Sustainable Computing: Informatics and Systems 49 (2026) 101286

Contents lists available at ScienceDirect

Sustainable Computing: Informatics and Systems

journal homepage: www.elsevier.com/locate/suscom

# Balancing carbon footprint and algorithm performance in recommender systems: A comprehensive benchmark

Giuseppe Spillo [a], Allegra De Filippo [b,*], Cataldo Musto [a], Michela Milano [b], Giovanni Semeraro [a]

[a] University of Bari Aldo Moro, Via Edoardo Orabona 4, Bari, 70125, Italy
[b] University of Bologna, Viale Risorgimento 2, Bologna, 40136, Italy

ARTICLE INFO

*Keywords:*
Recommender systems
Evaluation
Sustainability
Carbon footprint

ABSTRACT

In this paper, we present a reproducible pipeline to benchmark the trade-off between carbon emissions and recommendation performance across 14 algorithms and three publicly available datasets. In particular, we contribute: *(a)* a standardized protocol to account for carbon emissions of recommendation algorithms; *(b)* an empirical quantification of the carbon cost of hyperparameter tuning, and *(c)* an evaluation of data-reduction strategies as a low-cost approach to reduce emissions while improving certain non-accuracy metrics. Unlike previous literature, which mainly focused on the trade-off between performance and emissions, our benchmark reveals the cost of hyperparameter tuning. It examines the impact of data reduction techniques on the path toward sustainability-aware recommender systems. Our results show that simpler algorithms often deliver competitive accuracy at significantly lower emissions, and that exhaustive tuning can dramatically increase carbon costs with limited accuracy gains. Generally speaking, this study aims to discuss the challenges of energy consumption in recommender systems and to develop a new generation of algorithms that prioritize sustainability. All code and experiment traces are publicly released for reproducibility on Github.[1]

## 1. Background and motivations

In recent years, Artificial Intelligence (AI) algorithms have emerged as transformative tools, revolutionizing industries ranging from healthcare [1,2], finance [3–5], and the food industry [6]. However, this unprecedented progress has come at a high cost, *i.e.*, the escalating energy consumption associated with AI systems [7]. As these algorithms become increasingly sophisticated, their demand for computational resources has grown exponentially, leading to a surge in power consumption and environmental concerns [8–10].

The exponential rise in energy consumption by AI algorithms is particularly relevant. Indeed, as stated in [11,12], it affects the concentration of greenhouse gases (GHGs) in the atmosphere, which, in turn, significantly contributes to climate change. For example, Strubell et al. [13] found that training recent translation engines once can emit nearly 300 tonnes of $CO_2$. Based on some estimation,[2] this is equivalent to the yearly energy use of more than 40 houses. Accordingly, it is urgent to establish benchmarks for measuring and mitigating the environmental impact of these algorithms [14], as emphasized by contributions in the area of *Green AI* [15]. However, despite the interest toward designing and implementing AI algorithms that are more energy-efficient, [16–18], the environmental impact of complex algorithms, such as *recommender systems* (RSs) [19], and the trade-off between sustainability and their overall performance, is generally overlooked. This is a relevant research gap, as RSs are a data-intensive technology [20] that requires a substantial amount of training data and considerable computation time to learn an effective model. Moreover, the problem is further exacerbated because RSs must be continuously retrained whenever a user provides new information about their preferences and tastes.

Accordingly, this paper contributes to research on RSs and sustainability [21], aiming to shed light on the problem of carbon emissions and energy consumption associated with RSs. Indeed, this article introduces a methodology for conducting an empirical benchmarking of several representative recommendation algorithms by analyzing the trade-off between their performance and carbon footprint. The methodological framework presented in this work aligns with the principles of some of the Sustainable Development Goals [22]. Indeed,

* Corresponding author.
*E-mail address:* allegra.defilippo@unibo.it (A. De Filippo).

[1] https://github.com/swapUniba/RecSysCarbonFootprint
[2] https://www.epa.gov/energy/greenhouse-gas-equivalencies-calculator

https://doi.org/10.1016/j.suscom.2025.101286
Received 18 April 2025; Received in revised form 9 October 2025; Accepted 18 December 2025
Available online 24 December 2025

benchmarking the carbon footprint of recommendation algorithms is closely aligned with the objectives of SDG7 and SDG13, which address clean energy and climate action, respectively.[3]

Generally speaking, the primary outcome of this work is that the choice of the optimal recommendation algorithm requires a thorough analysis, as less sophisticated algorithms remain a viable alternative for *greener* recommendations, especially when parameter tuning is necessary. Finally, the manuscript also demonstrated that data reduction can increase diversity and reduce popularity bias in recommendations [23], while maintaining low emissions. To conclude, this paper provides the following contributions:

1. It defines a reproducible and replicable experimental protocol to analyze carbon emissions of recommendation algorithms;
2. It analyzes the trade-off between carbon emissions and the predictive accuracy, by also investigating the impact of hyper-parameter tuning on the overall carbon emissions of the algorithms;
3. It studies the effect of data reduction strategies as a solution to reduce the carbon emission of recommendation algorithms;
4. It sketches the take-home messages and the main limitations of this work. This aims to foster research in the area and drive the development of a new generation of recommendation algorithms that also consider sustainability.

In the following, the structure of the article is presented: Section 2 introduces the related works, while Section 3 introduces an overview of RSs; next, Section 4 introduces the strategy exploited to track and monitor carbon emission information; Section 5 introduces the experimental design and a deep discussion of the results, while Section 6 presents useful take-home messages; finally, Section 7 summarizes our findings, and sketches future works.

## 2. Related work

The dichotomy between *Red* and *Green* AI was first introduced by Schwartz in 2020 [14]. Following this work, research in this area has rapidly expanded, as also demonstrated by the recent review on Green AI [24], which compared several studies quantifying the energy consumption of AI models.

One of the first attempts in this direction is due by Strubell et al. [13], who estimated the emissions of training and fine-tuning a large Transformer model. The recent proliferation of large language models has paved the way for numerous studies aimed at assessing both the effectiveness and the emissions of such algorithms [25]. Other extensive benchmarks are presented in [26,27], where the authors provide a survey of the carbon emissions associated with machine learning models. Finally, in [28], the authors quantify the carbon emission of state-of-the-art large language models. As regards neural networks, this perspective has been explored by Patterson et al. [29]. Finally, in [30] the authors estimate the carbon footprint of medical image segmentation pipelines.

Regarding these works, the distinctive trait of this article is its analysis of the carbon footprint of recommender systems (RSs). RSs are a *data-intensive* technology that requires long training times and substantial energy. Moreover, it is worth noting that the training process is repeated almost every time users provide new evidence of their preferences, further exacerbating the problem. Unfortunately, while the problem of estimating the $CO_2$ footprint of AI algorithms has been addressed in several recent papers [29,31], the estimation of the $CO_2$ footprint of RSs is currently under-investigated. The only exceptions are [32] (of which this article represents a significant extension) and [33], in which the authors repeated the experiments presented in [32] on a different recommendation framework and observed the same findings. This article significantly extends the findings of [32] by discussing the impact of hyperparameter tuning and investigating whether data reduction strategies can be a suitable solution for building recommendation algorithms with lower emissions. Indeed, these topics are gaining increasing attention in the literature, as demonstrated by the call to action from Beel et al. [34] to design greener recommender systems, as well as by recent tutorials presented at major conferences in the field [35].

Regarding the role of hyperparameter tuning, numerous publications have demonstrated that it significantly impacts energy costs. Accordingly, some attempts initially aimed to reduce the number of iterations required to tune the parameters [36], or to incorporate power consumption into the optimization process [37,38]. The impact of hyper-parameter optimization on energy consumption is also discussed in [39]. Regarding the aforementioned work, the novelty of the current contributions lies in their focus on recommendation algorithms. To our knowledge, the impact of hyper-parameter tuning on the trade-off between accuracy and efficiency of recommendation algorithms has never been investigated.

Finally, the third hallmark of the current work is the analysis of the impact of data reduction on carbon footprint and the accuracy of the algorithms. In this research line, Zhang et al. [40] showed that many neurons in deep learning models are redundant and can be removed to reduce energy consumption. Similarly, in [41], the authors demonstrate that discarding outdated information enhances the mining of data streams. Other works focus on finding a trade-off between accuracy loss and energy consumption [42]. In this case, the novelty of the work lies again in the fact that the impact of data reduction techniques on the trade-off between efficiency and accuracy in recommender systems is overlooked in current research. The only partial exception is [43], which focuses on the recommendation task and shows that the popularity of an item in the movie domain can be improved when only the most recent information is considered. Similarly, in [44], the authors show how considering only the most recent information to train a RS improves its accuracy. Regarding these pieces of literature, this contribution conducts an extensive empirical benchmark of several representative recommendation algorithms using a reproducible experimental protocol.

Finally, regarding tools to quantify the carbon footprint of algorithms, it is worth mentioning the Machine Learning Emissions Calculator [31] and the framework introduced in [11], which aims to estimate the carbon footprint of any computational task. Similarly, in [45] the authors present ECO2AI, a package that tracks the energy consumption of AI models. As discussed in more detail in the following sections, this work uses CodeCarbon to estimate the carbon footprint of recommendation algorithms.

Among other relevant works in this area, [46] proposes a framework for estimating the $CO_2$ emissions generated by a recommender system (RS) algorithm when executed on a given dataset and a specific hardware platform. To this end, the authors trained several regression models on a dataset comprising multiple runs of RS algorithms across different datasets and hardware configurations. By leveraging information about the dataset (e.g., number of users, items, and interactions, sparsity, and associated knowledge), the characteristics of the recommendation algorithm, and hardware specifications, they accurately predicted the emissions produced by each run with a small prediction error.

Another work worth discussing is presented in [47], in which the authors propose a Green Early-Stop (GES) Criterion. The core idea of the GES is to track the emissions generated during the training of an RS algorithm and stop training when the improvement in the validation metric is too marginal to justify the increase in emissions.

[3] https://sdgs.un.org/2030agenda

## 3. Basics of recommender systems

Recommender Systems [48] represent a class of AI systems that provide users with personalized recommendations based on their past choices and preferences [49]. The first approaches for RSs only relied on users' rating, and were based on the assumption that similar users share similar interests; this approach is known as *Collaborative Filtering* (CF) [50], and is typically implemented through matrix factorization techniques [51] or, more recently, its variants based on deep learning and neural networks [52].

In parallel, *content-based RSs* [53] emerged: such systems exploit descriptive item features (*e.g.*, based on the genre of a movie or the topic of a news) to build item profiles; similarly, a user profile is built based on the elicitation of users' interest. In this way, it is possible to provide users with recommendations by matching users' interests with item profiles. Both paradigms evolved, resulting in systems that leverage both CF information and knowledge from large, external knowledge bases; such systems are referred to as Knowledge-aware Recommender Systems (KARSs) [54]. Several KARSs have been proposed in the literature, and most of them are based on Knowledge Graphs (KGs) [55,56] that encode *side-information* related to the items; other KARSs exploit not only KGs, but also other knowledge sources [57,58].

