Eco-Aware Graph Neural Networks for Sustainable Recommendations

Antonio Purificato ⓘ and Fabrizio Silvestri ⓘ

Sapienza University of Rome, Rome, Italy
`{purificato, fsilvestri}@diag.uniroma1.it`

Abstract. Recommender systems play a crucial role in alleviating information overload by providing personalized recommendations tailored to users' preferences and interests. Recently, Graph Neural Networks (GNNs) have emerged as a promising approach for recommender systems, leveraging their ability to effectively capture complex relationships and dependencies between users and items by representing them as nodes in a graph structure.

In this study, we investigate the environmental impact of GNN-based recommender systems, an aspect that has been largely overlooked in the literature. Specifically, we conduct a comprehensive analysis of the carbon emissions associated with training and deploying GNN models for recommendation tasks. We evaluate the energy consumption and carbon footprint of different GNN architectures and configurations, considering factors such as model complexity, training duration, hardware specifications and embedding size.

By addressing the environmental impact of resource-intensive algorithms in recommender systems, this study contributes to the ongoing efforts towards sustainable and responsible artificial intelligence, promoting the development of eco-friendly recommendation technologies that balance performance and environmental considerations. Code is available at:
`https://github.com/antoniopurificato/gnn_recommendation_and_environment`.

Keywords: Graph Neural Networks · Recommendation Systems · Environmental Impact.

1 Introduction

Recommender systems (RSs) play a crucial role in mitigating information overload by providing personalized recommendations, benefiting users and service providers across various platforms, including e-commerce (e.g., Tmall, Amazon) (Ge et al., 2020) and social networks (e.g., Gowalla, Facebook) (Peng et al., 2020; Zhao et al., 2016). Different approaches have been proposed, ranging from collaborative filtering techniques that leverage user-item interactions to content-based methods that analyze item features (Chen et al., 2018) to sequential recommendation, aiming to capture the sequential patterns in user behavior and provide recommendations based on the current context or session (Betello et al., 2024b; Bacciu et al., 2023).

In addition to sequential recommendation, Graph Neural Networks (GNNs) have emerged as a promising approach for recommender systems (Liu et al., 2024; Mancino et al., 2023). GNNs can effectively capture the complex relationships and dependencies between users and items by representing them as nodes in a graph structure. By propagating and aggregating information along the edges of the graph, GNNs can learn rich representations that encode high-order connectivity patterns, leading to improved recommendation performance (Wu et al., 2022). Furthermore, GNNs can naturally incorporate various types of auxiliary information, such as user profiles, item attributes, and social connections, into the graph structure, enabling the exploitation of heterogeneous data sources for more accurate recommendations (Purificato et al., 2024). Nowadays, resource-intensive algorithms have become prevalent in modern recommender systems, resulting in higher energy consumption for recommendation experiments (Betello et al., 2024a). However, despite some studies having been carried out in regarding the environmental impact of SRSs (Betello et al., 2024a; Spillo et al., 2023), only one work presented an analysis of the computational consumption of GNN-based RSs Spillo et al. (2023).

In this work, we analyse the environmental impact of GNN-based RSs experiments by faithfully replicating representative experimental pipelines. Through a comprehensive comparative analysis, we shed light on the carbon emissions attributable to the training and deployment of GNN-based RSs. Our study serves as a clarion call for sustainability, underscoring the need to reconcile the pursuit of technological advancements with environmental consciousness within the realm of RSs. This study aims to answer to the following research questions:

- **RQ1**: Which model performs the best, and what are the trade-offs in terms of resource consumption?
- **RQ2**: How does the embedding size of the GNN affect the environmental impact of the results?

2   Related Work

In the Glasgow Agreement (Hunter et al., 2021), participating nations committed to reducing CO_2 emissions, underscoring the urgent need for environmental action. This commitment is particularly relevant to our field, as the environmental impact of GPU training in machine learning is significant. The energy consumption associated with these computational processes contributes significantly to CO_2 emissions, exacerbating climate change (Patterson et al., 2021). It is therefore our responsibility to raise awareness of this issue.

Although the environmental impact of deep learning algorithms has been investigated in certain domains, such as Natural Language Processing (Bender et al., 2021; Wang et al., 2023) and Information Retrieval (Scells et al., 2022), there is a dearth of research examining the environmental footprint of Recommender Systems.

