CLC ___________ Number___________

UDC___________ Available for reference ☐Yes ☐No

Southern University of Science and Technology

Undergraduate Thesis

Thesis Title:	**Decentralized Recommendation System Based on BSO**
Student Name:	**Tengfei Liu**
Student ID:	**11911720**
Department:	**Computer Science and Engineering**
Program:	**Computer Science and Engineering**
Thesis Advisor:	**Professor Yuhui. Shi**

Date: June 13, 2024

Commitment of Honesty

1. I solemnly promise that the paper presented comes from my independent research work under my supervisor's supervision. All statistics and images are real and reliable.

2. Except for the annotated reference, the paper contents no other published work or achievement by person or group. All people making important contributions to the study of the paper have been indicated clearly in the paper.

3. I promise that I did not plagiarize other people's research achievement or forge related data in the process of designing topic and research content.

4. If there is violation of any intellectual property right, I will take legal responsibility myself.

Signature: 刘腾飞

Date: June 13, 2024

Decentralized Recommendation System Based on BSO

[**ABSTRACT**]: This thesis primarily explains how the Brainstorming Optimization Algorithm (BSO) can optimize the FedFast (Federated Fast Algorithm), which is prone to falling into local optima. Federated learning recommendation systems are an excellent solution for companies to provide personalized recommendations while protecting personal privacy. In this common architecture, a central server aggregates model updates from various clients and then feeds the updated model back to the clients. This way, the training data of the clients is stored only on local devices and is not uploaded to the server. FedFast uses a clustering method based on user similarities to accelerate the convergence of the federated learning recommendation system, grouping clients based on the model updates and aggregating updates within each group. However, this method may lead to similar clustering results in successive iterations, causing the model to fall into local optima and fail to achieve global optima. Therefore, we propose a new method in this thesis, called FedBSO, which uses the Brainstorming Optimization Algorithm to optimize the aggregation process. Performance evaluations conducted on three datasets show that FedBSO outperforms FedFast in terms of Hit Rate (HR) and Normalized Discounted Cumulative Gain (NDCG).

[**Key words**]: Federated Learning, Recommendation System, Brainstorming Optimization Algorithm

[摘要]：本文主要说明了 BSO（头脑风暴优化算法）如何优化 FedFast（联邦快速算法）容易陷入局部最优解的问题。联邦学习推荐系统是企业在保护个人隐私的同时能够提供个性化推荐的优秀解决方案。在这种常见的架构中，中心服务器聚合各客户端的模型更新，再将更新后的模型反馈给客户端。这样，客户端的训练数据就仅存储在本地设备上，无需上传至服务器。FedFast 利用一种基于用户间相似性的聚类方法来加速联邦学习推荐系统的收敛，通过模型更新情况将客户端进行分组，并在各个群组内进行更新聚合。尽管如此，这种方式可能导致连续多次迭代中聚类结果相似，从而使模型陷入局部最优解，无法达到全局最优。因此，我们在本文中提出了一种新的方法，称为 FedBSO，该方法采用头脑风暴优化算法对聚合过程进行优化。在 3 个数据集进行的性能评估表明，与 FedFast 相比，FedBSO 在命中率（HR）和归一化折损累计增益（NDCG）方面表现更佳。

[关键词]：联邦学习，推荐系统，头脑风暴优化算法

Table of Content

1. Introduction

1.1 Background

Recommender systems are crucial tools that assist users in finding relevant information and services in a timely and appropriate manner. They achieve this by analyzing user preferences and behavior to provide tailored suggestions. However, this level of personalization requires access to personal data, leading to a dilemma where the advantages of customization must be weighed against the potential privacy concerns of the users.

With the introduction of stricter data privacy regulations such as the GDPR[1] in Europe, along with increasing public concern over personal data misuse, the importance of balancing personalization with privacy in recommender systems has grown. This has pushed the development of recommender technologies that prioritize user privacy.

Advancements are being made towards creating recommender systems that maintain high personalization standards without heavily depending on personal data. Innovations include employing techniques like federated learning, which allows data to remain on the user's device, and incorporating methods like differential privacy to ensure that the information used cannot be traced back to any individual. The goal of these advancements is to refine the capability of recommender systems to offer personalized suggestions while strictly adhering to privacy-preserving measures. The evolution of these systems continues to focus on achieving the dual objectives of effective personalization and robust privacy protection.

The traditional centralized training methods for recommender systems (RS), which often handle databases containing data on millions of users and items, face numerous challenges. These challenges include the significant demands on computational power and storage, escalating the operational costs for organizations. Furthermore, centralized systems are prone to creating potential privacy risks and efficiency bottlenecks.

Federated Learning (FL)[2], as an alternative, shifts the paradigm by decentralizing the training process of RS models. This shift allows the computational tasks to be distributed across a multitude of user devices, rather than relying on a central server. By keeping the personal data localized on individual devices, FL significantly enhances the security and privacy of user information. This decentralized approach also reduces the load on each device, as it allows for simpler models that are more manageable on smaller, less powerful devices.

This research paper delves into the potential of combining recent innovations in neural network-driven recommender[3-4] systems with the decentralized framework of FL. Such a combination is poised to address the dual challenges of scalability and data privacy that plague traditional RS. By integrating cutting-edge neural network techniques, which can refine and personalize recommendations more effectively, with FL's capability to process data locally on user devices, the resulting system can deliver highly accurate recommendations without compromising user privacy[5-6].

Exploring this integrated approach could revolutionize the field of RS by making them more adaptable to the growing demands for efficiency and data protection. This could lead to the development of recommender systems that are not only more responsive to user needs but also more compliant with global data protection standards, thereby setting a new benchmark in the balance between user-centric performance and privacy preservation.

1.2 Motivation

Most existing applications of Federated Learning (FL)[5] in recommender systems prioritize enhancing both accuracy and privacy, as noted in various studies. FL's unique capability allows for the training of a global model while ensuring that users' data never leaves their personal devices. However, the level of privacy preservation offered by FL is still under scrutiny. Critics argue that private information might still be deduced from shared model parameters, a concern highlighted in some recent research. This possibility opens up debates about the true effectiveness of FL as a privacy-preserving technology in the context of recommender systems.

