Graph Neural Controlled Differential Equations For Collaborative Filtering

Ke Xu
kxu25@uic.edu
University of Illinois Chicago
Chicago, USA

Weizhi Zhang
wzhan42@uic.edu
University of Illinois Chicago
Chicago, USA

Zihe Song
zsong29@uic.edu
University of Illinois Chicago
Chicago, USA

Yuanjie Zhu
yzhu224@uic.edu
University of Illinois Chicago
Chicago, USA

Philip S. Yu
psyu@uic.edu
University of Illinois Chicago
Chicago, USA

Abstract

Graph Convolution Networks (GCNs) are widely considered state-of-the-art for recommendation systems. Several studies in the field of recommendation systems have attempted to apply collaborative filtering (CF) within the Neural ODE framework. These studies follow the same idea as LightGCN, which either removes the weight matrix or employs a discrete weight matrix. However, we argue that weight control is critical for neural ODE-based methods. Weight plays a crucial role in creating tailored graph convolution for each node, and employing a fixed or discrete weight prevents adjustment over time within the ODE function. This rigidity in the graph convolution reduces its adaptability, consequently hindering the performance of recommendations. In this study, to create an optimal control for Neural ODE-based recommendation, we introduce a new method called Graph Neural Controlled Differential Equations for Collaborative Filtering (CDE-CF). Our method improves the performance of the Graph ODE-based method by incorporating weight control in a continuous manner. To evaluate our approach, we conducted experiments on various datasets. The results show that our method surpasses competing baselines, including GCNs-based models and state-of-the-art Graph ODE-based methods.

CCS Concepts

• **Information systems** → **Collaborative filtering**; **Recommender**.

Keywords

Graph Recommendation, Neural ODE, Collaborative Filtering

ACM Reference Format:
Ke Xu, Weizhi Zhang, Zihe Song, Yuanjie Zhu, and Philip S. Yu. 2025. Graph Neural Controlled Differential Equations For Collaborative Filtering. In *Companion Proceedings of the ACM Web Conference 2025 (WWW Companion '25), April 28-May 2, 2025, Sydney, NSW, Australia.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3701716.3715594

WWW Companion '25, Sydney, NSW, Australia

ACM ISBN 979-8-4007-1331-6/2025/04
https://doi.org/10.1145/3701716.3715594

1 Introduction

In recent years, there has been a surge in the popularity of Graph Convolutional Networks (GCNs) [16] for machine learning tasks involving graph data. GCNs have gained popularity in recent years for their effectiveness in learning node embeddings by exploiting the structure of graphs. Collaborative Filtering (CF) [7, 13] is a popular approach in recommender systems. Since GCNs can effectively capture relationships between users and items in a graph, they have shown promising results in improving the performance of collaborative filtering, particularly in scenarios with complex and sparse user-item interaction data.

Several studies [6, 14, 15] have shown that linear GCN architectures outperform non-linear ones for collaborative filtering [7]. Moreover, linear GCNs can be easily interpreted as an ordinary differential equation. This concept [1, 3] has led to the development of LT-OCF, a Neural Ordinary Differential Equations (NODEs)-based CF method [2]. LT-OCF demonstrates the suitability of NODEs-based approaches for collaborative filtering. The main idea behind LT-OCF is to create a continuous version of the GCN layer, resembling LightGCN [6] but with a customizable number of layers.

GODE-CF [12] is another NODE-based method for collaborative filtering. Inspired by of Graph Neural Ordinary Differential Equations (GODEs)[10], instead of creating a continuous message-passing layer, GODE-CF parametrizes the ODE using one or two GCN layers. It tries to utilize the information captured by these GCN layers to estimate the final state of the embedding by solving an ODE problem. Unlike LT-OCF [2], GODE-CF incorporates a discrete weight for each node embedding. However, it remains unclear whether the weight is helpful or not. The experimental results in GODE-CF indicate that the weight does not always improve performance across all cases. We argue that a discrete weight may also limit the performance of GODE-CF. ODE is a continuous form, and incorporating a weight matrix in a continuous manner should further enhance the performance of GODE-CF.

