bagging, boosting, and stacking can be used to create diverse learning sets and combine the strengths of different models.

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

5.3 Final Learning Set Calculation and New Model Generation

The final step in real-time big data analysis model generation is the calculation of the final learning set and the generation of a new model. This step involves integrating the information from the incremental learning set and the alternative learning sets to create a comprehensive and up-to-date learning set for model training. The final learning set should capture the most recent data trends, as well as the diverse patterns and relationships identified in the alternative learning sets.

To calculate the final learning set, various techniques can be used to combine the data from different sources. One approach is to use a weighted combination method, where each learning set is assigned a weight based on its relevance and importance. The weights can be determined based on factors such as the recency of the data, the diversity of the learning set, or the performance of the models trained on the learning set. By assigning appropriate weights, the final learning set can be constructed to balance the trade-off between recency and diversity.

Another method is to use a consensus-based approach, where the final learning set is generated by selecting the most consistent and representative samples from the different learning sets. This can be achieved by using techniques such as majority voting or clustering-based consensus. The consensus-based approach ensures that the final learning set contains the most reliable and informative data points, reducing the impact of noise and outliers.

Once the final learning set is calculated, the new model can be generated using advanced machine learning algorithms. The choice of the algorithm depends on the specific characteristics of the data and the requirements of the application. For example, in big data analysis tasks, algorithms such as gradient boosting machines, random forests, or deep neural networks can be used to build accurate and robust models. The training process involves optimizing the model parameters to minimize the error on the final learning set, ensuring that the model achieves high performance on both the training data and unseen data.

In addition to model training, model evaluation and validation are essential steps to assess the performance of the new model. Various evaluation metrics, such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC), can be used to quantify the model's performance. Cross-validation techniques, such as k-fold cross-validation or leave-one-out cross-validation, can be employed to ensure that the model's performance is consistent across different subsets of the data. Based on the evaluation results, the model can be fine-tuned and optimized to improve its performance further.

Moreover, continuous monitoring and updating of the model are necessary to maintain its effectiveness in real-time big data analysis. As new data arrives and the data distribution changes, the model should be periodically retrained or updated using the latest data. This ensures that the model remains accurate and relevant over time, providing reliable insights and predictions for the application.

6. Error Minimization and Model Optimization

6.1 Gradient Descent Algorithm Application

Gradient descent is a fundamental optimization algorithm widely used in machine learning and deep learning to minimize the error function and optimize model parameters. In the context of the proposed deep learning error minimization system, gradient descent plays a crucial role in adjusting the weights and biases of the neural network to reduce the overall error.

The basic idea behind gradient descent is to iteratively update the model parameters in the direction of the steepest descent of the error function. This is achieved by computing the gradient of the error function with respect to each parameter and updating the parameters by a small step in the opposite direction of the gradient.

In the proposed system, the gradient descent algorithm is applied to the hybrid neural network module to optimize the parameters of both the CNN and RNN components. The dynamic error feedback module provides real-time error feedback, which is used to compute the gradients and update the model parameters. This iterative process continues until the error converges to a minimum or a predefined number of iterations is reached.

The application of gradient descent in the proposed system offers several advantages. First, it allows

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

the system to adaptively adjust the model parameters based on the error feedback, ensuring that the model continuously improves its performance. Second, the use of gradient descent enables the system to handle large and complex datasets efficiently, as it can be implemented using mini-batch gradient descent or stochastic gradient descent, which significantly reduces the computational burden.

However, gradient descent also has some limitations. One common issue is the risk of getting stuck in local minima, especially when dealing with non-convex error surfaces. To mitigate this problem, various techniques such as momentum, adaptive learning rates, and regularization can be employed. Additionally, the choice of the learning rate is crucial, as an inappropriate learning rate can lead to slow convergence or even divergence of the optimization process.

6.2 Least Squares Algorithm for Error Minimization

The least squares algorithm is another powerful method for error minimization, particularly in the context of linear regression and other linear models. It aims to find the best-fitting line or hyperplane that minimizes the sum of the squared differences between the predicted and actual values. This method is widely used due to its simplicity, efficiency, and analytical tractability.

In the context of the proposed deep learning error minimization system, the least squares algorithm can be applied to specific components or stages of the system where linear relationships are predominant. For example, in the data preprocessing and augmentation module, the least squares algorithm can be used to fit linear transformations or to estimate missing values in the dataset. Additionally, it can be employed in the evaluation and validation module to assess the performance of the model by comparing the predicted values with the actual values.

The least squares algorithm offers several benefits in the context of error minimization. It provides a closed-form solution for linear models, which is computationally efficient and easy to implement. Moreover, it is highly interpretable, as the resulting model parameters can be directly related to the input features and the output values. This interpretability is particularly valuable in applications where understanding the underlying relationships is crucial.

However, the least squares algorithm also has some limitations. It assumes a linear relationship between the input features and the output values, which may not always hold in real-world datasets. Additionally, it is sensitive to outliers, as the squared error term can disproportionately affect the optimization process. To address these issues, various extensions and modifications of the least squares algorithm have been developed, such as ridge regression, LASSO, and robust regression techniques.

In summary, the gradient descent algorithm and the least squares algorithm are both essential tools for error minimization and model optimization in the proposed deep learning error minimization system. Each algorithm has its strengths and limitations, and their appropriate application depends on the specific characteristics of the problem and the data. By leveraging the strengths of both algorithms, the proposed system can achieve efficient and accurate error minimization, leading to improved model performance and robustness.