In this work, we benchmarked 14 recommendation algorithms that represent the paradigms currently at the state of the art. In particular, we included general recommendation models, algorithms that utilize deep learning techniques, graph-based methods, and recent knowledge-aware algorithms. The selected models are listed below, along with a brief description of the algorithm. For the sake of completeness, it is worth emphasizing that we have limited the analysis to the models implemented in the recommendation library RecBole [59,60].[4] This guarantees a replicable and reproducible experiment, as suggested in [61]. Extension to further algorithms will be carried out as future work. The list of the recommendation algorithms follows:

- General recommendation models:
  - **Bayesian Personalized Ranking (BPR)** [62]: a recommendation model based on Bayesian statistics and optimized for the top-*k* ranking task.
  - **ItemKNN** [63]: a recommendation model that exploits neighbors and similarity measure to provide top-*k* recommendations.
  - **Large-scale Information Network Embedding (LINE)** [64]: a recommendation model suitable for large datasets, that efficiently learns embeddings to provide users with recommendations.
- Deep learning-based models:
  - **Deep Matrix Factorization (DMF)** [52]: a matrix factorization-based recommendation model implemented through deep neural networks.
  - **Variational Autoencoders for CF (MultiDAE)** [65]: a model that adapts the use of variational autoencoders to the recommendation in a CF scenario.
- Graph-based models:
  - **Spectral CF** [66]: a CF recommendation model that tackles the cold-start problem by performing convolutions on the spectral domain, to find more connections between users and items.
  - **Neural Graph Collaborative Filtering (NGCF)** [67]: a recommendation model which learns user and item embeddings and injects CF signals in the learnt embeddings.
  - **Disentangled Graph Collaborative Filtering (DGCF)** [68]: a recommendation model able to distinguish the various user intents when interacting with items. In particular, DGCF learns intent-specific embeddings via disentangled message passing, combined with a correlation regularization loss, and then combines them to provide users with recommendations.
  - **LightGCN** [69]: a Graph Convolutional Network (GCN) [70] that learns user and item embeddings (starting from a CF graph) by performing node aggregation through the mean operator, resulting in an efficient yet effective recommendation model.
- Knowledge-aware models:
  - **Collaborative Knowledge Base Embeddings (CKE)** [57]: a recommendation model that learns heterogeneous embeddings from different knowledge bases.
  - **Collaborative Filtering over Knowledge Graphs for explainable recommendation (CFKG)** [55]: a recommendation model that exploits knowledge graphs to learn item embeddings that encode external knowledge, and provide recommendations;
  - **Knowledge Graph Convolution Networks (KGCN)** [56]: a recommendation model that exploits GCNs to learn item similarity starting from a knowledge graph and the attributes it encodes, learning item embeddings by combining the relevant attributes for a given node;
  - **Knowledge-aware Graph Neural Networks with Label Smoothness Regularization (KGNN-LS)** [71]: a recommendation model that exploits knowledge graph relations to build several user-specific weighted graphs and then learns user-specific item embeddings through GCNs with label smoothness regularization. Regularization is used in the loss function to enforce similar representations for semantically close entities.
  - **RippleNet** [72]: a recommendation model that uses knowledge graphs as a source of external knowledge; it propagates user preferences in items over the knowledge graph entities to extend the set of potential interests of users.

## 4. Tracking carbon emissions and energy consumption

Nowadays, many organizations and governments use the carbon dioxide equivalent[5] (or $CO_2$ equivalent, abbreviated as $CO_2$-eq) as a standard metric to track the emissions of various greenhouse gases. Given an algorithm $a$, a state-of-the-art method [73] to compute or estimate the $CO_2$-eq is based on the combination of the values of Carbon Intensity (CI) and Power Consumed (PC). Formally:

$$emissions(a) = CI \cdot PC \qquad [\text{kg/kWh kWh} \equiv \text{kg}] \tag{1}$$

The Carbon Intensity (CI) of the electricity consumed is calculated by combining the emissions from the various sources that generate it. Of course, both fossil and renewable sources have specific carbon intensities (i.e., known amounts of carbon dioxide emitted).

The values of CI generally depend on the energy mix used for computation, with each source having its own CO2 emissions. Indeed, the overall CI of an energy mix is commonly computed as a weighted average of the emissions of each energy source in the mix. Formally, let $S$ be an energy mix, composed of $s$ energy sources denoted as $e_S$;

[4] https://recbole.io/

[5] https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Glossary:Carbon_dioxide_equivalent

each source $e_s$ is present with a percentage of $p_s$ in the mixture; then, the Carbon Intensity *CI* of the mix is computed as follows:

$$CI = \sum_{s \in S} e_s \cdot p_s \quad (2)$$

However, since each different countries use different energy source mixtures, they have different carbon intensities; just to give some examples[6] [74], in 2022 Italy used a mixture composed of 8% of coal, 51% of gas, 2% of oil, 11% of hydroelectric, 7% of wind, 10% of solar, 6% of bio energy, and the rest is represented by other renewable sources (including geothermal, tidal and wave generation); in the same year, United Kingdom, on the other hand, used a mixture composed of 1.5% of coal, 38% of gas, 4% of oil, 15% of nuclear, 1.5% of hydroelectric, 25% of wind, 4% of solar, 11% of bio-energy; Since all these mixtures are different, *CI* for Italy and *CI* for UK in 2022 were different: we have 261 g of $CO_2$-eq per kilowatt-hour in UK, and 373 g of $CO_2$-eq per kilowatt-hour for Italy. This kind of analysis is not trivial, as it motivates big companies to deploy their data centers in specific countries [75], since the location where computation occurs significantly influences the overall carbon footprint.

In our case, to provide a fair, independent comparison, we used the world average as a reference mixture. The amount of emissions based on the *world average* mixture is about 475 g of $CO_2$-eq per kilowatt-hour. Next, regarding Power Consumption (PC), it reflects the total energy required and consumed by the hardware throughout the entire computation. It can be obtained by exploiting low-level libraries that track CPU, GPU, and other component usage (including RAM). For this paper, we have tracked the PC every 15 s and obtained an aggregated energy consumption result at the end of the computation. Currently several available tools can estimate or measure the $CO_2$-eq values of AI algorithms (and, more in general, any computation) by estimating *CI* and *PC* values; for this paper, our choice fall on CodeCarbon,[7] since it *measures* the energy consumption, resulting in a more precise and reliable tracking, while other tools (such as, ML $CO_2$ Impact[8]) just *estimate* it.

## 5. Experimental evaluation

To answer our research questions (RQs), a large benchmark of the recommendation models we previously presented was conducted. In particular, through the experiments, we aimed to answer the following RQs:

- **RQ1 - Accuracy vs Sustainability:** *Is there any trade-off between the performance and the carbon emissions of the main recommendation algorithms?*
- **RQ2 - Tuning of Hyper-Parameters:** *What is the impact of hyperparameter tuning on the trade-off between carbon footprint and performance of the algorithms?*
- **RQ3 - Data Reduction:** *Is data reduction a good solution to trade-off emissions and performance of recommendation algorithms?*

The high-level workflow we followed to answer our RQs is depicted in Fig. 1, and the methodology flowchart is shown in Fig. 2. To answer RQ1, we ran all algorithms with their optimal hyperparameters and assessed the trade-off between emissions and performance. To this end, for each recommendation model, we fed the algorithm with the training data. Next, we trained the model and collected the recommendations by also tracking $CO_2$ emissions. Next, to answer RQ2, we repeated the aforementioned protocol for all parameter combinations available for each algorithm, and also included in our analyses the *overall* amount of $CO_2$-eq emissions required to *find* the optimal combination of parameters. Finally, to answer RQ3, we split the original dataset into *5* equal-sized folds. Then, we run the experiments by feeding the algorithms only the first fold. Next, we added one fold at a time (starting with 20% of the data, then 40%, and so on) until the complete dataset was obtained. In this case, we compared both emissions and performance by varying the number of folds. Further details regarding all the experimental settings are provided next.

**Table 1**
Statistics of the Datasets.

| | Amazon-Books | Movielens1M | Mind |
|---|---|---|---|
| Users | 22,155 | 6036 | 23,679 |
| Items | 54,458 | 3192 | 4414 |
| Interactions | 1,465,871 | 757,451 | 1,048,575 |
| Sparsity | 99.88% | 96.07% | 98.99% |
| Interactions/users | 66.16 | 125.49 | 44.28 |
| Interactions/items | 26.92 | 237.3 | 237.56 |
| KG Entites | 26,316 | 20,454 | no data available |
| KG Relations | 18 | 13 | |
| KG Triples | 96,476 | 70,668 | |
| Items in the KG | 10,878 | 2823 | |

### 5.1. Experimental design

**Datasets.** To run our analysis, we exploited three state-of-the-art datasets in the area of movies, books, and news recommendations. In particular, we run our benchmarks on Movielens-1M, Amazon-Books, and Mind. The choice of three domains was made to ensure good generalizability of the results. Moreover, to ensure a replicable and reproducible evaluation protocol, each dataset has been uploaded to our repository.[9] Table 1 presents the statistics about Movielens-1M, Amazon-Books, and Mind. It is worth noting that knowledge features are not available in Mind, so knowledge-aware algorithms are not run on this data. To carry out the experiments, we initially split the datasets into train, validation, and test (80%, 10%, 10%). To obtain the splits, we relied on the splitting strategy implemented in RecBole. This ensures that each experiment uses the same splits.

**Recommendation Algorithms.** In this work, we used the RecBole implementation of each recommendation algorithm and limited our selection to those available in the library. In total, we evaluated 14 different recommendation algorithms (three general recommendation models, two deep learning models, four graph-based models, and five knowledge-aware models). The list has been previously discussed in Section 3. Clearly, the protocol can be easily replicated (and extended) with a different framework. To train our models, we exploited a system based on Ubuntu (version 20.04), equipped with an Intel i7 (3.40 GHz CPU) and a Titan X (12 GB) as GPU. The number of epochs was set to a fixed value (50 epochs for MovieLens, 20 epochs for Amazon Books, and 50 epochs for Mind). Epochs were set based on numbers chosen through convergence analysis, not arbitrarily.

**Recommendation Performance.** Evaluation metrics are calculated by using RecBole,[10] as well. We run our analysis by considering all the principal recommendation metrics. For simplicity, we present the results in terms of Recall, NDCG (*the higher, the better*), average popularity, and Gini Index (*the lower, the better*). This allows us to assess the accuracy, ranking, popularity bias, and diversity of the recommendations. In our repository, we have uploaded all the results, including other relevant metrics.

[6] https://ourworldindata.org/electricity-mix
[7] https://mlco2.github.io/codecarbon/
[8] https://mlco2.github.io/impact/
[9] https://github.com/swapUniba/RecSysCarbonFootprint
[10] https://recbole.io/docs/recbole/recbole.evaluator.metrics.html#module-recbole.evaluator.metrics

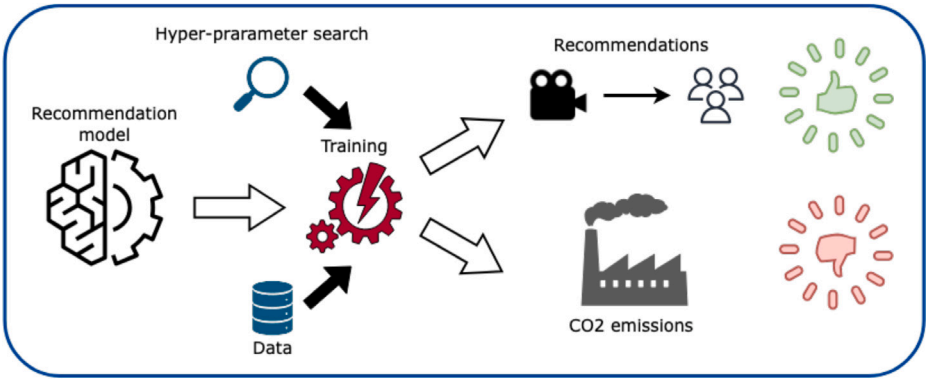



**Fig. 1.** High-level description experimental protocol.

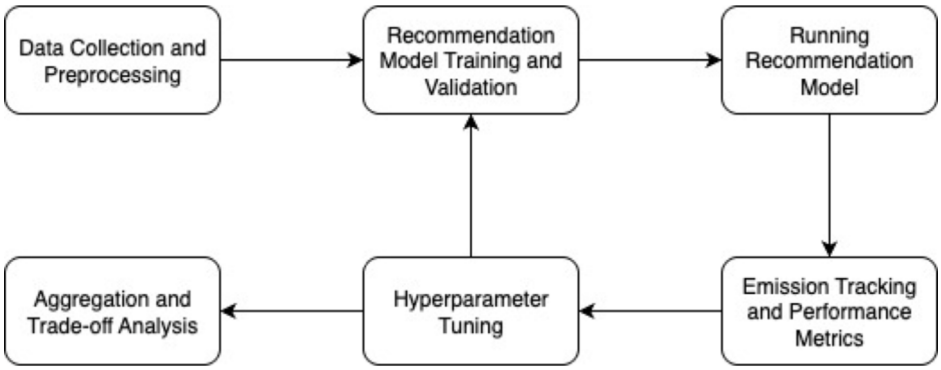



**Fig. 2.** Flowchart of the proposed methodology.

**Hyperparameter Tuning.** To answer RQ1 and RQ2, we designed a protocol for hyper-parameter tuning based on the iteration of several runs for each algorithm. In this case, we defined a set of suitable parameter values for each recommendation model. We adopted grid search to exhaustively explore the hyperparameter space and provide a comprehensive benchmark, rather than focusing on an optimized or energy-efficient search strategy. The complete list of parameters we considered for each algorithm is reported in Tables 2 and 3. Based on this list, we run each algorithm several times. The differences in the total number of runs reported in the tables stem from the heterogeneity of hyperparameter spaces across algorithms. Models with more tunable components (e.g., KGCN, CKE) naturally generate larger grid searches, while simpler models (e.g., BPR, DMF) require fewer combinations. In some cases, algorithms have similar run counts because of comparable parameterizations in RecBole, even though they belong to different families. These variations highlight the trade-off between model complexity, tuning flexibility, and associated carbon costs. Next, to answer RQ1, we selected the best-performing run (along with its emissions). To answer RQ2, we quantified the algorithm's emissions as the total emissions obtained by summing those from all runs. Additionally, for RQ2, we considered the best value obtained for each metric. The total number of runs is also presented in Table 3. It is worth noting that two of the models, *i.e.*, DGCF and RippleNet, were excluded from these analyses. Our decision was based on sustainability considerations. Indeed, in preliminary experiments, both models already exhibited substantially higher carbon emissions than other algorithms, even without parameter tuning. Running exhaustive hyperparameter searches for these models would have further inflated emissions by several orders of magnitude, while offering little additional insight into the trade-offs, as the results consistently favored simpler alternatives.