Another significant challenge associated with FL-based systems is the distribution of costs associated with model training. Typically, the burden falls predominantly on the users, who may suffer from reduced device performance and increased data usage due to the heavy communication requirements needed during the model updating process. This increase in resource consumption can lead to slower device operation, potentially frustrating users. Furthermore, the latency in updating and improving the model means that users might have to endure suboptimal recommendation quality for extended periods. Such drawbacks can negatively impact user satisfaction and may lead to lower adoption rates of the system, especially if users are keen on immediate and high-quality personalized recommendations.

Given these complexities[2], it is crucial for developers and researchers working with FL to delve deeper into the trade-offs involved. There is a pressing need to innovate ways to streamline communication and computation processes within FL systems to speed up the convergence of the model without compromising the quality of recommendations or the privacy of user data. Enhancing the efficiency of FL could involve optimizing data transmission methods, developing lightweight model architectures, or devising more effective incremental learning strategies that require less frequent updates yet still capture significant improvements in the model. Understanding these dynamics is essential for advancing FL technology and making federated recommender systems both faster and more accurate, ultimately enhancing user experience and acceptance of such systems.

FedFast has achieved considerable success in addressing the communication and computational efficiency issues within federated learning systems. However, its reliance on clustering methods based on user similarity has inherent limitations. Specifically, the clustering operation tends to produce highly similar results across multiple training phases, which can cause the model to converge to local optima, preventing the achievement of a global optimum.

To overcome this challenge, the introduction of the Brainstorming Optimization Algorithm (BSO)[7] is worth considering. This algorithm employs dynamic strategies for replacing and exchanging cluster centers, effectively breaking the repetitive patterns formed by traditional clustering methods. Consequently, this prevents the model from stagnating at local optima. Such an approach not only enhances the global search capabilities of the model but also optimizes the overall performance of the recommender system, making it more precise in meeting user needs, enhancing user experience, and deepening users' trust and acceptance of the system. The integration of innovative algorithms with practical technologies is key to advancing federated learning technologies.

1.3 Contributions

Thus, we introduce FedBSO, a framework[8] based on Generalized Matrix Factorization (GMF) that, while slightly slower in convergence compared to FedFast[9], more closely approximates the performance of centralized models. This approach balances the trade-off between convergence speed and model accuracy, particularly in how it handles the intricacies of data distribution across different clients in a federated learning environment. By integrating GMF, FedBSO enhances its capability to tackle sparse data scenarios commonly encountered in decentralized settings, thereby improving the robustness and reliability of the recommendations. This strategic incorporation of GMF not only aids in achieving a more accurate approximation of centralized algorithms but also ensures that the model remains effective across diverse and distributed datasets, making it a more versatile and reliable solution in federated learning scenarios.

1.4 Organization of the Thesis

This thesis is organized into five chapters:

Chapter 1 introduces the background and motivation for the research on improving federated learning recommendation systems using the Brainstorming Optimization (BSO) algorithm. It highlights the key contributions of the thesis and provides an overview of the thesis structure.

Chapter 2 presents the fundamental theories and related work relevant to the research. It covers the problem formulation, matrix factorization techniques, learning from implicit feedback data, and evaluation metrics. The chapter also discusses related algorithms, including GMF[4], BSO[7], and FedFast.

Chapter 3 describes the design and research methods employed in this study. It starts by stating the problem observed in the replication of the FedFast algorithm and proposes enhancements using BSO concepts. The chapter then details the FedBSO algorithm, explaining the key steps and parameters involved.

Chapter 4 focuses on the experimental setup and results. It introduces the datasets used, the evaluation metrics, the baselines for comparison, and the parameter settings. The chapter presents the experimental results, analyzing the accuracy and convergence rate of the FedBSO algorithm compared to the baselines.

Chapter 5 concludes the thesis by summarizing the key findings, including the performance comparison, convergence speed, stability in training, the plateau phase, and implications for federated learning. It highlights the effectiveness of FedBSO in enhancing federated learning recommendation systems and suggests future research directions.

2. Fundamental Theories and Related Word

2.1 Fundamental Theories

This chapter provides a comprehensive overview of the theoretical foundations and prior studies pertinent to our research. It begins with a discussion on fundamental theories that underpin the research, followed by an exploration of algorithms and methodologies from related work that have informed our approach.

2.1.1 Problem Formulation

Assume M and N represent the total numbers of users and items, respectively[4]. We construct a user-item interaction matrix $Y \in \mathbb{R}^{M\times N}$ based on the implicit feedback from users, defined as:

$$y_{ui} = \begin{cases} 1 & \text{if there is a recorded interaction between user } u \text{ and item } i; \\ 0 & \text{otherwise.} \end{cases}$$

In this context, a value of 1 for y_{ui} signifies a recorded interaction between user u and item i, though it does not confirm whether u prefers i. Conversely, a value of 0 does not imply dislike; it may simply indicate that the user is not familiar with the item. These dynamics introduce complexities in learning from implicit data as it primarily offers noisy indications of user preferences. While entries that are observed provide some insight into what users might be interested in, entries that are not observed often represent mere gaps in the data, and the lack of negative feedback inherently limits the available information.

The issue of generating recommendations based on implicit feedback is essentially the task of predicting the scores for unobserved entries in the matrix Y, which serve to rank the items. Model-based methods posit that such data can be generated or represented by a foundational model. More formally, this can be described as:

$$\hat{y}_{ui} = f(u, i \mid \Theta),$$

where $\hat{y}_{ui}$ represents the predicted interaction score between user u and item i, Θ refers to the parameters of the model, and f is defined as the function that maps these parameters to the predicted score, commonly termed as the *interaction function*.

2.1.2 Matrix Factorization

Matrix Factorization (MF)[10] techniques associate each user u and item i with a corresponding real-valued feature vector. Denote $\mathbf{P}_u$ and $\mathbf{Q}_i$ as the latent feature vectors for user u and item i, respectively. The interaction between a user and an item is estimated by the inner product of their respective latent vectors:

$$\hat{y}_{ui} = \langle \mathbf{P}_u, \mathbf{Q}_i \rangle = \mathbf{P}_u^\top \mathbf{Q}_i = \sum_{k=1}^{K} p_{uk} q_{ik},$$

where K is the number of latent factors, representing the dimensionality of the latent space.

MF models not only facilitate a quantifiable estimation of user-item interactions but also simplify the computation by assuming a linear relationship between the latent dimensions. This assumption posits that each dimension of the latent space contributes independently and equally to the overall interaction score, effectively modeling the interactions as a linear combination of the latent features. Consequently, this approach conceptualizes the MF model as a linear model of latent factors, where the complexity of user-item relationships is captured in the simplicity of linear algebra.