Spillo et al. (2023) benchmark several state-of-the-art recommendation algorithms in terms of both recommendation performance and carbon emissions and

analyze the trade-off between energy consumption, carbon emissions and the predictive accuracy of recommendation algorithms. A difference with respect to our approach is that they do not study the impact of some important factors on the performance and on the emissions, such as the embedding size of each model.

Betello et al. (2024a) provide a code resource and a robust framework for developing RSs and establishing a foundation for consistent and reproducible experimentation. They also study how the number of parameters influences the CO_2 consumption of the proposed algorithms. Differently from our work, they do not consider GNN-based approaches, but only sequential recommendation algorithms. While their research aims to provide a reproducibility analysis with an associated framework, the objective of our work is to highlight the significance of implementing environmentally sustainable solutions for recommendation tasks.

In the next Section we present our methodological approach to assess the environmental impact of recommender systems.

3 Method

3.1 Calculating CO_2 Emissions

In this study, we use CodeCarbon[1] (Courty et al., 2023), a tool designed to track the power consumption of both CPUs and GPUs. This allows us to measure carbon dioxide equivalent (CO_2-eq), a widely accepted standard used by numerous organisations and governments to monitor emissions of various greenhouse gases (Kim and Neff, 2009). CO_2-eq facilitates the comparison of greenhouse gas emissions by converting quantities of different gases into an equivalent amount of CO_2, based on their respective global warming potentials. By using CodeCarbon, we can accurately assess the environmental impact of our training processes, in line with our commitment to sustainability and responsible research practices in machine learning.

Our decision to focus specifically on CO_2-eq emissions is motivated by several factors. First, CO_2-eq is a widely recognized and accepted metric for quantifying the combined impact of various greenhouse gases on global warming, allowing us to provide a standardized and comparable measure of the overall environmental impact Betello et al. (2024a); Spillo et al. (2023). Second, CO_2 emissions are particularly relevant in the context of energy-intensive machine learning training processes, which typically rely on electricity generated from fossil fuel sources. Tracking CO_2-eq emissions allows us to directly assess the carbon footprint associated with the energy consumption of our experiments.

3.2 Models

The Neural Graph Collaborative Filtering (NGCF) model (Wang et al., 2019) represents user-item interactions as a bipartite graph and learns user and item

---

[1] https://codecarbon.io

embeddings by propagating them on the graph through message passing layers. The key idea is to capture high-order connectivity patterns by recursively propagating embeddings over the graph using graph convolutional layers. NGCF is trained end-to-end to minimize the difference between predicted and actual user-item interactions. It exploits high-order connectivity, learning non-linear relationships, and providing inductive capabilities for new users or items.

LightGCN (He et al., 2020) learns user and item embeddings by propagating them on the user-item interaction graph through a series of graph convolutional layers. Unlike NGCF, LightGCN removes the feature transformation and non-linear activation layers, making it a linear model. The key idea is to simplify the GCN architecture while preserving the ability to capture high-order connectivity patterns between users and items. LightGCN is trained end-to-end to optimize the BCE between the predicted and actual user-item interactions. It has demonstrated competitive performance while being computationally efficient.

SimGCL (Yu et al., 2022) add random uniform noise to hidden representations for augmentations, resulting in more uniform node representations that mitigate the popularity bias. Adjusting the noise magnitude can improve regulation of representation uniformity, leading to advantages on recommendation accuracy and model training efficiency.

In LightGCL (Cai et al., 2023), the graph augmentation is guided by singular value decomposition (SVD) to not only distill the useful information of user-item interactions but also inject the global collaborative context into the representation alignment of contrastive learning. Instead of generating two handcrafted augmented views, important semantic of user-item interactions can be preserved with their paradigm. This enables self-augmented representations to be reflective of both user-specific preferences and cross-user global dependencies.

3.3   Experimental Pipeline

To evaluate the impact of different embedding sizes on the model's performance and computational requirements, experiments with embedding sizes of 32, 64, 128, and 256 are conducted. The results of these experiments are presented in Section 5. Prior to commencing the training process, we initialize a Code-Carbon tracker to monitor the carbon emissions associated with the training process. Additionally, we utilize the DeepSpeed[2] library to compute the number of floating-point operations (FLOPs) required for each model configuration.