**Measuring Carbon Footprint.** As previously stated, to track the emission and the carbon footprint of the recommendation models, we used the Python library CodeCarbon; moreover, we used the world average value of Carbon Intensity.[11] In our experiments, we have consistently used the same hardware, which has been dedicated solely to conducting these experiments, with no additional workload imposed on the hardware. Power consumption (PC) was estimated using CodeCarbon's internal monitoring pipeline, which measures energy consumption across hardware by using specialized tools for each component and platform. It tracks GPU usage via the pynvml library, monitors CPU energy through Intel RAPL on Linux, Intel Power Gadget on older macOS and Windows systems, and powermetrics on Apple Silicon. RAM usage is estimated based on the number and type of memory modules. When direct measurements are not possible, CodeCarbon estimates CPU energy use from processor TDP values and CPU load data. No manual adjustments were applied, ensuring reproducibility across systems with identical configurations. More details about how CodeCarbon tracks the energy consumption can be found in the official documentation.[12] Additionally, we set CodeCarbon to track the single process that affects emissions, rather than the entire machine.[13] This is also true for other aspects, including the cooling overhead, since we used the same hardware and cooling fan; this factor can be normalized in our experimental setting without affecting the reliability and comparability of the results. All the values we reported are presented in grams (g).

To answer our RQs, we measured the carbon footprint of a recommendation algorithm $r$. To answer RQ1 and RQ3, we considered the emissions of the best-performing run, as calculated in Formula (1).

[11] https://www.iea.org/reports/global-energy-co2-status-report-2019/emissions

[12] https://mlco2.github.io/codecarbon/methodology.html

[13] see the tracking mode parameter: https://mlco2.github.io/codecarbon/parameters.html.

**Table 2**
List of hyper-parameter values set for the grid search (part 1).

| Model | Parameters | Total runs |
|---|---|---|
| **General Recommendation Models** | | |
| BPR | *embedding_size*: [32, 64, 128], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 27 |
| ItemKNN | *k*: [10, 50, 100, 200, 250, 300, 400, 500, 1000], *shrink*: [0.0, 1.0], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 162 |
| LINE | *embedding_size*: [32, 64, 128], *order*: [1, 2], *second_order_loss_weight*: [0.3, 0.6, 1], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 162 |
| **Deep Learning Models** | | |
| DMF | user_layers_dim: [[64,64],[64,32],[128,64]], item_layers_dim: [[64,64],[64,32],[128,64]], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 81 |
| MultiDAE | *latent_dimension*: [32, 64, 128], *mlp_hidden_size*: [[100], [300], [600]], *dropout_prob*: [0.0, 0.5], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 162 |

**Table 3**
List of hyperparameter values set for the grid search (part 2).

| Model | Parameters | Total runs |
|---|---|---|
| **Graph-based Recommendation Models** | | |
| SpectralCF | *embedding_size*: [32, 64, 128], *reg_weight*: [0.01, 0.003], *n_layers*: [1, 2], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 108 |
| NGCF | *embedding_size*: [32, 64, 128], *hidden_size_list*: [[64, 64, 64], [128, 128, 128], [256, 256, 256], [512, 512, 512]], *node_dropout*: [0.0, 0.1, 0.2], *message_dropout*: [0.0, 0.1, 0.2, 0.3], *reg_weight*: [1e−5, 1e-2], *delay*: [1e−5, 1e−4, 1e−3, 1e−2, 1e-1], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 216 |
| LightGCN | *embedding_size*: [32, 64, 128], *n_layers*: [1, 2, 3, 4], *reg_weight*: [1e−05, 1e-02], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 216 |
| **Knowledge-aware Recommendation Models** | | |
| CKE | *embedding_size*: [32, 64, 128], kg_*embedding_size*: [32, 64, 128], *reg_weights*: [[0.1, 0.1], [0.01, 0.01], [0.001, 0.001]], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 243 |
| CFKG | *embedding_size*: [32, 64, 128], *loss_function*: [inner_product, transe], *margin*: [0.5, 1.0, 2.0], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 162 |
| KGCN | *embedding_size*: [32, 64, 128], *aggregator*: [sum, concat], *reg_weight*: [1e-7], *neighbor_sample_size*: [2, 3, 4], *n_iter*: [1, 2], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 324 |
| KGNN-LS | *embedding_size*: [32, 64, 128], aggregator: [sum, neighbor, concat], *reg_weight*: [1e−5, 1e-7], *neighbor_sample_size*: [4], *n_iter*: [1], *ls_weight*: [0.5], *epochs*: [10, 50, 100], *learning_rate*: [0.01, 0.001, 0.0001] | 162 |

Conversely, for RQ2, we calculated emissions as the sum across all $n$ runs of the algorithm $r$. Formally:

$$totalEmissions(r) = \sum_{i=1}^{n} emissions(r_i) \quad (3)$$

where $r_i$ represents the $CO_2 - eq$ emitted by the $i$th run of the algorithm $r$, and $n$ is the total number of runs required to tune the hyperparameters (see Tables 2 and 3).

**Data Reduction Protocol.** As previously stated, to answer RQ3 we apply data reduction. In particular, we randomly split the training data into $n = 5$ non-overlapping subsets (*folds*). Then, we trained the models with gradually larger training sets. This is done by starting with one of the subsets (20% of the training data), then merging it with the second (40% of the training data), and so on, until the complete dataset is obtained again.

### 5.2. Discussion of the results - RQ1: Accuracy vs sustainability

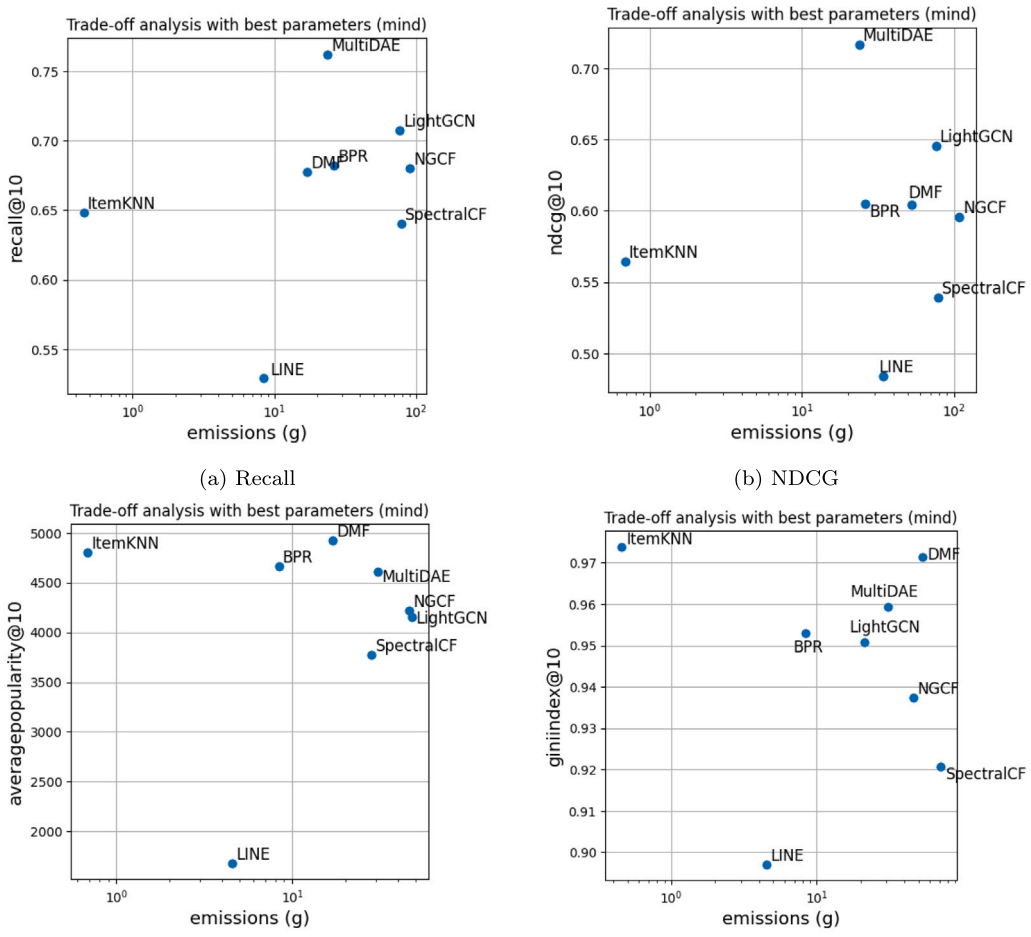
First, we analyzed the trade-off between model performance and carbon emissions. The results, calculated for all the datasets, are presented in Figs. 3 (Mind), 4 (MovieLens), and 5 (Amazon).

As regards **Mind datasets**, ItemKNN has the lowest carbon footprint (0.45 g) while NGCF has the highest value (around 89 g). Regarding performance, MultiDAE (emitting 23 g) achieved the best overall recall and NDCG. Accordingly, it seems that the use of complex learning mechanisms, as implemented in MultiDAE, leads to an improvement in the accuracy of recommendations, at the cost of an approximately $50x$ increase in emissions w.r.t. simpler models such as ItemKNN.

Another interesting outcome of the experiment is that *high-emissions* models such as SpectralCF (emitting 78 g of $CO_2$-eq) do not support their computational demands with an equivalent increase in performance. On the contrary, a different behavior emerged when non-accuracy metrics are taken into account. Indeed, the increase in $CO_2$ Emissions may justify SpectralCF and LINE based on the Gini Index, whose scores are significantly lower than those of the other algorithms we considered. Conversely, the analysis of average popularity shows that an algorithm with lower emissions, such as LINE, again produces results better than those of high-emissions algorithms, such as BPR and DMF. Clearly, it is worth emphasizing again that all the results are obtained by considering the best parameters.[14] However, it is important to stress that adopting parameter tuning procedures, *i.e.,* grid search, entails multiple algorithm runs, which in turn further increase overall carbon emissions.

Next, we analyze the results obtained for **MovieLens data**. Simple models have very low emissions (0.31 g of $CO_2$ equivalent for ItemKNN, 3.25 for BPR), while complex models have a close carbon footprint (44 g for DGCF) to the one we obtained on Mind data. The first relevant outcome of this dataset is that models in the *knowledge-aware* family exhibit a large amount of carbon emissions (40 g for LightGCN). Unfortunately, such a demand is only partially supported by their performance, as models with lower emissions, such as MultiDAE and CFKG, obtain better results. Next, the analysis of the NDCG

[14] The amount of emissions is based *only* on the analysis of the *best* run. The role of tuning will be investigated in RQ2.




(a) Recall (b) NDCG

(c) Avg. Pop. (d) Gini Index

**Fig. 3.** Comparison between carbon emissions (*X-axis*) and performance *(Y-axis)* on **Mind** data. For Recall, NDCG, the higher, the better, while for Average Popularity and Gini index metrics, the lower, the better. *X-axis values report the emissions of the best-performing run of a specific algorithm. Y-axis reports the emissions in a log-10 scale.*

yields findings consistent with those obtained for the Mind dataset, as CFKG justifies the increase in carbon footprint with an equivalent improvement in ranking precision. Similar results were also obtained for MultiDAE and LightGCN. Finally, regarding non-accuracy metrics, results generally show that *higher-emissions* approaches such as CFKG and LightGCN tend to reduce popularity bias and increase diversity. However, results clearly show that methods such as LINE generally offer the best trade-off, as they achieve a relatively low carbon footprint (just 1.4 g of $CO_2$-eq).