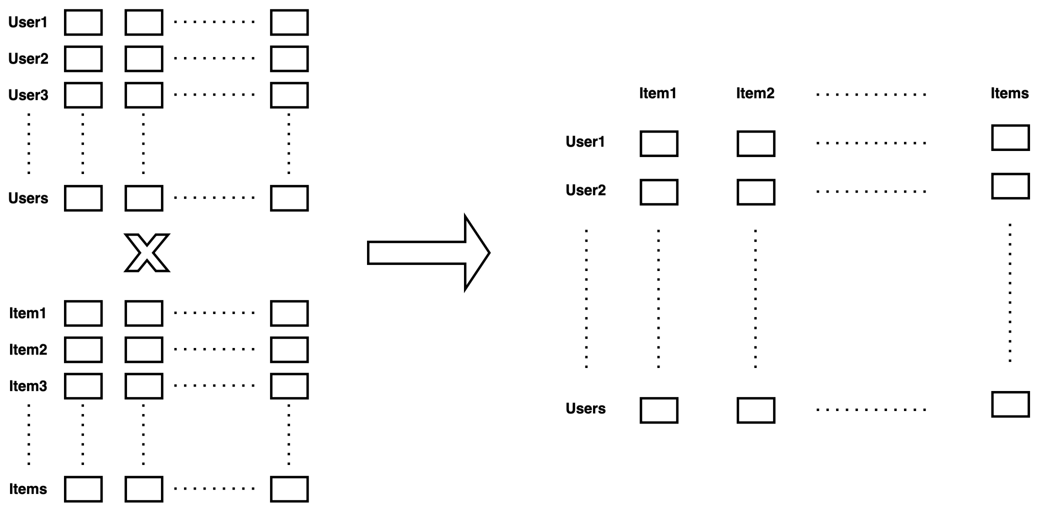


Figure 1 MF

2.1.3 Learning MF

To train model parameters, many current pointwise methods[11-12] primarily use regression with squared loss:

$$L_{\text{sqr}} = \sum_{(u,i)\in Y\cup Y^-} w_{ui}(y_{ui} - \hat{y}_{ui})^2$$

In this context, Y denotes the set of observed interactions, while Y^- represents the set of negative instances, which can either encompass all unobserved interactions or a sampled subset. The parameter w_{ui} serves as the weight for each training instance (u, i). While the squared loss is offten justified by assuming observations are drawn from a Gaussian distribution, this assumption is not well-suited for implicit feedback data. Implicit data involves binary target values y_{ui}, indicating whether user u has interacted with item i.

Hence a probabilistic approach for learning pointwise MF is more suitable, which duly considers the binary nature of implicit data. Here, y_{ui} functions as a binary label: a value of 1 signifies item i is relevant to user u, while 0 indicates irrelevance To imbue NCF with a probabilistic interpretation, we constrain $\hat{y}_{ui}$ within the interval [0, 1] by employing a probabilistic function (e.g., Logistic or Probit function) as the activation function for the output layer.

Under these settings, we define the likelihood function as:

$$p(Y, Y^{-}|P, Q, \Theta_f) = \prod_{(u,i)\in Y} \hat{y}_{ui} \prod_{(u,j)\in Y^{-}} (1 - \hat{y}_{uj})$$

By taking the negative logarithm of this likelihood function, we derive:

$$L = -\sum_{(u,i)\in Y} \log \hat{y}_{ui} - \sum_{(u,j)\in Y^{-}} \log(1 - \hat{y}_{uj})$$

This can be further simplified to:

$$L = -\sum_{(u,i)\in Y\cup Y^{-}} [y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui})]$$

This objective function is minimized using stochastic gradient descent (SGD). Notably, this formulation is equivalent to the binary cross-entropy loss, also referred to as log loss. By adopting a probabilistic framework for MF, we address the recommendation problem with implicit feedback as a binary classification task.

2.1.4 Evaluation Metrics

In the context of evaluating recommender systems, we often use metrics such as Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG)[13]. Below, we provide detailed explanations of these metrics.

2.1.5 Hit Ratio (HR)

Hit Ratio, also known as Recall, measures the fraction of relevant items that are successfully recommended[14]. It is defined as follows:

$$HR@k = \frac{1}{|U|} \sum_{u\in U} I(\text{hit}(@k)) \tag{1}$$

where U is the set of users, and $I(\text{hit}(@k))$ is an indicator function that returns 1 if any of the top k recommended items are relevant to the user u, and 0 otherwise. Essentially, HR@k checks whether at least one relevant item is present in the top k recommendations for each user.

2.1.6 Normalized Discounted Cumulative Gain (NDCG)

NDCG[13] is a more sophisticated metric that considers the position of the relevant items in the recommended list. It assigns higher scores to relevant items that appear higher in the ranking. NDCG is defined as follows:

$$NDCG@k = \frac{1}{|U|} \sum_{u \in U} \frac{DCG@k}{IDCG@k} \tag{2}$$

where $DCG@k$ (Discounted Cumulative Gain) is calculated as:

$$DCG@k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{3}$$

In this formula, rel_i denotes the relevance score of the item at position i. For binary relevance (0 or 1), rel_i is 1 if the item is relevant and 0 otherwise. $IDCG@k$ (Ideal DCG) is the maximum possible DCG@k, used to normalize the DCG score, making NDCG a value between 0 and 1.

By utilizing HR and NDCG, we gain insights into both the presence of relevant items in the recommendations (HR) and the ranking quality of those items (NDCG).

2.2 Related Work

2.2.1 GMF Algorithm

Matrix factorization algorithms have already demonstrated impressive performance in the field of recommendation systems. The generalized matrix factorization (GMF) model, which incorporates neural networks[3-4,6], can further enhance this performance. Therefore, we have chosen GMF as the foundational model for FedBSO, FedAVG, and FedFast to achieve superior recommendation results.

The GMF model is designed to learn the latent vector representation of users and items by leveraging neural networks.