During the training process, the CodeCarbon library is used to log the power consumption every 30 seconds, allowing us to track the energy consumption in real-time. Upon completion of the training, we compute the total carbon emissions and various performance metrics, as described in Section 4.

In the next Section we will presents the experimental setup and describe the different metrics and datasets used in the experiments.

---

[2] https://www.deepspeed.ai

4 Experiments

4.1 Datasets

Our analyses encompass a collection of datasets, ensuring comprehensive and robust insights. By incorporating datasets with diverse characteristics, such as varying user and item counts, we aim to unravel the intricate interplay between these factors and our findings. This approach enables us to capture a view of real-world scenarios, thereby fortifying the applicability of our conclusions across a broad spectrum of contexts. All the statistics of these datasets are presented in Table 1.

- MovieLens[3]: The MovieLens dataset (Harper and Konstan, 2015) is widely recognized as a benchmark for evaluating recommendation algorithms. We utilize MovieLens 1M (ML-1M).
- Amazon: These datasets consist of product reviews collected from Amazon.com (McAuley et al., 2015). The data are organized into distinct datasets based on Amazon's primary product categories. For our study, we focus on the "Beauty" category (Beauty).
- DianPing: This dataset contains the user reviews as well as the detailed business meta data information crawled from a famous Chinese online review website[4].

Our data preprocessing pipeline adheres to well-established practices in the field. We adopt an implicit approach, treating all interactions as binary events without considering rating values, as done in (Kang and McAuley, 2018; Sun et al., 2019).

For dataset partitioning, we employ a widely-used strategy in sequential recommendation tasks (Sun et al., 2019; Kang and McAuley, 2018). The most recent interaction for each user is held out for testing, while the second-to-last interaction is reserved for validation. The remaining interactions constitute the training set, providing a chronological sequence of user behavior.

Table 1: Dataset statistics after pre-processing. Density and sparsity are percentage values.

Dataset name	Users	Items	Interactions	Density	Sparsity
Amazon Beauty	1,210,271	249,274	2,023,070	0.001	99.999
MovieLens 1M	6,040	3,952	999,611	4.189	95.810
DianPing	542,706	243,247	4,422,473	0.003	99.997

---

[3] https://grouplens.org/datasets/movielens
[4] DianPing.com

4.2   Metrics

To evaluate the performance of sequential recommendation algorithms, we employed four widely adopted metrics commonly used in Information Retrieval (IR) (Kang and McAuley, 2018; Purificato et al., 2024): Precision, Recall, Normalized Discounted Cumulative Gain (NDCG), and Hit Ratio (HIT). These metrics provide a comprehensive assessment of the recommendation system's ability to identify relevant items and rank them effectively.

- Precision: This metric calculates the proportion of correctly identified relevant items among the recommended items. It measures the system's ability to avoid irrelevant recommendations.
- Recall: It quantifies the fraction of correctly identified relevant items among the recommendations relative to the total number of relevant items in the dataset. This metric evaluates the system's capability to retrieve as many relevant items as possible.
- Normalized Discounted Cumulative Gain (NDCG): This metric evaluates the performance of a ranking system by considering the position of relevant items in the ranked list. It assigns higher scores to relevant items ranked higher, as they are typically where a user's attention is focused. NDCG captures the importance of ranking relevant items at the top of the recommendation list.
- Hit Ratio (HIT): Is a key metric in recommendation systems that measures whether relevant items appear within the top K positions of a model's recommendation list. For each user, if at least one relevant item is included in the top K recommendations, it counts as a "hit." The HIT@K score is then calculated as the proportion of users for whom the model successfully includes at least one relevant item within the top K.
- Emissions: Represents the CO_2-eq (measured in Kg) required for training a single model and is the sum of the single CO_2-eq emissions over each epoch.

By employing these four metrics, we can comprehensively assess the recommendation system's ability to identify relevant items, rank them effectively, and provide high-quality recommendations tailored to the user's preferences and interests.

4.3   Reproducibility

In order to facilitate a rigorous and unbiased comparison, a standardized experimental setup was adopted for all models. The training regime consisted of 400 epochs, with batch sizes of 2048 and 4096 for the training and validation stages, respectively. The optimization was carried out using the Adam algorithm, with a learning rate fixed at 0.001. Furthermore, to mitigate the effects of random initialization and promote reproducibility, an identical seed was employed across all experiments. In order to see how all the experiments evolve over time, no early stopping procedures were applied.