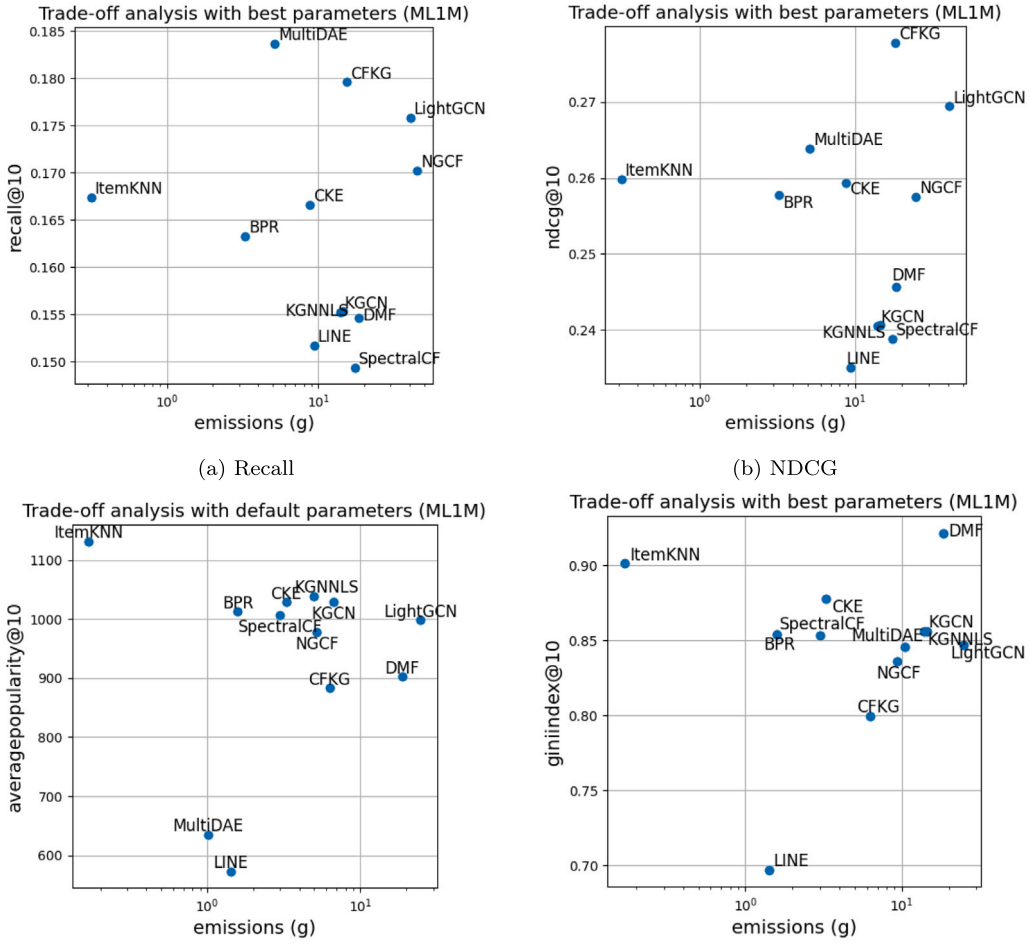
Finally, as regards **Amazon Books**, the lowest emissions were obtained with ItemKNN (0.65 g), while the highest carbon footprint was observed with NGCF (261 kg of $CO_2$ equivalent). Different from what emerged in the previous analyses, here ItemKNN provides the best trade-off between carbon emissions and ranking accuracy, as shown by NDCG results. Conversely, algorithms with higher emissions, such as CKE and CFKG, justify their increased emissions with an equivalent increase in predictive accuracy (as shown by the Recall). Next, knowledge-aware algorithms such as SpectralCF do not justify their computational requirements by delivering comparable performance.

**RQ1 - Take-Home Messages:** To sum up, we can answer the first research question by clearly stating that the choice of the recommendation algorithm having the optimal trade-off between performance and carbon emissions is not trivial. Indeed, ItemKNN emerged as the best option in terms of the trade-off between emissions and recall across all settings. Other algorithms, such as MultiDAE, yield good results on Mind and MovieLens, but with a notable increase in terms of emissions that must be carefully analyzed and considered when selecting the algorithm. Moreover, algorithms such as LINE provide a good compromise between non-accuracy metrics and emissions. This may be important for identifying the most suitable algorithm in settings where non-accuracy metrics must be prioritized. Finally, a key observation from our benchmark is the systematic underperformance of knowledge-aware methods in terms of sustainability. These models, by design, rely on external knowledge graphs, which significantly increase training time and energy consumption. Although such complexity can lead to improvements in specific scenarios, our results show that proportional gains rarely offset the carbon costs of improved predictive performance. This highlights a structural trade-off: while knowledge-aware recommenders offer richer modeling capacity, their computational overhead raises concerns about sustainable deployment, particularly in industrial settings where efficiency and scalability are critical.

### 5.3. Discussion of the results - RQ2: Hyper-parameter tuning

Next, to answer RQ2, we followed the protocol previously presented and ran hyperparameter tuning across all recommendation models, varying the parameters listed in Tables 2 and 3.

As previously stated, in this part of the experiment, the algorithm's emissions are calculated as the sum of emissions from each run. For each metric, we considered the best value across all runs (as in RQ1). Of course, the goal of the experiment is to quantify how the search for the optimal hyperparameter combination affects the trade-off between




(a) Recall (b) NDCG

(c) Avg. Pop. (d) Gini Index

**Fig. 4.** Comparison between carbon emissions (*X-axis*) and performance *(Y-axis)* on **MovieLens** data. For Recall and NDCG, the higher, the better; for Average Popularity and Gini index metrics, the lower, the better. *X-axis values report the emissions of the best-performing run of a specific algorithm. Y-axis reports the emissions in a log-10 scale.*

emissions and performance. Results are presented in Figs. 6, 7 and 8. Generally speaking, the results are in line with expectations: all datasets show that the search for optimal hyperparameters significantly increases the algorithms' overall emissions; however, this increase is not offset by an equally significant improvement in performance. For example, for Recall, MultiDAE now emerges as the best-performing algorithm, with a 14.6% increase (from 0.6241 to 0.7169). However, this comes at the cost of an exponential increase in emissions, rising from 5.9 g (with default parameters) to almost 7000 (equivalent to 7 kilos, by summing all the runs). Similar findings hold for LightGCN. Overall, less sophisticated algorithms such as BPR emerge as suitable alternatives, achieving a 28.59% increase in performance at the cost of 260 g of emissions, which is 35 times lower than NGCF. Finally, the use of hyperparameter tuning for very simple algorithms, such as ItemKNN, must be carefully considered. Indeed, in this case, the increase in performance is relatively small (from 0.6456 to 0.6481), while the emissions are considerably higher (from 2 g to 170 g).

Next, by extending the analysis to other datasets, *i.e.*, ML1M, it is possible to study the impact of hyper-parameter tuning on knowledge-aware methods such as CKE. In this case, the introduction of knowledge features led to a 28.68% increase in recall; however, emissions increased to 1600 g (from 2.5 g) exponentially. Based on these data, MultiDAE emerged again as the algorithm that offers the best compromise between performance and emissions, alongside CFKG.

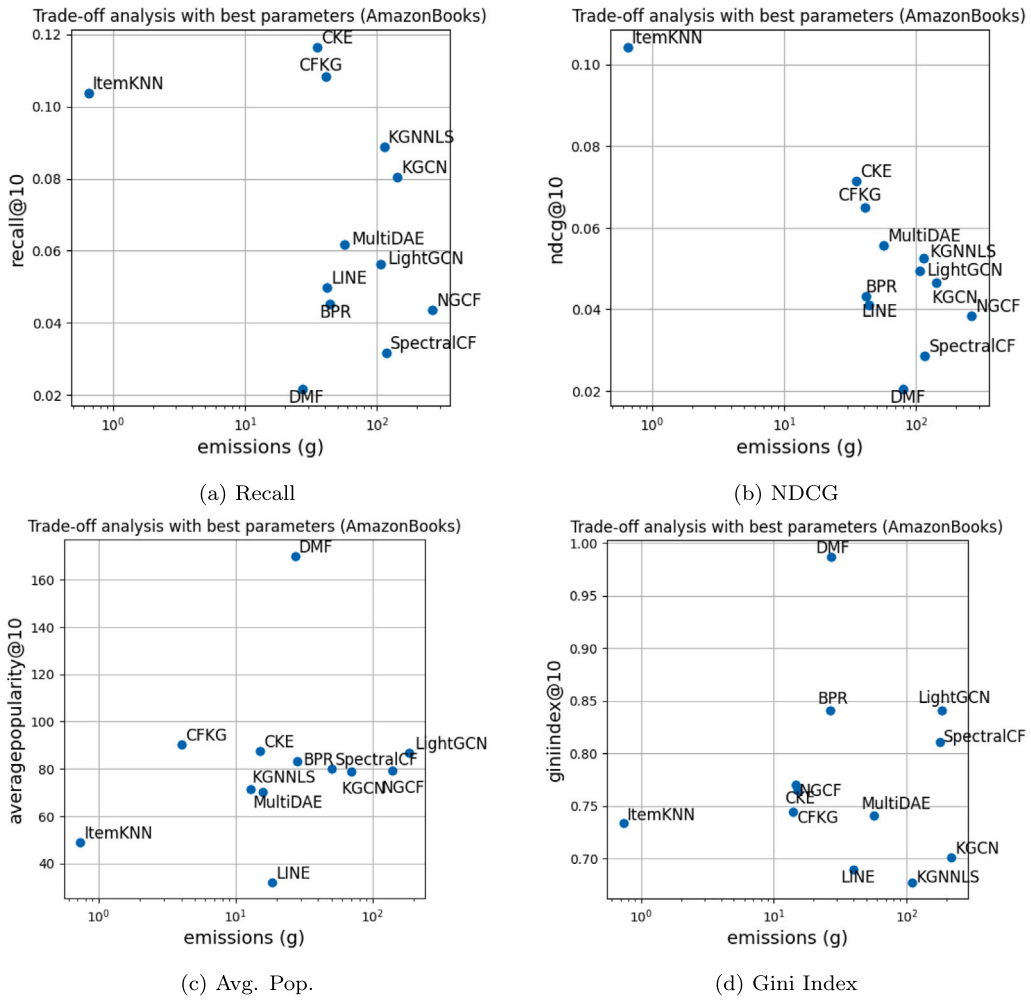
All the issues that characterize hyperparameter tuning clearly emerge from analyzing Amazon results. In this case, algorithms such as NGCF, KGCN, and LightGCN emit a substantial amount of $CO_2$-eq (between 14 and 16 kilos[15]). However, their performance still significantly lags behind that of algorithms such as ItemKNN. Overall, methods such as CKE and CFKG emerged again as the strategies that offer the best compromise between increased performance and reduced emissions.

**RQ2 - Take-Home Messages:** To sum up, all these findings showed that the common practice of tuning hyperparameters to optimize the performance of the algorithms has to be carefully considered. Indeed, in many settings, the increase in performance is not justified by the massive amount of emissions required to find the best-performing configuration of the algorithm. Conversely, less sophisticated algorithms typically offer a better compromise, as they allow for a better trade-off.

### 5.4. Discussion of the results - RQ3: Data reduction

To answer RQ3, we plotted (Figs. 9(a), 9(b) and 9(c), for ML1M, Amazon-Books and Mind, respectively) the trends of the emissions of $CO_2$-eq on the varying of the different data reduction settings (*i.e.,* with different training data). As shown in the figures, we can observe that the trends are similar, with DGCF, RippleNet and NGCF being the models emitting the most quantity of $CO_2$-eq for both the datasets, regardless of the amount of training data. From a quantitative perspective, the experiment demonstrated that data reduction significantly

[15] For the sake of readability, we preferred to use kilos instead of grams.