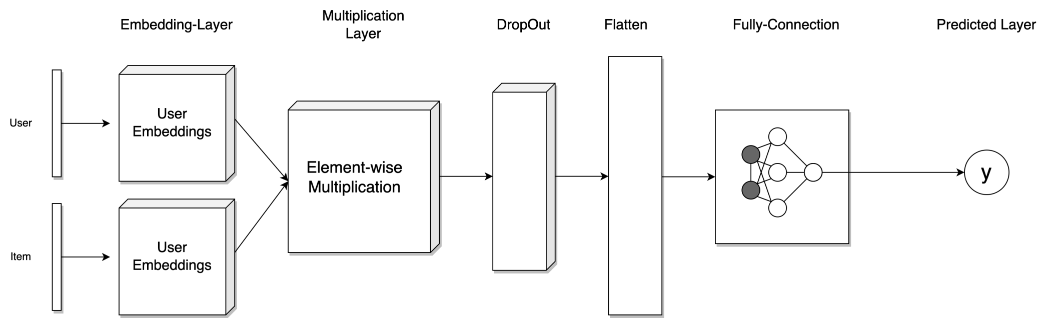


Figure 2 GMF Architecture

Due to the one-hot encoding of user (item) IDs in the input layer, the resulting embedding vector can be interpreted as the latent vector for the user (item). Let the user latent vector $\mathbf{p}_u$ be represented as $\mathbf{P}^T\mathbf{v}_u^U$ and the item latent vector $\mathbf{q}_i$ as $\mathbf{Q}^T\mathbf{v}_i^I$. We define the transformation function of the initial neural collaborative filtering layer as:

$$\phi_1(\mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u \odot \mathbf{q}_i,$$

where $\odot$ signifies the element-wise[11] product of vectors. This vector is then projected to the output layer:

$$\hat{y}_{ui} = a_{\text{out}}(\mathbf{h}^T(\mathbf{p}_u \odot \mathbf{q}_i)),$$

where a_{out} and $\mathbf{h}$ represent the activation function and edge weights of the output layer, respectively. Conceptually, if an identity function is used for a_{out} and $\mathbf{h}$ is set as a uniform vector of ones, the matrix factorization (MF) model can be precisely recovered.

2.2.2 BSO Algorithm

Optimization algorithms have been widely utilized in various fields to solve complex problems. Traditional optimization algorithms, such as evolutionary algorithms and swarm intelligence algorithms, are inspired by natural phenomena. However, the Brain Storm Optimization (BSO) algorithm[7] is a novel approach inspired by the human brainstorming process. The BSO algorithm leverages the creative problem-solving abilities of humans to enhance optimization performance.

The brainstorming process involves generating a diverse set of ideas through collaborative efforts. It typically follows these steps:

1. Form a diverse group of individuals.
2. Generate as many ideas as possible without judgment.
3. Select promising ideas and refine them further.
4. Use selected ideas to generate more ideas, iterating the process until a satisfactory solution is found.

This collaborative and iterative approach to problem-solving inspired the development of the BSO algorithm.

```
Data: Population size $N$, Number of clusters $K$
Result: Best solution found
Initialize the population with $N$ solutions;
Evaluate the fitness of each solution;
while stopping criteria not met do
    Group the population into $K$ clusters;
    Select one or more clusters probabilistically;
    Select one or more solutions from the selected clusters;
    Perform a combination (or perturbation) operation on the selected solutions to
      generate new solutions;
    Evaluate the fitness of the new solutions;
    Select the best solutions to form the new population;
end
Output the best solution found;
```

Algorithm 1: Brain Storm Optimization (BSO) Algorithm

2.2.3 FedFast Algorithm

FedFast[9] is a federated learning algorithm designed to enhance the efficiency and accuracy of training recommender systems. Unlike traditional federated learning approaches that randomly select users and average their local models to form a global model, FedFast introduces novel techniques to accelerate the training process and improve the quality of recommendations from early training stages. Here’s an overview of the FedFast algorithm and its components.

User Clustering: Users are grouped into clusters based on their similarities. This clustering process aims to exploit the homogeneity among users to facilitate more efficient learning. **Active Sampling (ActvSAMP)**: From each cluster, a subset of users is actively sampled. This sampling process ensures that representative users from each cluster are selected for the subsequent parameter update phase**??**.

Parameter Update: The selected users from each cluster update their local model parameters. This update is based on the user’s local data and the current state of the model. **Active Aggregation (ActvAGG)**: The updated parameters from the sampled users are then aggregated to form the global model. This aggregation step involves combining the updates from all clusters to refine the overall model parameters, ensuring that the global model benefits from the diverse data distributed across different users.

3. Design and Research methods

3.1 Problem Statement

In our attempt to replicate the FedFast[9] algorithm, we observed a significant issue where the algorithm fails to learn effectively in the latter stages of training. This stagnation in learning, we hypothesize, is due to the inherent nature of the clustering process in FedFast. The clustering results often remain unchanged or very similar across consecutive iterations, which suggests that the algorithm is converging to local optima and is unable to explore other potential solutions.

To mitigate this issue, we proposed an enhancement to the FedFast algorithm by integrating concepts from the Brainstorming Optimization (BSO)[7] algorithm. The key idea behind BSO is to introduce dynamic changes to the clustering process, thereby preventing the algorithm from getting stuck in repetitive patterns.

Steps for Enhancement:

- **Dynamic Cluster Center Adjustment**: During each clustering iteration, we introduce randomness in the selection of cluster centers. This involves randomly altering the cluster centers, which helps in exploring a broader solution space and avoiding local optima.

- **Cluster Center Swapping**: We further enhance the clustering process by probabilistically swapping the centers of different clusters. This method ensures that the clusters do not become static and encourages more dynamic interactions between data points.

By applying these BSO[7,15] principles, we aim to disrupt the repetitive clustering patterns that FedFast experiences, thereby improving its ability to learn and converge effectively. These modifications are intended to enhance the model's performance by ensuring that it does not get confined to suboptimal solutions and can continue to learn and adapt throughout the training process. This innovative approach not only addresses the limitations observed in FedFast but also leverages the strengths of BSO to achieve more robust and reliable model training outcomes.

3.2 FedBSO Algorithm

Parameter Notation: We introduce the following notation for the parameters in the FedFast algorithm. Let $\mathcal{K}$ represent the entire set of clients participating in the federated learning process. The complete set of parameters for the recommendation model is denoted by **w**, which encompasses both user and item embeddings, along with any additional parameters. The number of parameters in **w** is determined by the specific model architecture being used. For a given index j, we use the notation $w[j]$ to refer to the j-th component of the parameter set. This notation can be extended to $w[\mathcal{J}]$ to represent the subset of parameters indexed by any subset $\mathcal{J}$ of the full index set.