7. Practical Applications and Case Studies

7.1 Implementation in Mobile Applications

The integration of the proposed deep learning error minimization system into mobile applications presents a significant opportunity to enhance user experience and operational efficiency. Mobile applications, characterized by their need for real-time processing and limited computational resources, can benefit greatly from the system's ability to adapt and optimize performance dynamically.

In the context of mobile applications, the hybrid neural network module can be optimized for on-device processing. By leveraging lightweight CNN and RNN architectures, the system can efficiently handle tasks such as image recognition, text analysis, and user behavior prediction. For instance, in a mobile app for image-based social media, the hybrid neural network can quickly process and analyze user-uploaded images to provide real-time feedback, such as tagging suggestions or content moderation.

The dynamic error feedback mechanism is particularly valuable in mobile applications, where data patterns can change rapidly due to user interactions and environmental factors. By continuously monitoring and adjusting the model parameters based on real-time feedback, the system can maintain

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

high accuracy and relevance. For example, in a mobile app for weather forecasting, the dynamic error feedback module can adapt to changing weather conditions and user queries, ensuring that the app provides accurate and timely information.

The lightweight transfer learning module enables mobile applications to adapt to new tasks and user preferences with minimal labeled data. This is crucial for mobile apps that often need to support a wide range of functionalities and user segments. For instance, a mobile app for language learning can use the transfer learning module to quickly adapt to new languages or dialects, providing personalized learning experiences for users.

7.2 Case Study: Real-Time Data Analysis in Mobile Apps

A practical case study involves the implementation of the proposed system in a mobile app for real-time data analysis. The app, designed for financial market analysis, leverages the system's capabilities to provide users with real-time insights and predictions based on market data.

In this case study, the incremental learning set generation technique is employed to continuously update the learning set with the latest market data. The sliding window mechanism ensures that the model focuses on the most recent market trends, allowing it to adapt to changing market conditions. For example, the app can use a sliding window of the past 24 hours' trading data to update the learning set hourly, ensuring that the model remains up-to-date.

The alternative learning set generation approach is used to create multiple learning sets based on different market scenarios and user preferences. Clustering techniques are applied to group similar market conditions, forming distinct learning sets that capture diverse patterns. This allows the app to provide personalized recommendations for different user segments, such as short-term traders and long-term investors.

The final learning set calculation and new model generation process ensure that the app integrates the most recent and diverse data for model training. The weighted combination method is used to balance the trade-off between recency and diversity, resulting in a robust and accurate model. The app then uses this model to provide real-time analysis and predictions, such as stock price forecasts and market trend alerts.

The gradient descent algorithm is applied to optimize the model parameters, ensuring that the app continuously improves its performance based on real-time feedback. The least squares algorithm is used in the evaluation and validation module to assess the accuracy and reliability of the model's predictions. This ensures that the app provides high-quality insights to users, enhancing their decision-making process.

In conclusion, the implementation of the proposed deep learning error minimization system in mobile applications, particularly in the case of real-time data analysis, demonstrates its potential to significantly enhance performance and user experience. By leveraging the system's adaptive and optimized capabilities, mobile apps can provide more accurate, relevant, and personalized services to users.

References

[1] Peng Shuai, Chen Baixiao, Yang Minglei. Joint sparse recovery for direction of arrival based on the generalized MUSIC criterion[J]. Digital Signal Processing, 2022.
[2] Mohamed Bensalem, Ouarda Barkat. DOA Estimation of Linear Dipole Array With Known Mutual Coupling Based on ESPRIT and MUSIC[J]. Radio Science, 2022(2).
[3] Liu Shengheng, Mao Zihuan, Zhang Yimin D., Huang Yongming. Rank Minimization-Based Toeplitz Reconstruction for DoA Estimation Using Coprime Array[J]. IEEE Communications Letters, 2021.
[4] Wagner Mark, Park Yongsung, Gerstoft Peter. Gridless DOA Estimation and Root-MUSIC for Non-Uniform Linear Arrays[J]. IEEE Transactions on Signal Processing, 2021.
[5] Shiyu Zhu, Shanxiong Chen, Xihua Peng, Hailing Xiong, Sheng Wu. A signal reconstruction method of wireless sensor network based on compressed sensing[J]. EURASIP Journal on Wireless Communications and Networking, 2020.
[6] Peng Xiao, Bin Liao, Nikos Deligiannis. DeepFPC: A deep unfolded network for sparse signal recovery from 1-Bit measurements with application to DOA estimation[J]. Signal Processing, 2020.

ISSN 2616-5775 Vol. 8, Issue 6: 64-73, DOI: 10.25236/AJCIS.2025.080608

[7] Li Yuanxin, Chi Yuejie. Off-the-Grid Line Spectrum Denoising and Estimation With Multiple Measurement Vectors[J]. IEEE Transactions on Signal Processing, 2016.
[8] Boyd Stephen. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers[J]. Foundations and Trends® in Machine Learning, 2010.
[9] Kukreja Sunil L., Löfberg Johan, Brenner Martin J. A LEAST Absolute Shrinkage And Selection Operator (Lasso) For Nonlinear System Identification[J]. IFAC Proceedings Volumes, 2006.
[10] Emmanuel J. Candès, Justin K. Romberg, Terence Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information[J]. IEEE Transactions on Information Theory, 2006.