(a) Recall (b) NDCG

(c) Avg. Pop. (d) Gini Index

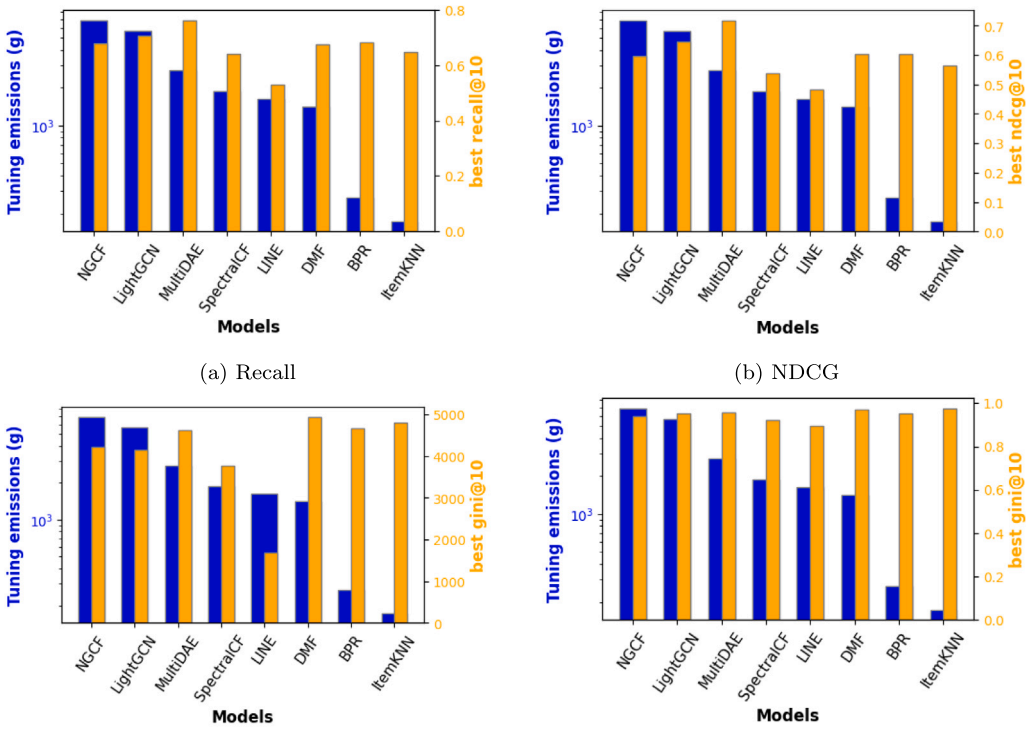
**Fig. 5.** Comparison between carbon emissions (*X-axis*) and performance *(Y-axis)* on **Amazon-Books** data. For Recall and NDCG, the higher, the better; for Average Popularity and Gini index metrics, the lower, the better. textitX-axis values report the emissions of the best-performing run of a specific algorithm. *Y-axis reports the emissions in a log-10 scale.*

impacts model emissions. For example, DGCF algorithm on Amazon-Books emits about 125 g of $CO_2$-eq when trained on the 20% of the training data, and almost 3000 g of $CO_2$-eq when trained on the whole dataset (more than 20x increase). On the contrary, DMF emits 70 g of $CO_2$-eq when trained on the entire dataset, and only 15 g when trained on 20% of the available data (a more than fourfold increase). Given these results, we grouped the models into two clusters: *sustainable* and *not sustainable*. To assign a model to a specific cluster, we calculated the decrease in $CO_2$-eq by reducing the amount of training data to 20%. The first group consists of models that emit more $CO_2$-eq (DGCF, NGCF, LightGCN), and the second comprises more sustainable models (the remaining ones). The clusters will be used and treated differently to answer the research question, as we have obviously focused on models with higher emissions.

Next, we considered the impact of data reduction on recommendation metrics; results are plotted in Figs. 10, 11, and 12, for ML1M, Amazon-Books, and Mind, respectively. For the less sustainable models, we plotted the metrics at varying data reduction levels. For the more sustainable options, we only considered performance and emissions when the entire dataset was included. This choice was made because the amount of $CO_2$-eq emitted by sustainable models is significantly lower than that of less sustainable ones, so it is not necessary to apply data reduction to these algorithms.

As expected, data reduction reduces the accuracy of the models. Of course, the more the data is reduced, the greater the decrease in performance. For example, on Amazon-Books, the recall of LightGCN decreases from 0.53 (with no data reduction) to 0.18, even though the emissions significantly drop. Similarly, emissions of DGCF decrease noticeably, but the recall decreases from 0.47 to 0.12, and similar outcomes emerge from ML1M. Of course, in most cases, a substantial decrease in accuracy is obtained. This is not surprising, as reducing the training data makes the recommendation task more challenging. However, in some settings, we observed that data reduction can make a non-sustainable algorithm, such as LightGCN, perform more sustainably, akin to CKE. Indeed, when LightGCN is trained on 60% of the Amazon books data, both recall and emissions are very close, with a decrease in data storage requirements following data reduction.

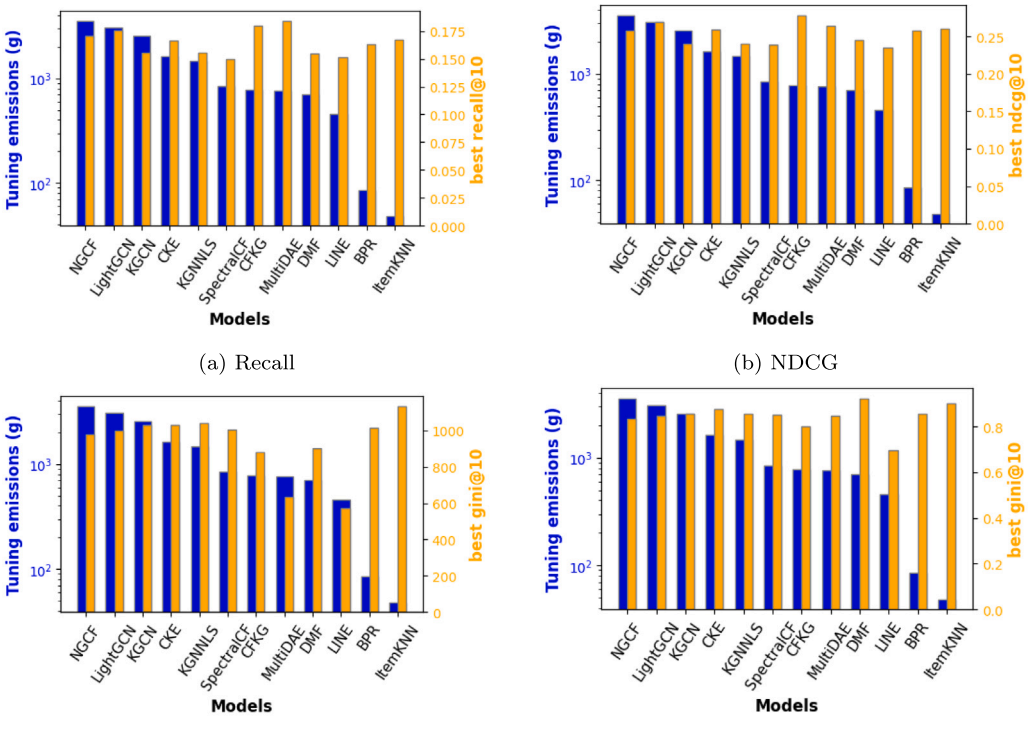
On the other hand, much more interesting outcomes emerge when non-accuracy metrics (Gini index and popularity) are analyzed. Indeed, in this case, the decrease in emissions resulting from the reduction of training data also leads to an increase in diversity and a decrease in the average popularity of the recommendations (*i.e.,* less popularity bias). This is an exciting outcome, showing that data reduction improves non-accuracy metrics as a noteworthy *side* effect. For example, on the ML1M dataset, both the Gini Index and Average Popularity improve notably. With 80% of the training data, Gini increases by 0.5% and popularity decreases by 20%. These effects grow as training data decreases: at 60%, Gini rises by 5% and popularity drops by 57%; at 20%, by 8% and 77%, respectively. Similar trends appear on AmazonBooks: with 80% of data, Gini improves by 1.3% and popularity decreases by 17%;




(a) Recall (b) NDCG

(c) Avg. Pop. (d) Gini Index

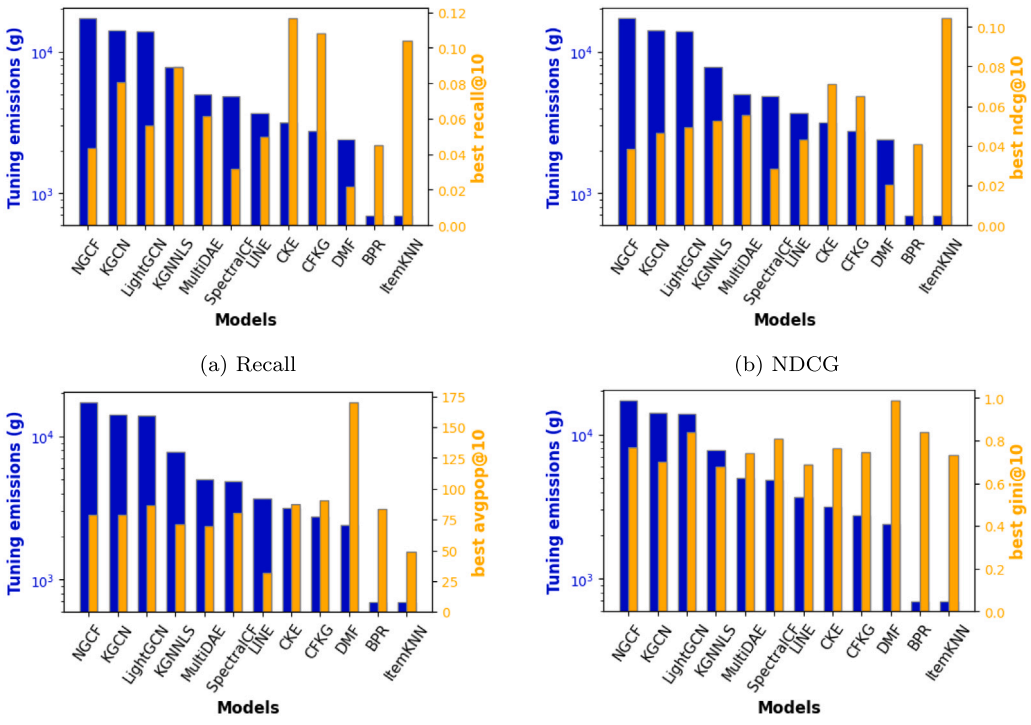
**Fig. 6. Mind Dataset.** Cost of the hyperparameter tuning in terms of $CO_2$ emissions (g) and performances obtained in terms of recall. Blue bars represent the emissions, while orange bars indicate the considered metric. *Y-axis values report the emissions obtained by summing all the runs of a specific algorithm, and are reported in a log-10 scale.* (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)




(a) Recall (b) NDCG

(c) Avg. Pop. (d) Gini Index

**Fig. 7. ML1M Dataset.** Cost of the hyperparameter tuning in terms of $CO_2$ emissions (g) and performances obtained in terms of recall. Blue bars represent the emissions, while orange bars indicate the considered metric. *Y-axis values report the emissions obtained by summing all the runs of a specific algorithm, and are reported in a log-10 scale.* (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)




(a) Recall (b) NDCG

(c) Avg. Pop. (d) Gini Index

**Fig. 8. Amazon-Books Dataset.** Cost of the hyperparameter tuning in terms of $CO_2$ emissions (g) and performances obtained in terms of recall. Blue bars represent the emissions, while orange bars indicate the considered metric. *Y-axis values report the emissions obtained by summing all the runs of a specific algorithm, and are reported in a log-10 scale.* (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

with 60%, by 2.8% and 35%; and with 20%, by 6% and 74%. On the Mind dataset, popularity decreases by 32% and 77% with 60% and 20% of training data, respectively, while Gini Index improves modestly by 0.3% and 1.2%. Accordingly, we can state that data reduction led to an expected decrease in accuracy and to an unexpected (but essential) increase in diversity and a decrease in popularity bias.

**RQ3 - Take-Home Messages:** Results show that all the algorithms provide a significant decrease in both emissions and accuracy, balanced by a substantial increase in terms of non-accuracy metrics. These findings can be very interesting for developing a new generation of multi-objective recommender systems [76] that also consider sustainability. For example, a company that wants to focus on recommending less popular items or providing more diverse recommendations might train the models using just a few training data points. From this perspective, this guarantees the best performance and very low $CO_2$-eq emissions. On the other hand, if a company needs to increase the accuracy of its recommendations, it needs more data, and a lighter data reduction might be the best trade-off.