Motivated by this idea and based on the framework of GODE-CF, we propose a new method called Graph Neural Controlled Differential Equations for Collaborative Filtering (CDE-CF). Our method is based on the framework of GODE-CF. However, unlike GODE-CF, we incorporate MLPs to control the ODE instead of using a discrete weight for each node. Such MLP weight generator can be regarded as part of the ODE function producing continuous weight values for the continuous time slots. To evaluate the performance of our method, we use four public review datasets and compare them with state-of-the-art methods. The results indicate that our method consistently outperforms the mentioned methods in all datasets. Moreover, we showcase the efficiency of our method by demonstrating its faster training compared to most GCN-based methods. Furthermore, we explore the influence of different ODE solvers on various datasets. To encourage future exploration of CDE-CF, we have made our work open-source on https://github.com/DavidZWZ/CDE-CF. In summary, our contributions can be outlined as follows:

- We identify the limitations of non-weight and discrete weight in graph-based collaborative filtering and demonstrate the importance of the control matrix in ODE-based methods.
- We have developed CDE-CF, a novel method that can adaptively control the weight along the time and for different nodes in the Graph Neural ODE function for collaborative filtering.
- We conduct extensive experiments on four real-world datasets to test the effectiveness of CDE-CF. It achieves the highest performance with the same training time, demonstrating the remarkable efficacy of CDE-CF.

2 Preliminaries

Neural Ordinary Differential Equations (NODEs) refers to a method that is used to model the continuous dynamics of hidden states within neural networks [1]. This is achieved by characterizing the dynamics through an ordinary differential equation (ODE) that is parameterized by a neural network. The main objective of this method is to learn implicit differential equations from data. By employing neural networks to parameterize the ODEs, it becomes possible to capture intricate patterns in the data that would be difficult to capture using discrete methods. NODEs offer a framework for modeling complex systems by leveraging ODEs to capture continuous behavior. The formula for NODEs is written as follows:

$$h(t_1) = h(t_0) + \int_{t_0}^{t_1} f(h(t), \theta) dt \tag{1}$$

where f is a neural network parameterized by θ that approximates $\frac{dh(t)}{dt}$. This approximation allows us to derive $h(t_1)$ from $h(t_0)$. The parameter θ is trained using data. The variables t_0 and t_1 represent the starting and ending times, with t_0 often set to 0. In Neural ODEs, t_1 can be considered as the number of layers in a neural network.

3 Proposed Method

Graph Neural Ordinary Differential Equations-based method for Collaborative Filtering (GODE-CF) [12] is a method that draws inspiration from the concept of Graph-based NODEs [10]. Instead of creating a continuous message-passing layer, GODE-CF directly parameterizes the derivative function using one or two layers of GCNs. In other words, GODE-CF utilizes the information captured by two LightGCN layers to estimate the final state of the embedding by solving an ODE problem.

Different from LightGCN [6], GODE-CF does not involve layer combinations, as the integration can be viewed as the summation of all layers from time 0 to t_1. The initial embeddings serve as the input for the ODE, and the output of the ODE becomes the final

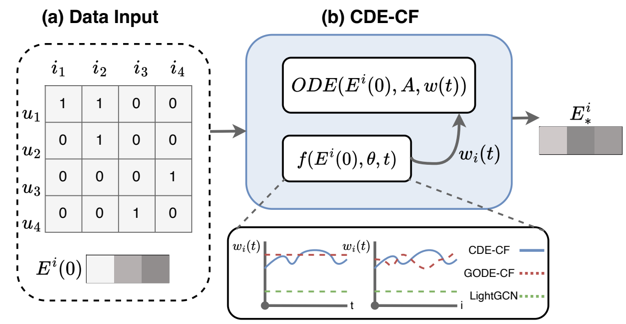

Figure 1: The graph convolution process for CDE-CF to generate an item final embedding E^i_* and the user embedding E^u_* can be produced in a similar process. (a) The data input contains the bipartite adjacency matrix and initial embedding. (b) The architecture of the proposed CDE-CF includes the ODE function to model the graph convolution and the weight generator to produce continuous weight. The plot below compares the weight value of CDE-CF, GODE-CF, and LightGCN with the change of time t and the node index i.

embedding. The overall formula can be expressed as:

$$\begin{aligned} E^u_* &= E^u(0) + \int_0^{t_1} W(A^n - I)E^i(t)dt \\ E^i_* &= E^i(0) + \int_0^{t_1} W(A^n - I)E^u(t)dt \end{aligned} \tag{2}$$

where the normalized adjacency matrix is denoted by A, and the initial user and item embeddings are represented by E^u_0 and E^i_0, respectively. The discrete weight matrix is denoted by W, and n represents the number of layers. The final user embeddings and item embeddings are denoted as E^u_* and E^i_*, respectively.