We further define $\mathcal{U}$ as the set of indices corresponding to all user embeddings in the model. For each client $k \in \mathcal{K}$, we use $\mathcal{U}k$ to denote the indices of the user embeddings associated with that particular client. The union of all client-specific user embedding indices forms the complete set of user embedding indices, i.e., $\mathcal{U} = \bigcup k \in \mathcal{K}\mathcal{U}_k$.

Similarly, let $\mathcal{I}$ represent the set of indices corresponding to all item embeddings in the model. During each round of the FedFast algorithm, a client k may update a subset of item embeddings, which we denote as $\mathcal{I}_k \subseteq \mathcal{I}$.

Finally, we use $\mathcal{N}$ to represent the indices of all other non-embedding parameters in the model. The complete set of parameter indices can be expressed as the union of user embedding indices, item embedding indices, and non-embedding indices, i.e., $\mathcal{U} \cup \mathcal{I} \cup \mathcal{N}$.

To enhance the performance of the FedFast algorithm, we incorporate a Brainstorming Optimization algorithm. This optimization algorithm is applied to randomly modify the clusters generated by the k-means algorithm, introducing an element of stochasticity to the clustering process. By randomly altering the clusters, the Brainstorming Optimization algorithm aims to explore a wider range of possible cluster configurations and potentially discover more optimal clusterings that can improve the overall performance of the FedFast algorithm.

The FedBso Algorithm

```
initialise $w_0, P_0, p, G_0$
for each round $t = 0, 1, \dots$ do
    $G_t \leftarrow$ ClusterUsersByCenter$(K, p, P, G_{t-1})$
    $G_t \leftarrow$ GetBestUser$(G_t)$
    Generate a random number $r_1$ uniformly in $[0, 1]$
    if $r_1 < p_5$ then
        Randomly select a user $u$
        Randomly replace the center of a cluster with $u$
    end if
    Generate a random number $r_2$ uniformly in $[0, 1]$
    if $r_2 < p_6$ then
        Randomly select two clusters $u$
        Randomly swap the center of two clusters
    end if
    for each cluster $c \in G_t$ in parallel do
        $m \leftarrow$ max$(\lceil \alpha * len(c) \rceil, 1)$
        $C \leftarrow$ Randomly select $m$ users from $c$
        for each user $k \in C$ in parallel do
            $w_{t+1}^k \leftarrow$ ClientUpdate$(k)$
        end for
        $W[C] \leftarrow \frac{\sum_{k \in C} n_k w_{t+1}^k}{\sum_{k \in C} n_k}$
    end for
end for
```

Algorithm 2: The FedBSO algorithm

The FedBSO algorithm, delineated in Algorithm 2, advances the federated learning frontier by optimizing the aggregation of model updates. It commences with initializing the model parameters w_0, the clustering from the previous round P_0, and a probability threshold p. Each subsequent round t unfolds as follows:

- **User Clustering:** Leveraging 'ClusterUsersByCenter', the algorithm clusters users based on the set of all clients K, the threshold p, the clustering from the prior round P, and the previous cluster centers G_{t-1}, resulting in an updated set of cluster centers G_t.

- **Stochastic Cluster Center Modification:** The algorithm injects randomness into the clustering process; if a uniformly generated random number r_1 is less than the threshold p_5, it randomly selects a user u to replace the center of an existing cluster.

- **Cluster Center Swapping:** A second random decision occurs if a generated number r_2 falls below the threshold p_6, wherein two clusters are chosen, and their centers are swapped. This step introduces additional randomness that can prevent convergence to local optima.

- **Active Sampling and Updating:** Within each cluster c, the algorithm samples m users, dictated by $\max(\lceil \alpha \cdot \text{len}(c) \rceil, 1)$. For every user k in the sample C, the algorithm updates their model parameters w_{t+1}^k in parallel via 'ActvSamp', taking into account the current model parameters w_t, user data n_k, probability threshold p, and a decay factor y.

- **Weighted Aggregation:** Post updates, a weighted aggregation of the parameters is conducted where $W[C]$ is computed as the weighted average, factoring in the contribution significance or data volume from each user k via n_k.

The FedBSO algorithm iterates over these steps for a pre-set number of rounds or until it meets convergence criteria. By interspersing random elements in cluster center selection and employing a weighted averaging approach, FedBSO aims to strike a strategic balance between exploration and exploitation, crucial for robust global model achievement in federated learning settings.

4. Experiments

4.1 Experimental Setup

In this section, we evaluate FedBSO with the aim of answering the research questions described in the introduction section.

4.1.1 Datasets

Datasets We experimented with three datasets: Movielens 100K (ML100K), Movielens 1M(Ml-1M), Printerest.

MovieLens The MovieLens dataset, commonly utilized for the evaluation of collaborative filtering algorithms, was employed in this study. Specifically, we used the version comprising one million ratings, ensuring that each user has provided at least 20 ratings. Although this dataset originally represents explicit feedback, we chose to use it to explore the performance of learning from implicit signals derived from explicit feedback[16]. For this purpose, we converted the dataset into implicit feedback data by marking each entry as either 0 or 1, indicating whether the user has rated the item.

Pinterest The Pinterest dataset, originally constructed by[17] for evaluating content-based image recommendation, consists of implicit feedback data. The raw dataset is notably large yet highly sparse, with over 20% of users having only a single pin, presenting challenges for the evaluation of collaborative filtering algorithms. Therefore, similar to our preprocessing of the MovieLens data, we filtered the Pinterest dataset to retain only users with at least 20 interactions (pins). This filtering process resulted in a subset containing 55,187 users and 1,500,809 interactions. Each interaction indicates whether a user has pinned an image to their own board.

Table 1 Statistics of the evaluation datasets

Dataset	Interaction#	Item#	User#	Density
ML100K	100,000	1,628	943	6.3%
ML1M	1,000, 296	3706	6040	4.5%
Pinterest	1,500,809	9,916	55,187	0.27%

4.1.2 Evaluation Metrics

To evaluate the performance of item recommendation, we adopted the leave-one-out evaluation protocol, which has been widely used in the literature[18-19]. In this protocol, for each user, we held out their latest interaction as the test set, utilizing the remaining data for training the recommendation model. This method simulates a realistic scenario where the most recent user activity is unknown and needs to be predicted.

Given the computational inefficiency of ranking all items for every user during evaluation, we followed a common strategy[16,20] to mitigate this issue. Specifically, for each user, we randomly sampled 100 items that the user had not interacted with and ranked the test item among these 100 items. This approach reduces the evaluation complexity while maintaining the robustness of the performance metrics.