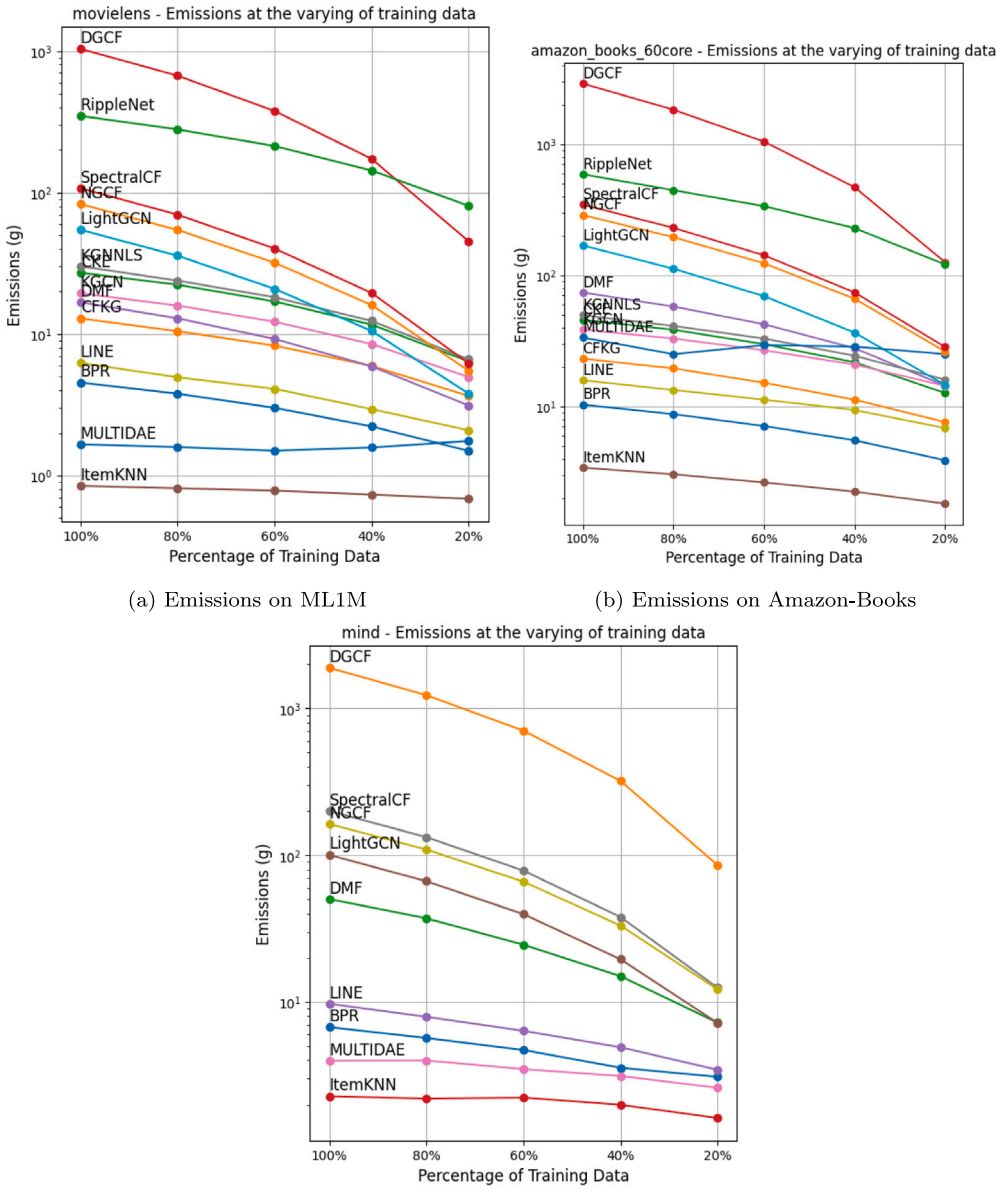
## 6. Discussion and limitations

Overall, there is global concern about sustainability and how AI (and RSs) algorithms impact overall emissions. Our paper allows us to shed light on this problem, and our analyses related to the trade-off between emission and performance, to the role of hyper-parameter tuning, and to the impact of data reduction, provide the community with a valuable outcome that can increase awareness and clarify the potential role of recommendation technologies in supporting sustainability goals. In our vision, there is an urgent need for sustainability-aware implementations, and our contribution provides researchers and practitioners with findings and take-home messages that have not been discussed previously.

From an applied perspective, our results suggest that the choice between less sophisticated and state-of-the-art algorithms should not be viewed as absolute, but rather as context-dependent. When energy budgets are limited or when sustainability goals are prioritized, simpler models such as ItemKNN or LINE often deliver a better balance between emissions and accuracy. Conversely, in mission-critical applications where accuracy outweighs sustainability concerns, advanced models may remain the preferred choice. Nonetheless, even in these cases, strategies such as reducing the number of hyperparameter tuning iterations or adopting data reduction can help mitigate energy costs. In practice, industry practitioners may consider hybrid pipelines where lightweight models serve standard workloads, while advanced models are selectively applied in scenarios that demand maximum predictive accuracy. This dual approach enhances sustainability without completely sacrificing performance. Moreover, our findings suggest that environmental objectives and social aspects are not in conflict but can often be jointly optimized. Data reduction, while lowering emissions, also reduced popularity bias and increased diversity, leading to fairer recommendation outcomes. This suggests that energy-efficient strategies may simultaneously promote social sustainability goals such as inclusivity and equity. Thus, future work should explicitly model these objectives together in multi-objective recommendation frameworks.

To conclude our analysis, we can identify the study's limitations and summarize the main findings of our benchmark.

- **Carefully Choose Your Algorithm:** Simpler algorithms generally obtain satisfying performance by emitting just a few $CO_2$-eq in comparison to more complex models. Accordingly, the choice of the best recommendation model has to be carefully considered.




(a) Emissions on ML1M

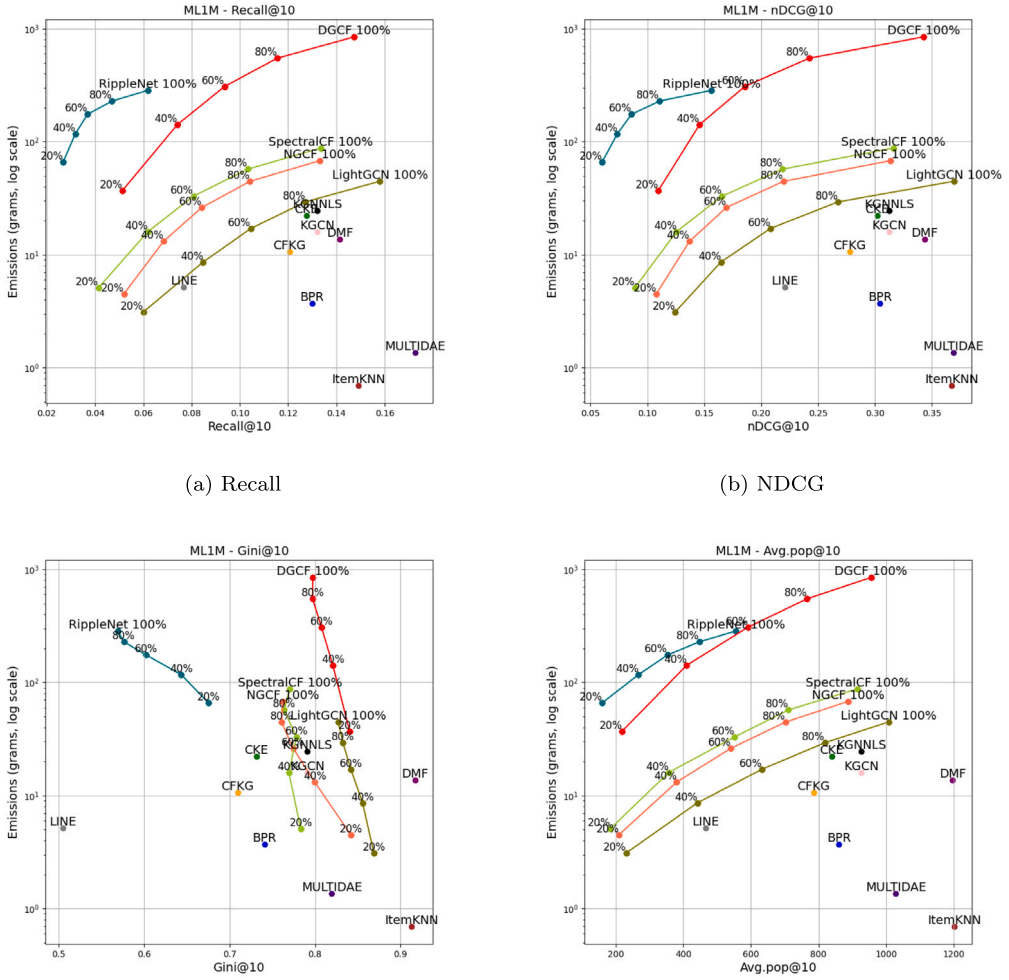
(b) Emissions on Amazon-Books

(c) Emissions on Mind

**Fig. 9.** Total emissions of the considered recommendation models at varying quantities of training data. The number of epochs was kept constant when reducing the dataset size. *Y-axis values report the emissions obtained by a single run of the algorithm, in a specific data-reduction setting, on a log-10 scale.*

- **Geographical Impact on Computation:** Our analyses showed the importance of computation location. While our study uses global average Carbon Intensity values for fairness, the choice of computation site can substantially affect emissions. For example, we have trained DGCF on AmazonBooks in Finland and Singapore, revealing a 37.8% gap in terms of emissions (23 kg in Finland, 37 kg in Singapore[16]). Accordingly, opting for areas with lower Carbon Intensity can enable the use of more advanced algorithms with comparable emissions.
- **Data Reduction Can Be a Solution:** Reducing the amount of data can lower energy consumption. Of course, this impacts performance metrics, so it is fundamental to balance data reduction with constraints and goals to align with company objectives and sustainability goals. This is in line with the outcomes of similar studies [77], comparing different data reduction strategies for recommendation tasks.
- **Social Impact of Data Reduction:** Reduced data sets can mitigate biases and enhance diversity, promoting social considerations within recommender system outcomes. This aligns with social sustainability goals, addressing gender equality and reduced inequalities.
- **Evaluation at Scale is Needed:** The highest values for carbon emissions are relatively low. For example, by also considering daily retraining of the algorithm with the highest emissions (NGCF on Amazon-Books), the overall emissions are 5.8 tonnes $CO_2 - eq$ (16 kg $\times$ 365), which is about 1/4 of BLOOM and still $<$ 2% of GPT-3. Accordingly, larger-scale evaluations are necessary for comprehensive insights. At scale, more significant differences in carbon footprint among algorithms may become even more pronounced, potentially exerting a greater environmental impact.

[16] Regarding the nation-wise carbon intensity values, they are based on International Energy Agency (IEA) data, which report average grid carbon intensity (g $CO_2$/kWh) for each country or region.




(a) Recall (b) NDCG

(c) Gini index (d) Avg. Pop.

**Fig. 10. ML1M Dataset.** Trade-off between recommendation metrics *(X-axis)* and grams of $CO_2$ emissions (*Y-axis*). The number of epochs was kept constant when reducing the dataset size. *Y-axis values report the emissions obtained by a single run of the algorithm in the different data-reduction settings, on a log-10 scale.*

- **Design Multi-objective Approaches:** Achieving balance across sustainability pillars demands multi-objective approaches. Balancing business objectives with sustainability goals requires careful selection of algorithms and data to optimize fairness-related metrics, emissions, and accuracy, suggesting avenues for developing novel multi-objective recommendation algorithms [78].
- **Definition of a Solid Protocol:** While all these findings emerged from an analysis focusing on recommendation algorithms, it is essential to point out that the experimental protocol is general enough to apply the same methodology and the same principles to other *data-intensive* AI algorithms, to extend the findings to different AI areas and AI applications.