Unlike typical GCN-based models [6, 7], which combine the embeddings from all layers, GODE-CF estimates the final embeddings by leveraging information from multiple GCN layers through an ODE function. Similar to other methods, the embeddings will be trained using the BPR [11] loss and ODE solvers like explicit Euler and RK4(Runge-Kutta 4th order method) will be used to solve the ODE problem.

Distinct from LightGCN, which removes the weight matrix, ODE-based methods rely on a weight matrix to regulate the progression of each node toward its optimal state. Without weights, all node embeddings would converge in the same state at the same timestep, resulting in suboptimal embeddings for some nodes. Since nodes may require different timesteps to reach their optimal states, weights play a crucial role in ODE-based methods. GODE-CF introduces a discrete weight that improves performance in specific scenarios. However, this method has limitations due to the continuous nature of the overall framework. We argue that incorporating weights in a continuous manner would further enhance the performance of GODE-CF.

Instead of simply creating a weight matrix like GODE-CF, we propose CDE-CF which builds MLPs to control the ODE. This ensures that each node reaches its optimal state. The underlying concept is straightforward. Fig 1 demonstrates ODE-based modeling for graph

Table 1: Overall performance of CDE-CF in comparison with different state-of-the-art baselines on four datasets.

Dataset	Beauty		Health		Cell Phone		Office	
Method	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
NGCF	0.07079	0.02995	0.03064	0.01226	0.04387	0.01691	0.05097	0.022137
layerGCN	0.07620	0.03144	0.02453	0.01009	0.03967	0.01508	0.04710	0.02105
UltraGCN	0.05661	0.02618	0.03170	0.01348	0.03604	0.01558	0.04689	0.02336
GTN	0.07146	0.03059	0.03287	0.01351	0.04559	0.01780	0.04363	0.02026
LightGCN	0.07776	0.03299	0.03030	0.01202	0.04429	0.01660	0.05647	0.02635
LT-OCF	0.07879	0.03309	0.03022	0.01197	0.04641	0.01739	0.05626	0.02596
GODE-CF	0.08075	0.03406	0.03387	0.01356	0.05079	0.01909	0.05667	0.02702
CDE-CF	**0.08129**	**0.03426**	**0.03468**	**0.01375**	**0.05082**	**0.01945**	**0.05728**	**0.02713**

convolution and a controller to produce the weight varying from the time t and the node index i. We use the initial embedding as the input for the MLPs, and the output of the MLPs serves as the weight matrix, which is integrated into the ODE framework. The whole framework can be treated as two parts: (1) *A Neural ODE with the initial node embeddings as the input to estimate the weight matrix.* (2) *A Graph Neural ODE with initial node embeddings and weight matrix to estimate the final embedding.* The first component is to control the ODE that ensures each node will reach an optimal embedding. We combine these two parts into one single ODE function. The overall formula can be written as follows:

$$\begin{aligned} E^u_* &= E^u(0) + \int_0^{t_1} \sigma(f(E^u(t),\theta))(A^n - I)E^i(t)dt \\ E^i_* &= E^i(0) + \int_0^{t_1} \sigma(f(E^i(t),\theta))(A^n - I)E^u(t)dt \end{aligned} \tag{3}$$

where E^u_* is the final users embeddings and E^i_* is the final items embeddings. $f(E^u(t),\theta)$ and $f(E^i(t),\theta)$ represent the MLPs with the user embedding and item embedding at time step t, respectively. Here, θ represents the parameters of the MLPs. Additionally, σ denotes the sigmoid function. The output of the MLPs is the weight matrix at time step t. The final user embeddings and item embeddings are denoted as E^u_* and E^i_*, respectively. Similar to GODE-CF, we employ the BPR loss for training the embeddings and ODE solvers, such as Euler or RK4, to solve the ODE. To make predictions, we follow the same settings as GODE-CF. Once we obtain the final embeddings, the prediction is calculated as the inner product of the user embeddings and item embeddings: $y_{u,i} = {E^u_*}^T E^i_*$.