4.4   Hardware

All experiments were performed on a single NVIDIA RTX A6000 with 10752 CUDA cores and 48 GB of RAM. The code is written in Python 3 and to train all the models it was used the RecBole library[5] (Xu et al., 2023).

In the next Section we will present the results of the proposed study, in terms of performance metrics and environmental impact.

5   Results

### 5.1   RQ1: Which model excels, and at what cost?

As shown in Table 2, LightGCN outperforms all competitors across all datasets, with the most significant lead observed on the Beauty dataset, while the gap narrows on the ML-1M dataset. LightGCN's superior performance can be attributed to two key factors: firstly, it is an advancement over NGCF, with its advantages clearly demonstrated in prior experiments (He et al., 2020). Secondly, both LightGCL and SimGCL, although promising, involve higher computational complexity, and 400 epochs may not suffice for them to converge to optimal results. In terms of carbon emissions, NGCF remains the most efficient on two out of the three datasets—a somewhat unexpected outcome, given that LightGCN is touted by its authors as being more lightweight than NGCF. The best trade-off in terms of performance-emission will probably remain LightGCN, which is the second-best model in terms of environmental impact on two of the three datasets.

| Dataset	Model	P@10	R@10	NDCG@10	HIT@10	P@100	R@100	NDCG@100	HIT@100	Emissions
Beauty	LightGCL	.0019	.0185	.0107	.0187	.0005	.0517	.0190	.0591	12.5521
	LightGCN	**.0022**	**.0212**	**.0121**	**.0217**	**.0006**	**.0583**	**.0195**	**.0594**	4.1341
	NGCF	.0017	.0160	.0087	.0164	.0005	.0508	.0156	.0519	**1.1147**
	SimGCL	.0004	.0039	.0019	.0039	.0002	.0227	.0054	.0230	12.9475
DianPing	LightGCL	.0044	.0239	.0221	.0499	.0018	.1673	.0393	.1734	15.0111
	LightGCN	**.0053**	**.0417**	**.0223**	**.0518**	**.0023**	**.1680**	**.0482**	**.2031**	**13.4835**
	NGCF	.0032	.0235	.0123	.0313	.0017	.1205	.0320	.1523	27.3234
	SimGCL	.0040	.0238	.0101	.0474	.0016	.1435	.0347	.1600	21.2422
ML-1M	LightGCL	.1986	.1591	.2546	.7341	.0761	.5231	.3366	.9538	0.1185
	LightGCN	**.2094**	**.1731**	**.2681**	**.7626**	**.0808**	**.5692**	**.3619**	**.9664**	0.2146
	NGCF	.2001	.1623	.2549	.7373	.0786	.5492	.3470	.9621	**0.1057**
	SimGCL	.1073	.1188	.1490	.6210	.0438	.4037	.2333	.9386	0.1281

Table 2: Results of the models in terms of Precision@K (P@K), Recall@K (R@K), NDCG@K and HIT@K, with K $\in \{\mathbf{10}, \mathbf{100}\}$ (↑ is better). Also the emissions in terms of CO_2-eq [kg] are shown (↓ is better). **Bold** denotes the best model for a dataset by the metric in the main group, underlined the second best.
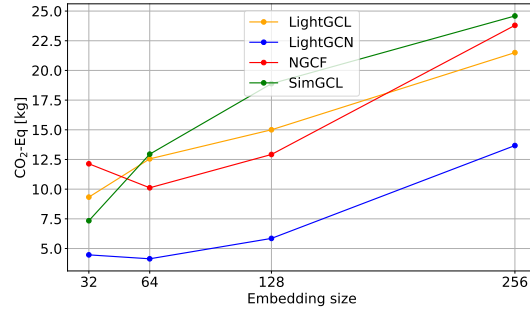
(a) Beauty
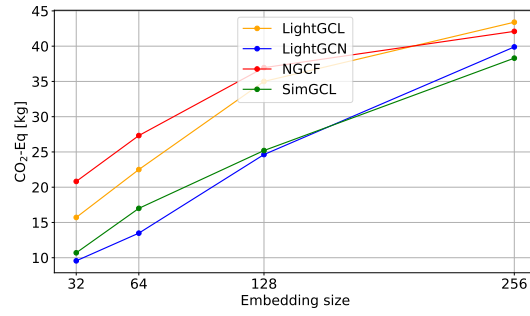
(b) DianPing
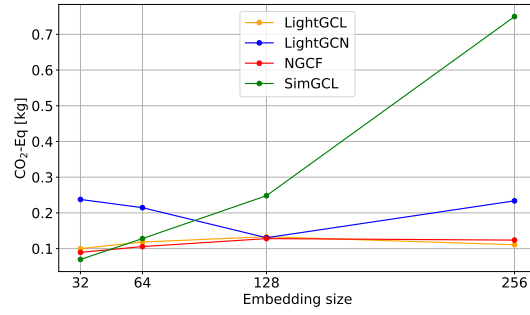
(c) ML-1M