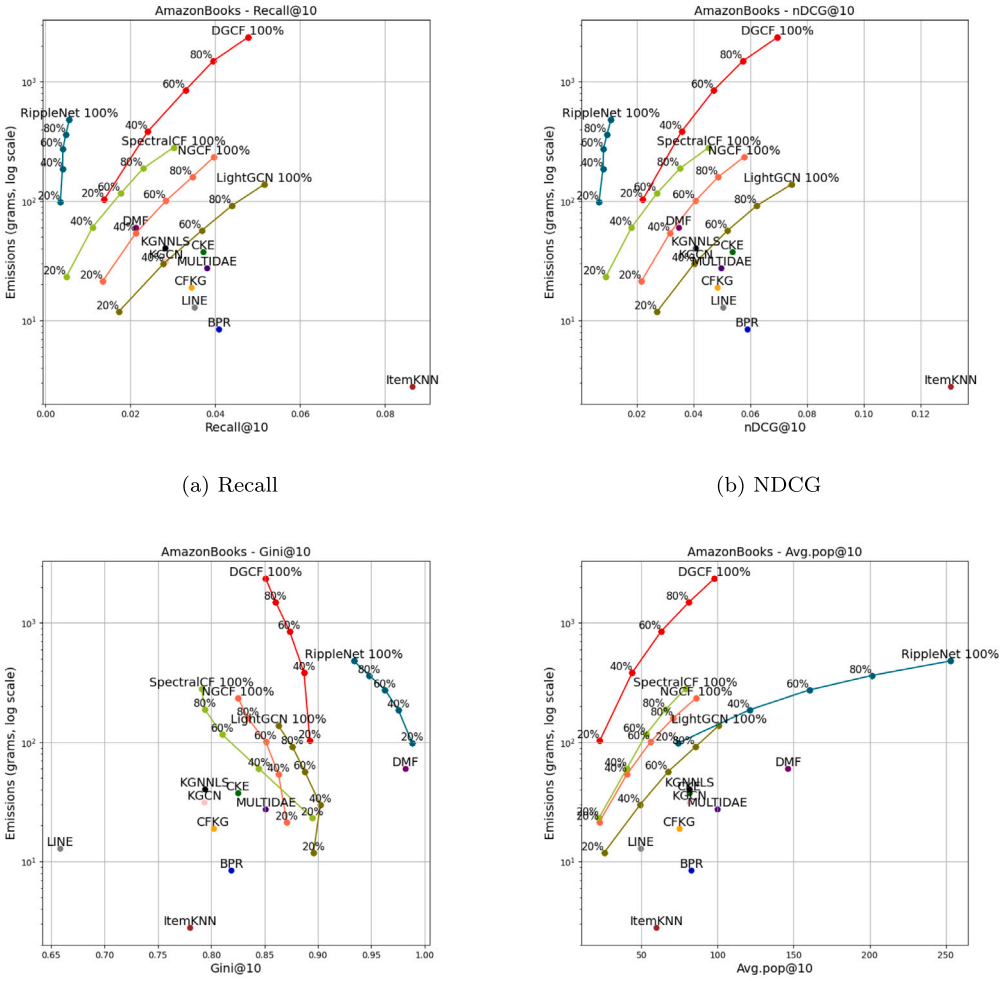
## 7. Conclusions

In this paper, we discuss the results of an extensive benchmark that investigates the trade-off between performance and carbon emissions of state-of-the-art recommender systems. Beyond the empirical results, our main contribution is the proposal of a sustainability-aware benchmarking methodology that integrates emissions tracking into reproducible RS evaluation protocols. This methodological framework can serve as a reference for future studies in the field.

Generally speaking, our experiments showed that less sophisticated algorithms, such as ItemKNN, still represent a viable alternative for implementing greener recommendation algorithms (−50x lower emissions than more recent implementations, with comparable accuracy). Moreover, we showed that hyperparameter tuning dramatically increased the emissions (up to 1000x) with an increase of performance that does not support that choice (between 10%–15% in most of the comparisons). Finally, we discussed the impact of data reduction techniques, which emerged as a suitable solution to reduce both popularity bias and emissions, thereby simultaneously decreasing carbon emissions.

These insights carry broader implications for RS design: future systems should adopt multi-objective approaches that balance accuracy, fairness-related metrics, and carbon emissions, paving the way for algorithms that are environmentally efficient and socially responsible. For practitioners, our results suggest that careful algorithm selection and lightweight training strategies can align recommendation performance with organizational sustainability goals.

As future work, we will extend our benchmark to consider different algorithms and machines. Moreover, we will move toward multi-objective recommendation by designing new approaches that simultaneously account for sustainability and accuracy.



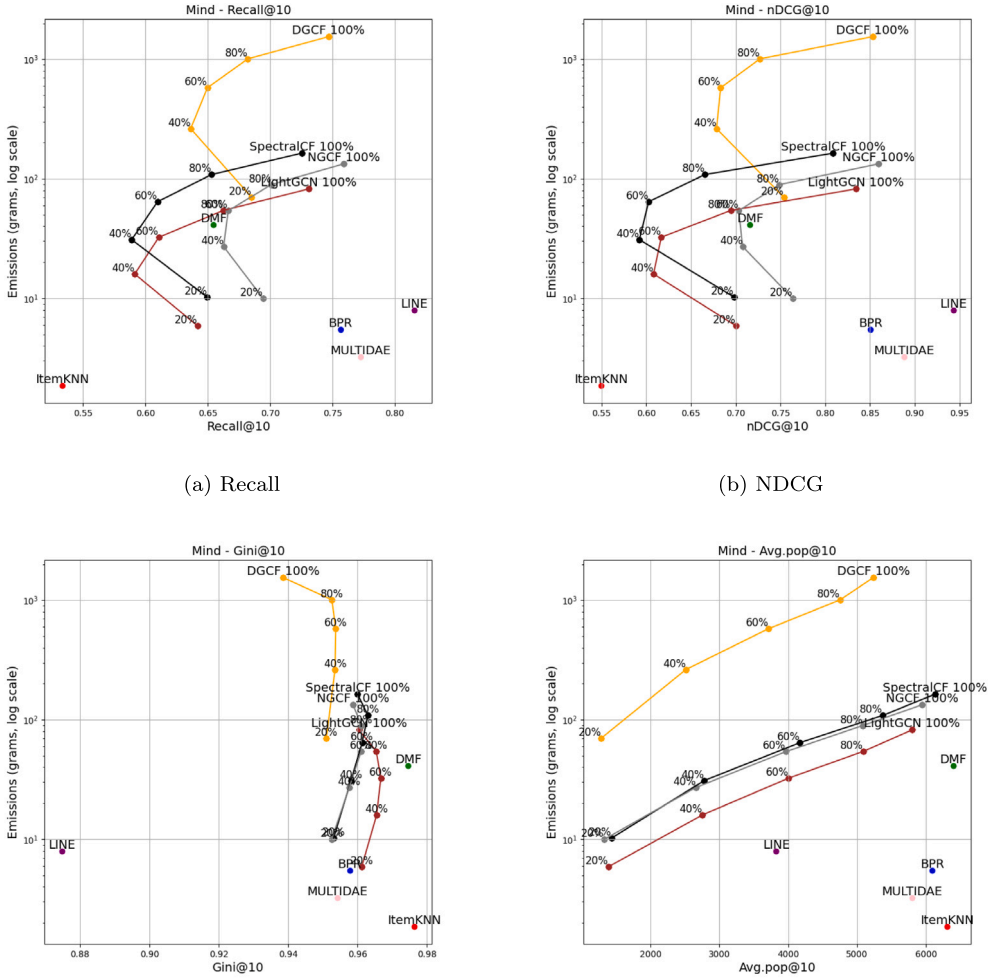

(a) Recall

(b) NDCG

(c) Gini index

(d) Avg. Pop.

**Fig. 11. Amazon-Books Dataset.** Trade-off between recommendation metrics *(X-axis)* and grams of $CO_2$ emissions (*Y-axis*). The number of epochs was kept constant when reducing the dataset size. *Y-axis values report the emissions obtained from a single run of the algorithm across different data-reduction settings, on a log-10 scale.*




(a) Recall (b) NDCG

(c) Gini index (d) Avg. Pop.

**Fig. 12. Mind Dataset.** Trade-off between recommendation metrics *(X-axis)* and grams of $CO_2$ emissions (*Y-axis*). The number of epochs was kept constant when reducing the dataset size. *Y-axis values report the emissions obtained by a single run of the algorithm in the different data-reduction settings, on a log-10 scale.*

## CRediT authorship contribution statement

**Giuseppe Spillo:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Data curation. **Allegra De Filippo:** Writing – review & editing, Writing – original draft, Validation, Supervision, Methodology, Conceptualization. **Cataldo Musto:** Writing – review & editing, Writing – original draft, Validation, Supervision, Methodology, Conceptualization. **Michela Milano:** Writing – review & editing, Conceptualization. **Giovanni Semeraro:** Writing – review & editing, Conceptualization.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## References

[1] P. Rajpurkar, E. Chen, O. Banerjee, E.J. Topol, AI in health and medicine, Nature Med. 28 (1) (2022) 31–38.
[2] S. Reddy, W. Rogers, V.-P. Makinen, E. Coiera, P. Brown, M. Wenzel, E. Weicken, S. Ansari, P. Mathur, A. Casey, et al., Evaluation framework to guide implementation of AI systems into healthcare settings, BMJ Health Care Inform. 28 (1) (2021).
[3] A.M. Musleh Al-Sartawi, K. Hussainey, A. Razzaque, The role of artificial intelligence in sustainable finance, 2022, pp. 1–6.
[4] P. Gogas, T. Papadimitriou, Machine learning in economics and finance, Comput. Econ. 57 (2021) 1–4.
[5] X.-l. Zheng, M.-y. Zhu, Q.-b. Li, C.-c. Chen, Y.-c. Tan, FinBrain: when finance meets AI 2.0, Front. Inf. Technol. Electron. Eng. 20 (7) (2019) 914–924.
[6] B. Wu, B. Gao, W. Xu, H. Wang, Y. Yi, R. Premalatha, Sustainable food smart manufacturing technology, Inf. Process. Manage. 59 (1) (2022) 102754.
[7] S. Bianchi, A. De Filippo, S. Magnani, G. Mosaico, F. Silvestro, Virtus project: a scalable aggregation platform for the intelligent virtual management of distributed energy resources, Energies 14 (12) (2021) 3663.
[8] C.-J. Wu, R. Raghavendra, U. Gupta, B. Acun, N. Ardalani, K. Maeng, G. Chang, F. Aga, J. Huang, C. Bai, et al., Sustainable ai: Environmental implications, challenges and opportunities, Proc. Mach. Learn. Syst. 4 (2022) 795–813.
[9] I.I. Al Barazanchi, D.H. Rasheed, The role of green technologies in mitigating carbon footprints in industrial sectors, Estidamaa 2024 (2024) 30–35.
[10] A.H. Ali, Green AI for sustainability: Leveraging machine learning to drive a circular economy, Babylon. J. Artif. Intell. 2023 (2023) 15–16, http://dx.doi.org/10.58496/BJAI/2023/004, URL: https://mesopotamian.press/journals/index.php/BJAI/article/view/205.
[11] L. Lannelongue, J. Grealey, M. Inouye, Green algorithms: quantifying the carbon footprint of computation, Adv. Sci. 8 (12) (2021) 2100707.

[12] N. Charef, A.B. Mnaouer, M. Aloqaily, O. Bouachir, M. Guizani, Artificial intelligence implication on energy sustainability in Internet of Things: A survey, Inf. Process. Manage. 60 (2) (2023) 103212.

[13] E. Strubell, A. Ganesh, A. McCallum, Energy and policy considerations for modern deep learning research, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, (09) 2020, pp. 13693–13696.

[14] R. Schwartz, J. Dodge, N.A. Smith, O. Etzioni, Green ai, Commun. ACM 63 (12) (2020) 54–63.

[15] G. Chowdhury, An agenda for green information retrieval research, Inf. Process. Manage. 48 (6) (2012) 1067–1077.

[16] A. De Filippo, M. Lombardi, M. Milano, The blind men and the elephant: Integrated offline/online optimization under uncertainty, in: IJCAI, 2021, pp. 4840–4846.

[17] A. De Filippo, M. Lombardi, M. Milano, Integrated offline and online decision making under uncertainty, J. Artificial Intelligence Res. 70 (2021) 77–117.

[18] A. Starke, M. Willemsen, C. Snijders, Promoting energy-efficient behavior by depicting social norms in a recommender interface, ACM Trans. Interact. Intell. Syst. (TiiS) 11 (3–4) (2021) 1–32.

[19] L. Boratto, S. Carta, G. Fenu, R. Saia, Semantics-aware content-based recommender systems: Design and architecture guidelines, Neurocomputing 254 (2017) 79–85.