Table 2: The statistics of the datasets

Datasets	Training	Validation	Testing	Sparsity
Office	43,448	4,905	4,905	0.44867%
Health	269,137	38,609	38,609	0.0484%
Cell Phone	138,681	27,879	27,879	0.0668%
Beauty	153,776	22,363	22,363	0.07335%

4 Experiment

4.0.1 ***Datasets.*** We use the public Amazon Reviews dataset [9] with four benchmark categories, including: *Beauty*, *Health*, *Cell Phones*, *Office Product*. The details of the datasets are summarized in Table 2. We follow the 5-core setting as existing works on users and the same transformation [4–7] of treating the existence of reviews as positives. We sort each user's interactions chronologically and adopt the leave-one-out setting, with the last interacted item for testing and the second last interaction for validation.

4.0.2 ***Baselines.*** In total, we compare CDE-CF with various types of the state-of-the-art models:

- layerGCN [17] is a GCN-based CF method with layer-refinement.
- LightGCN [6] is a lightweight linear GCN-based CF method.
- UltraGCN [8] is an ultra-simplified formulation of GCN that directly approximates the limit of infinite message-passing layers.
- GTN [4] is a graph trend filtering network framework to capture the adaptive reliability of the interactions.
- LT-OCF [2] is a NODE-based method that aims to learn the optimal architecture of the model for graph-based CF.
- GODE-CF [12] is a GODE-based method that uses two GCN layers of information to estimate the final embeddings.

4.0.3 ***Evaluation Metrics.*** For the evaluation metrics, Recall@K and NDCG@K are adopted for a fair comparison of all the baselines in the top-K recommendation task. K is set as 20 in the main performance evaluation and is set to 20 by default in the other experiments. The full-ranking strategy is adopted for all the experimental studies, i.e., all the candidate items not interacted with the user will be ranked in testing.

4.1 Overall Performance Comparison

In this comprehensive experimental study, we evaluated the performance of several state-of-the-art GCN-based methods and ODE-based methods on four diverse datasets. We use Recall@20 and NDCG@20 as evaluation metrics to measure the performance of the models. Table 1 is the overall performance, and we summarize the main results:

- Predominantly, CDE-CF achieved the highest NDCG@20 and Recall@20 scores across all datasets, highlighting its superior efficacy in recommendation tasks. Among ODE-based methods, GODE-CF significantly outperforms the strongest GCN-based baselines. For other baselines, LightGCN exhibits the best performance on *Beauty*, while GTN demonstrates the best performance on *Health* compared to other GCN-based baselines.
- Among all baselines, the ODE-based model GODE-CF shows state-of-the-art performance for all cases. This indicates the superiority of the ODE-based methods for modeling the high-order relationship in the graph convolution.

Table 3: Impact of Weight and Discrete Weight on Beauty, Health, Cell Phone, and Office Datasets.

Dataset	Beauty		Health		Cell Phone		Office	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
Without weight (W)	0.08076	0.03407	0.03403	0.01353	0.05058	0.01916	0.05586	0.02661
With discrete weight (W)	0.08027	0.03395	0.03362	0.01356	0.05079	0.01909	0.05668	0.02702
CDE-CF	**0.08129**	**0.03426**	**0.03468**	**0.01375**	**0.05082**	**0.01945**	**0.05728**	**0.02713**

Table 4: Efficiency comparison with LightGCN, LT-OCF, and GODE-CF on four datasets, with 1000 total training epochs.

Training Time				
Dataset	Beauty	Health	Cell Phone	Office
LightGCN	2393.25s	4892.29s	2192.09s	**382.89s**
LT-OCF	5876.53s	15785.80s	5614.14s	771.50s
GODE-CF	2120.92s	4823.32s	2136.18s	406.28s
CDE-CF	**2109.52s**	**4767.27s**	**2106.83s**	421.46s

5 Ablation Study

Impact on weight. Here, we present the comprehensive ablation study of weight components of the CDE-CF model. From the detailed analysis provided in Table 3, we observe interesting insights. Firstly, when incorporating a discrete weight, GODE-CF does not consistently outperform the version without weight. This indicates that a discrete weight alone may not be sufficient to effectively control the ODE system. In comparison, our method CDE-CF, which incorporates continuous weights, surpasses GODE-CF without weight in all cases. This highlights the critical role of continuous weight control in improving the performance of ODE methods.