The performance of the ranked list is assessed using two widely recognized metrics: Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG)[13]. Hit Ratio at rank k (HR@k) measures the presence of the test item within the top-k positions of the ranked list. In our evaluations, unless stated otherwise, we set $k = 20$, thus evaluating HR@20. This metric provides an intuitive measure of whether the test item appears in the top-20 recommendations.

Normalized Discounted Cumulative Gain at rank k (NDCG@k) evaluates the quality of the ranked list by considering the positions of the relevant items. NDCG assigns higher scores to hits at higher ranks, emphasizing the importance of correctly ranking relevant items near the top of the list. We also set $k = 20$ for NDCG, thus evaluating NDCG@20. This metric captures the ranking quality and helps in understanding the effectiveness of the recommendation model in placing relevant items higher in the list.

For each test user, we calculated HR@20 and NDCG@20 and reported the average scores across all users. This comprehensive evaluation provides insights into both the presence and the ranking quality of relevant items in the recommendations, offering a balanced view of the model's performance.

4.1.3 Baselines

We compared the performance of FedBSO with several state-of-the-art recommendation algorithms, including:

- **GMF.** GMF[4] serves as the foundational model for all our methods. It represents the upper bound of all federated learning models since it is not limited by the constraints of federated learning.

- **FedAVG.** This method[2] is a traditional federated learning algorithm known for its effectiveness in federated learning scenarios. It follows the settings described in **??** to adapt for implicit data[21].

- **FedFast.** FedFast[9] employs clustering operations similar to those in FedBSO. By comparing these methods, we aim to explore the impact of the BSO component in FedBSO.

4.2 Parameter Settings

We set the hyperparameters as follows: the embedding size is 16, the number of rounds is set to 1000 (360 for the Pinterest dataset due to its large size), the number of clusters is set to 20, the optimizer used is Adam, p_5 is set to 0.5, p_6 is set to 0.5, the learning rate is 0.1, and the batch size is 256, with a sampling ratio of 0.1 as shown in the figure.

4.3 Experimental Results

In this section, we present the experimental results of the FedBSO algorithm on the three datasets. We compare the performance of FedBSO with the baselines using the evaluation metrics described in the previous section.

4.3.1 The accuracy of the FedBSO algorithm

The table shows the performance of different models on three different datasets (ML100K, ML1M, and Pinterest) based on two metrics: HR (Hit Rate) and NDCG (Normalized Discounted Cumulative Gain).

Firstly, as expected, GMF (Generalized Matrix Factorization) has consistently maintained a leading position in our model performance analysis. This is mainly because all our models are built on the foundation of GMF. As a centralized model, GMF has demonstrated significant advantages in terms of performance.

The strength of GMF lies in its use of centralized data processing and computation, allowing it to fully utilize all available data for training, thereby achieving optimal performance. In the context of federated learning, the data from each participating party is distributed and cannot be directly accessed in its entirety. Even though we use GMF as the underlying model to build more complex systems in a federated learning environment, the performance is still constrained by data distribution and communication limitations. Hence, regardless of how much we optimize federated learning models, their performance ceiling is always set by the centralized GMF model.

Secondly, when comparing FedFast with FedAvg, it is evident that FedFast consistently outperforms FedAvg across all datasets, even with the same number of training rounds. This significant improvement can be attributed to the incorporation of clustering techniques in the FedFast model. By introducing the concept of clustering, FedFast leverages the similarity between users to enhance learning efficiency. This clustering mechanism allows the model to maximize the utilization of user similarities, leading to superior performance compared to FedAvg.

FedBSO introduces further improvements on top of FedFast. While FedFast accelerates the convergence of model parameters, it has a drawback: the clustering results tend to remain similar across training rounds. This similarity can easily lead the model parameters to get stuck in local optima. FedBSO addresses this issue by introducing a probabilistic mechanism that exchanges or replaces cluster centers. This mechanism disrupts the established clustering results to some extent, allowing for reorganization and potentially avoiding local optima.

Table 2

		Federated			Centralised
		FedBSO	FedAvg	FedFast	GMF
ML100K	HR	**0.8204**	0.7328	0.8091	**0.8378**
	NDCG	**0.4335**	0.3565	0.4237	**0.4442**
ML1M	HR	**0.8534**	0.6917	0.8346	**0.8577**
	NDCG	**0.4489**	0.3195	0.4325	**0.4560**
Pinterest	HR	**0.9529**	0.4604	0.9481	**0.9632**
	NDCG	**0.5603**	0.1946	0.5512	**0.5814**

4.3.2 The convergence rate of the FedBSO algorithm

Across three datasets, FedBSO consistently achieved good results, with final outcomes aligning closely with expectations. GMF, being a centralized model, converges the fastest and performs well due to its ability to learn from global information.

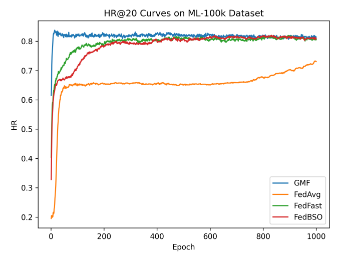
FedAvg, on the other hand, struggles to effectively learn sufficient information even with a large number of rounds. This limitation is inherent to its training structure. FedFast significantly outperforms FedAvg because it incorporates clustering operations. By leveraging the similarities between users, FedFast updates model parameters more efficiently, greatly accelerating convergence.

Building on the foundation of FedFast, FedBSO addresses the issue of local optima that FedFast can encounter. By mitigating this problem, FedBSO further enhances model performance, bringing it closer to that of centralized models.

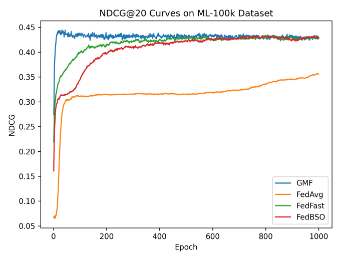
The improvement of FedBSO over FedFast is clearly evident in six graphs. FedBSO reaches performance levels close to GMF with fewer iterations, indicating that it has faster convergence and better overall performance.

In terms of HR@20 and NDCG@20, FedBSO consistently outperforms FedFast, demonstrating that its performance gains are generalizable across different datasets. Compared to FedFast, FedBSO shows more stability in the later iterations, with HR@20 and NDCG@20 values exhibiting minimal fluctuations. This highlights the advantages of FedBSO in model training, showcasing its superior and stable performance.