[20] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, Recommender Systems: an Introduction, Cambridge University Press, 2010.

[21] A. Felfernig, M. Wundara, T.N.T. Tran, S. Polat-Erdeniz, S. Lubos, M. El Mansi, D. Garber, V.-M. Le, Recommender systems for sustainability: overview and research issues, Front. Big Data 6 (2023) 1284511.

[22] B. Purvis, et al., Three pillars of sustainability: in search of conceptual origins, Sustain. Sci. 14 (2019) 681–695.

[23] L. Boratto, G. Fenu, M. Marras, G. Medda, Practical perspectives of consumer fairness in recommendation, Inf. Process. Manage. 60 (2) (2023) 103208.

[24] R. Verdecchia, J. Sallou, L. Cruz, A systematic review of green AI, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 13 (4) (2023) e1507.

[25] A.S. Luccioni, S. Viguier, A.-L. Ligozat, Estimating the carbon footprint of bloom, a 176b parameter language model, J. Mach. Learn. Res. 24 (253) (2023) 1–15.

[26] A.S. Luccioni, A. Hernandez-Garcia, Counting carbon: A survey of factors influencing the emissions of machine learning, 2023, arXiv preprint arXiv:2302.08476.

[27] J. Castaño, S. Martínez-Fernández, X. Franch, J. Bogner, Exploring the carbon footprint of hugging face's ML models: A repository mining study, in: 2023 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, IEEE, 2023, pp. 1–12.

[28] V. Liu, Y. Yin, Green AI: Exploring carbon footprints, mitigation strategies, and trade offs in large language model training, 2024, arXiv preprint arXiv:2404.01157.

[29] D. Patterson, J. Gonzalez, U. Hölzle, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D.R. So, M. Texier, J. Dean, The carbon footprint of ML training will plateau, then shrink, Computer 55 (7) (2022) 18–28.

[30] R. Selvan, N. Bhagwat, L.F. Wolff Anthony, B. Kanding, E.B. Dam, Carbon footprint of selecting and training deep learning models for medical image analysis, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2022, pp. 506–516.

[31] A. Lacoste, A. Luccioni, V. Schmidt, T. Dandres, Quantifying the carbon emissions of machine learning, 2019, arXiv preprint arXiv:1910.09700.

[32] G. Spillo, A. De Filippo, C. Musto, M. Milano, G. Semeraro, 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, 2023, pp. 856–862.

[33] T. Vente, L. Wegmeth, A. Said, J. Beel, From clicks to carbon: the environmental toll of recommender systems, in: Proceedings of the 18th ACM Conference on Recommender Systems, 2024, pp. 580–590.

[34] J. Beel, A. Said, T. Vente, L. Wegmeth, Green recommender systems: A call for attention, in: ACM SIGIR Forum, vol. 58, (2) ACM New York, NY, USA, 2025, pp. 1–5.

[35] A. De Filippo, L. Boratto, G. Spillo, Human-centered and sustainable recommender systems, in: Adjunct Proceedings of the 33rd ACM Conference on User Modeling, Adaptation and Personalization, 2025, pp. 10–12.

[36] D. Stamoulis, E. Cai, D.-C. Juan, D. Marculescu, Hyperpower: Power-and memory-constrained hyper-parameter optimization for neural networks, in: 2018 Design, Automation & Test in Europe Conference & Exhibition, DATE, IEEE, 2018, pp. 19–24.

[37] L.H.P. de Chavannes, M.G.K. Kongsbak, T. Rantzau, L. Derczynski, Hyperparameter power impact in transformer language model training, in: Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing, 2021, pp. 96–118.

[38] A. De Filippo, M. Lombardi, M. Milano, et al., How to tame your anticipatory algorithm, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019, pp. 1071–1077.

[39] A.E. Brownlee, J. Adair, S.O. Haraldsson, J. Jabbo, Exploring the accuracy–energy trade-off in machine learning, in: 2021 IEEE/ACM International Workshop on Genetic Improvement, GI, IEEE, 2021, pp. 11–18.

[40] B. Zhang, A. Davoodi, Y.H. Hu, Exploring energy and accuracy tradeoff in structure simplification of trained deep neural networks, IEEE J. Emerg. Sel. Top. Circuits Syst. 8 (4) (2018) 836–848.

[41] W. Fan, Systematic data selection to mine concept-drifting data streams, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004, pp. 128–137.

[42] Y. Wang, R. Ge, S. Qiu, Energy-aware DNN graph optimization, 2020, arXiv preprint arXiv:2005.05837.

[43] Y. Ji, A. Sun, J. Zhang, C. Li, A re-visit of the popularity baseline in recommender systems, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1749–1752.

[44] R. Verachtert, L. Michiels, B. Goethals, Are we forgetting something? Correctly evaluate a recommender system with an optimal training window, in: Perspectives@ RecSys, 2022.

[45] S.A. Budennyy, V.D. Lazarev, N.N. Zakharenko, A.N. Korovin, O. Plosskaya, D.V. Dimitrov, V. Akhripkin, I. Pavlov, I.V. Oseledets, I.S. Barsola, et al., Eco2ai: carbon emissions tracking of machine learning models as the first step towards sustainable ai, in: Doklady Mathematics, 106, (Suppl 1) Springer, 2022, pp. S118–S128.

[46] G. Spillo, A.G. Valerio, F. Franchini, A. De Filippo, C. Musto, M. Milano, G. Semeraro, Recsys carbonator: Predicting carbon footprint of recommendation system models, in: International Workshop on Recommender Systems for Sustainability and Social Good, Springer, 2024, pp. 98–110.

[47] G. Spillo, A. De Filippo, E. Fontana, M. Milano, G. Semeraro, Training green and sustainable recommendation models: Introducing carbon footprint data into early stopping criteria, in: Proceedings of the 33rd ACM Conference on User Modeling, Adaptation and Personalization, 2025, pp. 341–346.

[48] P. Resnick, H.R. Varian, Recommender systems, Commun. ACM 40 (3) (1997) 56–58.

[49] A. Jameson, M.C. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, L. Chen, Human decision making and recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 611–648.

[50] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.

[51] R. Mehta, K. Rana, A review on matrix factorization techniques in recommender systems, in: 2017 2nd International Conference on Communication Systems, Computing and IT Applications, CSCITA, 2017, pp. 269–274, http://dx.doi.org/10.1109/CSCITA.2017.8066567.

[52] H.-J. Xue, X.-Y. Dai, J. Zhang, S. Huang, J. Chen, Deep matrix factorization models for recommender systems, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI '17, AAAI Press, 2017, pp. 3203–3209.

[53] C. Musto, M.d. Gemmis, P. Lops, F. Narducci, G. Semeraro, Semantics and content-based recommendations, in: Recommender Systems Handbook, Springer, 2022, pp. 251–298.

[54] V.W. Anelli, P. Basile, D. Bridge, T. Di Noia, P. Lops, C. Musto, F. Narducci, M. Zanker, Knowledge-aware and conversational RecSys, in: ACM RECSYS 2018, RecSys '18, ACM, 2018, pp. 521–522, http://dx.doi.org/10.1145/3240323.3240338.

[55] Y. Zhang, Q. Ai, X. Chen, P. Wang, Learning over knowledge-base embeddings for recommendation, 2018, arXiv preprint arXiv:1803.06540.

[56] H. Wang, M. Zhao, X. Xie, W. Li, M. Guo, Knowledge graph convolutional networks for recommender systems, in: The World Wide Web Conference, WWW '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 3307–3313, http://dx.doi.org/10.1145/3308558.3313417.

[57] F. Zhang, N.J. Yuan, D. Lian, X. Xie, W.-Y. Ma, Collaborative knowledge base embedding for recommender systems, in: SIGKDD, KDD '16, 2016, pp. 353–362, http://dx.doi.org/10.1145/2939672.2939673.

[58] G. Spillo, C. Musto, M. Polignano, P. Lops, M. de Gemmis, G. Semeraro, Combining graph neural networks and sentence encoders for knowledge-aware recommendations, in: Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization, UMAP '23, Association for Computing Machinery, New York, NY, USA, 2023, pp. 1–12, http://dx.doi.org/10.1145/3565472.3592965.

[59] W.X. Zhao, S. Mu, Y. Hou, Z. Lin, Y. Chen, X. Pan, K. Li, Y. Lu, H. Wang, C. Tian, Y. Min, Z. Feng, X. Fan, X. Chen, P. Wang, W. Ji, Y. Li, X. Wang, J. Wen, RecBole: Towards a unified, comprehensive and efficient framework for recommendation algorithms, in: CIKM, ACM, 2021, pp. 4653–4664.

[60] L. Xu, Z. Tian, G. Zhang, L. Wang, J. Zhang, B. Zheng, Y. Li, Y. Hou, X. Pan, Y. Chen, W.X. Zhao, X. Chen, J.-R. Wen, Recent advances in RecBole: Extensions with more practical considerations, 2022.

[61] A. Zaharaddeen, N. Muktar, N author-centric framework error minimization in scholarly recommender system (Acfemsr), Babylon. J. Internet Things 2024 (2024) 161–168, http://dx.doi.org/10.58496/BJIoT/2024/018, URL: https://mesopotamian.press/journals/index.php/BJIoT/article/view/645.

[62] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: UAI, UAI '09, 2009, pp. 452–461.

[63] M. Deshpande, G. Karypis, Item-based top-N recommendation algorithms, ACM Trans. Inf. Syst. 22 (1) (2004) 143–177, http://dx.doi.org/10.1145/963770.963776.

[64] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 1067–1077.

[65] D. Liang, R.G. Krishnan, M.D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: Proceedings of the 2018 World Wide Web Conference, WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2018, pp. 689–698, http://dx.doi.org/10.1145/3178876.3186150.

[66] L. Zheng, C.-T. Lu, F. Jiang, J. Zhang, P.S. Yu, Spectral collaborative filtering, in: Proceedings of the 12th ACM Conference on Recommender Systems, 2018, pp. 311–319.

[67] X. Wang, X. He, M. Wang, F. Feng, T.-S. Chua, Neural graph collaborative filtering, in: SIGIR, SIGIR '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 165–174, http://dx.doi.org/10.1145/3331184.3331267.

[68] X. Wang, H. Jin, A. Zhang, X. He, T. Xu, T.-S. Chua, Disentangled graph collaborative filtering, in: SIGIR, SIGIR '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 1001–1010, http://dx.doi.org/10.1145/3397271.3401137.

[69] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, LightGCN: Simplifying and powering graph convolution network for recommendation, in: SIGIR, SIGIR '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 639–648, http://dx.doi.org/10.1145/3397271.3401063.

[70] S. Zhang, H. Tong, J. Xu, R. Maciejewski, Graph convolutional networks: a comprehensive review, Comput. Soc. Netw. 6 (1) (2019) 1–23.

[71] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, Z. Wang, Knowledge-aware graph neural networks with label smoothness regularization for recommender systems, in: ACM SIGKDD, KDD '19, ACM, 2019, pp. 968–977, http://dx.doi.org/10.1145/3292500.3330836.

[72] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, M. Guo, RippleNet: Propagating user preferences on the knowledge graph for recommender systems, in: CIKM, CIKM '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 417–426, http://dx.doi.org/10.1145/3269206.3271739.

[73] D. Pandey, M. Agrawal, J.S. Pandey, Carbon footprint: current methods of estimation, Environ. Monit. Assess. 178 (2011) 135–160.

[74] H. Ritchie, P. Rosado, Electricity mix, 2020, Our World Data, https://ourworldindata.org/electricity-mix.

[75] M. Uddin, Y. Darabidarabkhani, A. Shah, J. Memon, Evaluating power efficient algorithms for efficiency and carbon emissions in cloud data centers: A review, Renew. Sust. Energy Rev. 51 (2015) 1553–1563.

[76] Y. Zheng, D.X. Wang, A survey of recommender systems with multi-objective optimization, Neurocomputing 474 (2022) 141–153.

[77] G. Spillo, A. De Filippo, C. Musto, M. Milano, G. Semeraro, Comparing data reduction strategies for energy-efficient green recommender systems, J. Intell. Inf. Syst. (2025) 1–27.

[78] P. Dutta, A. Kumar, S. Sakici, B. Mensah, Enhancing point-of-interest recommendation systems through multi-modal data integration in location-based social networks: Challenges and future directions, EDRAAK 2025 (2025) 12–18, http://dx.doi.org/10.70470/EDRAAK/2025/003.