Efficiency Comparison. We provide empirical evidence demonstrating the superiority of CDE-CF in terms of training efficiency compared to other baselines. We train all models with a fixed number of 1000 epochs to eliminate the effect of varying epoch numbers. As in Table 4, though CDE-CF includes an additional component compared to GODE-CF, and CDE-CF has a faster training time than GODE-CF in three datasets. The time step t is the main factor that affects the training speed. We have found that the optimal value for t across all cases is approximately 8.5 for GODE-CF. For our method, the optimal t value is around 6.5, resulting in a reduced amount of time required to solve the ODE.

6 Conclusion

In this study, we propose a new method called CDE-CF based on the GODE-CF. In particular, a novel control weight is devised to cater to the continuous time in the ODE functions in order to better model the graph convolution process. The experimental results on four different real-world datasets demonstrate that CDE-CF outperforms various state-of-the-art baselines in terms of performance, while also having a shorter training time compared to GODE-CF. Additionally, the ablation study reveals that we have created a more reasonable weight matrix control compared to GODE-CF. For further study, we will explore more complex controllers for ODE-based methods. In conclusion, having a good controller is crucial to further enhance the performance of the ODE-based method and contribute to performance improvements.

7 Acknowledgment

This work is supported in part by NSF under grants III-2106758, and POSE-2346158.

References

[1] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2018. Neural Ordinary Differential Equations. *Advances in Neural Information Processing Systems* (2018), 6571–6583.

[2] Jeongwhan Choi, Jinsung Jeon, and Noseong Park. 2021. LT-OCF: Learnable-time ode-based collaborative filtering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management.* 251–260.

[3] Zhiwei Deng, Megha Nawhal, Lili Meng, and Greg Mori. 2019. Continuous graph flow. *arXiv preprint arXiv:1908.02436* (2019).

[4] Wenqi Fan, Xiaorui Liu, Wei Jin, Xiangyu Zhao, Jiliang Tang, and Qing Li. 2022. Graph trend filtering networks for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval.* 112–121.

[5] Ziwei Fan, Ke Xu, Zhang Dong, Hao Peng, Jiawei Zhang, and Philip S Yu. 2023. Graph collaborative signals denoising and augmentation for recommendation. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval.* 2037–2041.

[6] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval.* 639–648.

[7] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web.* 173–182.

[8] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: ultra simplification of graph convolutional networks for recommendation. In *Proceedings of the 30th ACM international conference on information & knowledge management.* 1253–1262.

[9] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval.* 43–52.

[10] Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. 2019. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532* (2019).

[11] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[12] Ke Xu, Yuanjie Zhu, Weizhi Zhang, and S Yu Philip. 2023. Graph Neural Ordinary Differential Equations-based method for Collaborative Filtering. In *2023 IEEE International Conference on Data Mining (ICDM).* IEEE, 1445–1450.

[13] Weizhi Zhang, Liangwei Yang, Yuwei Cao, Ke Xu, Yuanjie Zhu, and S Yu Philip. 2023. Dual-Teacher Knowledge Distillation for Strict Cold-Start Recommendation. In *2023 IEEE International Conference on Big Data (BigData).* IEEE, 483–492.

[14] Weizhi Zhang, Liangwei Yang, Zihe Song, Henry Peng Zou, Ke Xu, Liancheng Fang, and Philip S Yu. 2024. Do We Really Need Graph Convolution During Training? Light Post-Training Graph-ODE for Efficient Recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management.* 3248–3258.

[15] Weizhi Zhang, Liangwei Yang, Zihe Song, Henry Peng Zou, Ke Xu, Yuanjie Zhu, and Philip S Yu. 2024. Mixed Supervised Graph Contrastive Learning for Recommendation. *arXiv preprint arXiv:2404.15954* (2024).

[16] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI open* 1 (2020), 57–81.

[17] Xin Zhou, Donghui Lin, Yong Liu, and Chunyan Miao. 2023. Layer-refined graph convolutional networks for recommendation. In *2023 IEEE 39th international conference on data engineering (ICDE).* IEEE, 1247–1259.