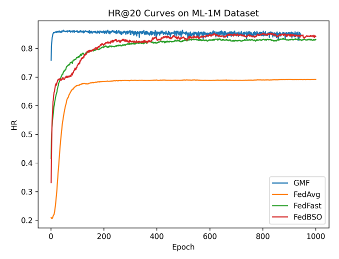
Throughout the convergence process, it is observed that FedBSO consistently experiences a small plateau phase. We hypothesize that this phenomenon is due to the inherent instability in the clustering operations during the early to middle stages of rapid convergence. At these stages, the frequent alterations in clustering results can disrupt the stability of parameter aggregation. This instability in parameter aggregation during the training period causes the model to temporarily plateau before resuming its convergence. Such behavior indicates that while FedBSO is efficient, the dynamic nature of its clustering mechanism requires a period of adjustment to stabilize the clustering results, ultimately leading to a more stable and effective convergence in later stages.



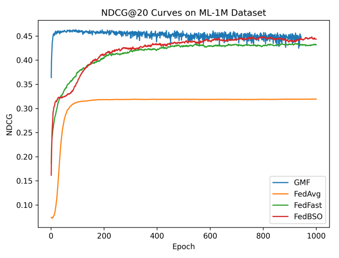
a) ml-100k HR@20



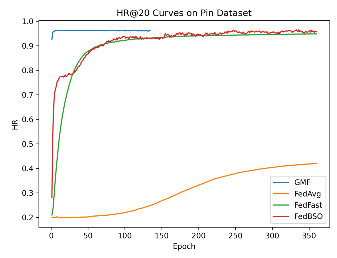
b) ml-100k NDCG@20



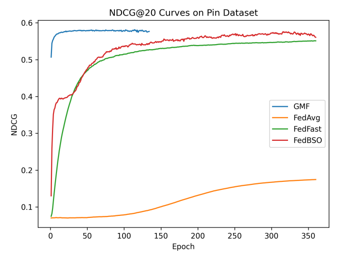
c) ml-1m HR@20



d) ml-1m NDCG@20



e) Pinterest HR@20



f) Pinterest NDCG@20

Figure 3 Figures side by side

5. Conclusion

In this thesis, we explored the effectiveness of the FedBSO algorithm in enhancing federated learning recommendation systems. The experimental results demonstrated that FedBSO consistently achieved superior performance across multiple datasets, confirming our hypothesis that incorporating the Brainstorming Optimization (BSO) algorithm can mitigate the limitations of existing federated learning models.

The FedBSO algorithm offers a significant advancement in federated learning recommendation systems by balancing the trade-offs between convergence speed and model accuracy. Its dynamic clustering and optimization strategies ensure efficient and stable performance, making it a valuable contribution to the field of federated learning. Future research could focus on further refining the clustering mechanisms and exploring the application of FedBSO in other domains requiring robust and scalable recommendation systems.

5.1 Key Findings

5.1.1 Performance Comparison

FedBSO consistently outperformed FedFast and FedAvg in terms of HR@20 and NDCG@20 metrics. This performance improvement was observed across all datasets used in our experiments. The results indicate that FedBSO's clustering and optimization strategies significantly enhance the accuracy and efficiency of the model.

5.1.2 Convergence Speed

While GMF, as a centralized model, achieved the fastest convergence due to its access to global information, FedBSO showed a marked improvement in convergence speed compared to FedFast and FedAvg. FedBSO's ability to reach performance levels close to GMF with fewer iterations underscores its efficiency in federated learning scenarios.

5.1.3 Stability in Training

FedBSO demonstrated more stable performance in later iterations, with minimal fluctuations in HR@20 and NDCG@20 values. This stability can be attributed to FedBSO's dynamic clustering mechanism, which prevents the model from getting stuck in local optima and ensures a smoother convergence trajectory.

5.1.4 Plateau Phase

During the convergence process, FedBSO exhibited a small plateau phase. This phenomenon can be explained by the instability in clustering operations during the early to middle stages of rapid convergence. Frequent changes in clustering results can disrupt the stability of parameter aggregation, causing temporary plateaus before the model resumes its convergence. This behavior indicates the need for a period of adjustment to stabilize the clustering results, ultimately leading to a more stable and effective convergence in later stages.

5.1.5 Implications for Federated Learning

The incorporation of BSO into federated learning frameworks provides a robust solution to the challenges of local optima and slow convergence. FedBSO's ability to dynamically adjust clustering centers and improve parameter aggregation highlights its potential for broader application in federated learning environments.

References

[1] ZHANG B, WANG N, JIN H. Privacy Concerns in Online Recommender Systems: Influences of Control and User Data Input[C/OL]. in: 10th Symposium On Usable Privacy and Security (SOUPS 2014). Menlo Park, CA: USENIX Association, 2014: 159-173. https://www.usenix.org/conference/soups2014/proceedings/presentation/zhang.

[2] MCMAHAN B, MOORE E, RAMAGE D, et al. Communication-Efficient Learning of Deep Networks from Decentralized Data[C/OL]. in: SINGH A, ZHU J. Proceedings of Machine Learning Research: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics: vol. 54. PMLR, 2017: 1273-1282. https://proceedings.mlr.press/v54/mcmahan17a.html.

[3] COVINGTON P, ADAMS J, SARGIN E. Deep Neural Networks for YouTube Recommendations[C/OL]. in: RecSys '16: Proceedings of the 10th ACM Conference on Recommender Systems. Boston, Massachusetts, USA: Association for Computing Machinery, 2016: 191-198. https://doi.org/10.1145/2959100.2959190. DOI: 10.1145/2959100.2959190.

[4] HE X, LIAO L, ZHANG H, et al. Neural Collaborative Filtering[C/OL]. in: WWW '17: Proceedings of the 26th International Conference on World Wide Web. Perth, Australia: International World Wide Web Conferences Steering Committee, 2017: 173-182. https://doi.org/10.1145/3038912.3052569. DOI: 10.1145/3038912.3052569.

[5] CHEN F, LUO M, DONG Z, et al. Federated Meta-Learning with Fast Convergence and Efficient Communication[J]., 2019. arXiv: 1802.07876 [cs.LG].