Fig. 1: Difference in terms of environmental impact when changing the embedding size of the GNNs. On the y-axis it can be seen the CO_2-eq while on the x-axis the embedding size.

5.2   Embedding Size and Environmental Impact

Figure 1 shows that increasing the size of the embeddings generally leads to a higher environmental impact across all models and datasets. This trend is particularly pronounced in the DianPing dataset, which contains a large number of interactions. Interestingly, on the DianPing dataset, NGCF has a higher cost than SimGCL, despite SimGCL being computationally more expensive. However, on the other datasets, SimGCL consistently remains the most computationally demanding model. The same Figure also illustrates how the dataset size affects CO_2-eq costs. On the ML-1M dataset, the costs are significantly lower compared to those on the DianPing dataset. This not only depends from the number of users and the number of items, but also from the interactions between users and items. On the DianPing and Beauty datasets, as the embedding size increases, emissions also increase, as expected. It is interesting to note that on the ML-1M dataset, for the NGCF and LightGCL models, emissions related to an embedding size of 256 are higher compared to those with an embedding size of 128. This result requires more detailed investigation in future work, particularly on how each parameter of a model influences the environmental impact of the respective model.

6   Conclusions

In this study, we examined the environmental impact of GNN-based recommender systems, an aspect often overlooked in AI research. Our analysis of carbon emissions and energy consumption across different GNN architectures and configurations highlights how model complexity, training duration, hardware specifications, and embedding size affect their environmental footprint.

While GNNs provide significant benefits in capturing complex relationships for recommendation tasks, our findings show that these gains can come with considerable environmental costs, especially when large datasets or extensive embeddings are used. By emphasizing these trade-offs, our study contributes to the discourse on sustainable AI, encouraging the integration of environmental considerations into the development of recommender systems. Future research should focus on optimizing GNN architectures to balance performance with sustainability, exploring new algorithms and energy-efficient methods.

We hope this work inspires further efforts to develop eco-friendly AI technologies that align with global sustainability goals.

Acknowledgment

---

[5] `https://recbole.io/`

Bibliography

Andrea Bacciu, Federico Siciliano, Nicola Tonellotto, and Fabrizio Silvestri. 2023. Integrating item relevance in training loss for sequential recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 1114–1119.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (Virtual Event, Canada) *(FAccT '21)*. Association for Computing Machinery, New York, NY, USA, 610–623. `https://doi.org/10.1145/3442188.3445922`

Filippo Betello, Antonio Purificato, Federico Siciliano, Giovanni Trappolini, Andrea Bacciu, Nicola Tonellotto, and Fabrizio Silvestri. 2024a. A Reproducible Analysis of Sequential Recommender Systems. *arXiv preprint arXiv:2408.03873* (2024).

Filippo Betello, Federico Siciliano, Pushkar Mishra, and Fabrizio Silvestri. 2024b. Investigating the Robustness of Sequential Recommender Systems Against Training Data Perturbations. In *Advances in Information Retrieval*. Springer Nature Switzerland, Cham, 205–220.

Xuheng Cai, Chao Huang, Lianghao Xia, and Xubin Ren. 2023. LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation. In *The Eleventh International Conference on Learning Representations*.