[6] NASR M, SHOKRI R, HOUMANSADR A. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning[C]. in: 2019 IEEE Symposium on Security and Privacy (SP). 2019: 739-753. DOI: 10.1109/SP.2019.00065.

[7] SHI Y. Brain Storm Optimization Algorithm[C]. in: TAN Y, SHI Y, CHAI Y, et al. Advances in Swarm Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011: 303-309.

[8] GRASS J, ZILBERSTEIN S. Anytime algorithm development tools[J/OL]. SIGART Bull., 1996, 7(2): 20-27. https://doi.org/10.1145/242587.242592. DOI: 10.1145/242587.242592.

[9] MUHAMMAD K, WANG Q, O'REILLY-MORGAN D, et al. FedFast: Going Beyond Average for Faster Training of Federated Recommender Systems[C/OL]. in: KDD '20: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. Virtual Event, CA, USA: Association for Computing Machinery, 2020: 1234-1242. https://doi.org/10.1145/3394486.3403176. DOI: 10.1145/3394486.3403176.

[10] RENDLE S. Factorization Machines[C]. in: 2010 IEEE International Conference on Data Mining. 2010: 995-1000. DOI: 10.1109/ICDM.2010.127.

[11] HE X, ZHANG H, KAN M Y, et al. Fast Matrix Factorization for Online Recommendation with Implicit Feedback[C/OL]. in: SIGIR '16: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. , Pisa, Italy, Association for Computing Machinery, 2016: 549-558. https://doi.org/10.1145/2911451.2911489. DOI: 10.1145/2911451.2911489.

[12] WANG M, FU W, HAO S, et al. Scalable Semi-Supervised Learning by Efficient Anchor Graph Regularization[J]. IEEE Transactions on Knowledge and Data Engineering, 2016, 28: 1-1. DOI: 10.1109/TKDE.2016.2535367.

[13] CHEN T, SUN Y, SHI Y, et al. On Sampling Strategies for Neural Network-based Collaborative Filtering[Z]. 2017. arXiv: 1706.07881 [cs.LG].

[14] DESHPANDE M, KARYPIS G. Item-based top-N recommendation algorithms[J/OL]. ACM Trans. Inf. Syst., 2004, 22(1): 143-177. https://doi.org/10.1145/963770.963776. DOI: 10.1145/963770.963776.

[15] CHENG S, QIN Q, CHEN J, et al. Brain storm optimization algorithm: a review[J]. Artificial Intelligence Review, 2016, 46: 445-458.

[16] KOREN Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C/OL]. in: KDD '08: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Las Vegas, Nevada, USA: Association for Computing Machinery, 2008: 426-434. https://doi.org/10.1145/1401890.1401944. DOI: 10.1145/1401890.1401944.

[17] GENG X, ZHANG H, BIAN J, et al. Learning Image and User Features for Recommendation in Social Networks[C]. in: 2015: 4274-4282. DOI: 10.1109/ICCV.2015.486.

[18] CHENG H T, KOC L, HARMSEN J, et al. Wide & Deep Learning for Recommender Systems[C]. in: 2016: 7-10. DOI: 10.1145/2988450.2988454.

[19] HE X, ZHANG H, KAN M Y, et al. Fast Matrix Factorization for Online Recommendation with Implicit Feedback[C/OL]. in: SIGIR '16: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. , Pisa, Italy, Association for Computing Machinery, 2016: 549-558. https://doi.org/10.1145/2911451.2911489. DOI: 10.1145/2911451.2911489.

[20] ELKAHKY A M, SONG Y, HE X. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems[C/OL]. in: WWW '15: Proceedings of the 24th International Conference on World Wide Web. Florence, Italy: International World Wide Web Conferences Steering Committee, 2015: 278-288. https://doi.org/10.1145/2736277.2741667. DOI: 10.1145/2736277.2741667.

[21] KOREN Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C/OL]. in: KDD '08: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Las Vegas, Nevada, USA: Association for Computing Machinery, 2008: 426-434. https://doi.org/10.1145/1401890.1401944. DOI: 10.1145/1401890.1401944.

First and foremost, I want to express my deepest gratitude to my girlfriend. During my five years of study at the Southern University of Science and Technology, I faced numerous challenges and setbacks. There were times when I would lie alone on the campus field, gazing up at the pitch-black night sky, with tears uncontrollably streaming down my face. Sometimes, I would helplessly hit my own body while taking a shower, as if it could alleviate the pain in my heart. Other times, I would act indifferent, as if everything in this world had nothing to do with me, losing all aspirations for life. However, when I hit rock bottom, I met my current girlfriend. It was her presence that made me rediscover the meaning of life. Because of her, I realized that in this world, there is still someone willing to selflessly devote herself to me, someone worth cherishing, respecting, and protecting. Her love, like a beacon guiding me through the darkness, dispelled the gloom that shrouded my heart. With her by my side, I no longer felt lonely and helpless. I rekindled my passion for life and became full of anticipation for the future. I know that no matter what difficulties I may encounter in the future, I am no longer fighting alone because she will always be there for me, giving me strength. Secondly, I want to express my heartfelt gratitude to my research supervisor, Professor Shi Yuhui, and my senior fellow apprentice, Qu Liang. During the most challenging moments of my graduation project, it was their immense help and support that pulled me through. My supervisor utilized his profound knowledge, keen insight, and rich experience to provide invaluable guidance for my research. He tirelessly answered my questions and helped me overcome one obstacle after another, encouraging me to bravely explore unknown academic territories. In particular, Senior Qu Liang gave me meticulous care and attention. He was not only a mentor and friend in my studies but also a brotherly companion in life. Whenever I felt lost and overwhelmed by pressure, he would always cheer me up and help me regain my confidence. It was under the careful guidance of my supervisor and senior fellow apprentice that I was able to successfully complete my graduation project and achieve satisfactory results. Looking back on my five-year journey of studying, although it was full of twists and turns, I was never fighting alone. My girlfriend, supervisor, and

senior fellow apprentice were the precious people bestowed upon me by heaven. It was their love, support, and help that gave me the courage to face all the difficulties and obstacles, shaping me into who I am today. In the future, I will always remember their kindness and repay them with my achievements and happiness. At the same time, I also hope to pass on this love and support to help and encourage more junior schoolmates who come after me, letting love and hope forever reside on our campus. Lastly, I sincerely wish all graduating students a bright future and a happy life! I also wish my alma mater, the Southern University of Science and Technology, continued success in cultivating more outstanding talents for our country! Thank you, everyone!