Rui Chen, Qingyi Hua, Yan-Shuo Chang, Bo Wang, Lei Zhang, and Xiangjie Kong. 2018. A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks. *IEEE Access* 6 (2018), 64301–64320. `https://doi.org/10.1109/ACCESS.2018.2877208`

Benoit Courty, Victor Schmidt, Goyal-Kamal, MarionCoutarel, Boris Feld, Jérémy Lecourt, SabAmine, kngoyal, Mathilde Léval, Alexis Cruveiller, inimaz, ouminasara, Franklin Zhao, Aditya Joshi, Alexis Bogroff, Amine Saboni, Hugues de Lavoreille, Niko Laskaris, Luis Blanche, Edoardo Abati, LiamConnell, Douglas Blank, Ziyao Wang, Armin Catovic, Michał Stęchły, alencon, JPW, MinervaBooks, Necmettin Çarkacı, and DomAlexRod. 2023. mlco2/codecarbon: v2.3.2. `https://doi.org/10.5281/zenodo.10213072`

Yingqiang Ge, Shuya Zhao, Honglu Zhou, Changhua Pei, Fei Sun, Wenwu Ou, and Yongfeng Zhang. 2020. Understanding Echo Chambers in E-commerce Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 2261–2270. `https://doi.org/10.1145/3397271.3401431`

F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (dec 2015), 19 pages. `https://doi.org/10.1145/2827872`

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* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 639–648. `https://doi.org/10.1145/3397271.3401063`

David B Hunter, James E Salzman, and Durwood Zaelke. 2021. Glasgow climate summit: Cop26.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

Brent Kim and Roni Neff. 2009. Measurement and communication of greenhouse gas emissions from US food consumption via carbon calculators. *Ecological Economics* 69, 1 (2009), 186–196.

Yuxi Liu, Lianghao Xia, and Chao Huang. 2024. SelfGNN: Self-Supervised Graph Neural Networks for Sequential Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 1609–1618. `https://doi.org/10.1145/3626772.3657716`

Alberto Carlo Maria Mancino, Antonio Ferrara, Salvatore Bufi, Daniele Malitesta, Tommaso Di Noia, and Eugenio Di Sciascio. 2023. KGTORe: Tailored Recommendations through Knowledge-aware GNN Models. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 576–587. `https://doi.org/10.1145/3604915.3608804`

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* (Santiago, Chile) *(SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 43–52. `https://doi.org/10.1145/2766462.2767755`

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training.

Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph Representation Learning via Graphical Mutual Information Maximization. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 259–270. `https://doi.org/10.1145/3366423.3380112`

Antonio Purificato, Giulia Cassarà, Federico Siciliano, Pietro Liò, and Fabrizio Silvestri. 2024. Sheaf4Rec: Sheaf Neural Networks for Graph-based Recommender Systems. arXiv:2304.09097 [cs.IR] `https://arxiv.org/abs/2304.09097`

Harrisen Scells, Shengyao Zhuang, and Guido Zuccon. 2022. Reduce, Reuse, Recycle: Green Information Retrieval Research. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 2825–2837. `https://doi.org/10.1145/3477495.3531766`

Giuseppe Spillo, Allegra De Filippo, Cataldo Musto, Michela Milano, and Giovanni Semeraro. 2023. Towards Sustainability-aware Recommender Systems: Analyzing the Trade-off Between Algorithms Performance and Carbon Footprint. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 856–862. `https://doi.org/10.1145/3604915.3608840`

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. arXiv:1904.06690 [cs.IR]

Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 165–174. `https://doi.org/10.1145/3331184.3331267`

Xiaorong Wang, Clara Na, Emma Strubell, Sorelle Friedler, and Sasha Luccioni. 2023. Energy and Carbon Considerations of Fine-Tuning BERT. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph Neural Networks in Recommender Systems: A Survey. *ACM Comput. Surv.* 55, 5, Article 97 (dec 2022), 37 pages. `https://doi.org/10.1145/3535101`

Lanling Xu, Zhen Tian, Gaowei Zhang, Junjie Zhang, Lei Wang, Bowen Zheng, Yifan Li, Jiakai Tang, Zeyu Zhang, Yupeng Hou, Xingyu Pan, Wayne Xin Zhao, Xu Chen, and Ji-Rong Wen. 2023. Towards a More User-Friendly and Easy-to-Use Benchmark Library for Recommender Systems. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan) *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 2837–2847. `https://doi.org/10.1145/3539618.3591889`

Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 1294–1303.

Zhou Zhao, Hanqing Lu, Deng Cai, Xiaofei He, and Yueting Zhuang. 2016. User Preference Learning for Online Social Recommendation. *IEEE Trans. on Knowl. and Data Eng.* 28, 9 (sep 2016), 2522–2534. `https://doi.org/10.1109/TKDE.2016.2569096`