Energy Footprint of Data Reduction Strategies for Recommender Systems

Marcell Herbák[1], Oleg Lesota[1], and Antonela Tommasel[1,2]

[1] Johannes Kepler University Linz, Austria
[2] ISISTAN CONICET-UNCPBA Tandil, Argentina

Abstract. Recommender systems are increasingly evaluated not only by recommendation quality, but also by their computational and environmental footprint. This work investigates whether data reduction can lower the energy consumption of recommender systems without disproportionately degrading performance. We compare random, temporal and structure-aware coreset strategies across three representative recommendation algorithms. Energy consumption is measured separately for data preprocessing and model training. Our results show that the sustainability benefits of data reduction depend strongly on both the recommendation model and the data reduction strategy. Reducing the training data can lower energy use while preserving competitive performance, but poorly chosen reductions may substantially harm accuracy or user coverage. Evaluation also shows that data reduction cannot be assumed to be cost-free. The energy required to construct a reduced dataset may offset part of the savings achieved during model training. These findings highlight the need to evaluate data reduction from a broader system-level perspective that jointly considers recommendation quality, user coverage, preprocessing overhead, and training costs.

Keywords: Green AI · Recommender Systems · Data Reduction · Energy Efficiency

1 Introduction

Recommender systems support personalized access to products, media and information across many digital platforms. Their growing scale and complexity, however, also increase their computational and environmental costs [21]. Recommendation models may be trained on millions of interactions, retrained frequently as preferences and catalogs evolve and queried continuously after deployment [16]. Consequently, even modest inefficiencies in training or inference can accumulate into a substantial energy footprint.

Recent work has therefore begun to investigate sustainability–aware recommender systems. For example, Spillo et al. [17] showed that complex recommendation algorithms can require substantially more energy while providing only limited improvements in predictive performance. Follow-up studies explored data reduction as a mitigation strategy [18], finding that smaller training sets can reduce emissions while retaining much of the original recommendation quality. Moreover, Schodl et al. [16] demonstrated that sustainability assessments should not focus exclusively on training, since validation and inference can contribute considerably to the overall footprint of a recommender system. Together, these studies establish that the most accurate model is not necessarily the most desirable one when environmental cost is considered.

Nevertheless, while data reduction offers a promising way to improve this balance, deciding exactly *which* data should be retained remains an open problem. Although

the broader literature on efficiency-oriented data reduction explores a wide range of sophisticated selection methods, sustainability-focused recommender-systems research has so far concentrated mainly on relatively simple strategies, such as random removal of interactions or the retention of only recent observations [18]. Although these methods are inexpensive to apply, they do not explicitly account for the structural importance of individual user-item interactions. Removing observations indiscriminately may change the distribution of collaborative signals in the data, with possible consequences for less active users and long-tail items. Temporal filtering can introduce a different problem. Users whose histories lie mostly before a global cutoff may lose their entire profile, making it impossible to ensure quality recommendations for them.

Coreset methods [12, 20, 9] offer an alternative. Rather than treating all observations as equally informative, a coreset attempts to retain a smaller subset that approximates relevant properties of the full dataset. In the recommendation setting, this structure–aware coreset selection could preserve influential interactions while discarding redundant ones, potentially yielding a more favorable balance between energy consumption and recommendation quality than random or purely temporal reduction. At the same time, computing such subsets may itself require costly matrix decompositions, clustering, gradient calculations or iterative optimization. A data reduction method can therefore lower the cost of the subsequent training run while still increasing the total footprint. Consequently, the effect of data reduction cannot be understood solely through the reduced energy consumption during training. To properly assess the sustainability trade-off, the data reduction rule, its impact on recommendation accuracy and the resulting user coverage, as well as the energy consumed during dataset reduction and model training, must all be considered jointly.

In this work, we present a preliminary evaluation of conventional and structure-aware data reduction strategies for recommender systems. Our evaluation includes user- and item-based random reduction, user and global temporal filtering, coreset SVD-based leverage sampling and cluster-based subset selection. The strategies are evaluated on MovieLens-1M using three recommendation algorithms under a shared leave-one-out split. Although inference is an important component of the broader system footprint, this study focuses on preprocessing and training, the phases directly affected by the construction and use of reduced training datasets.

This work makes two main contributions. First, it provides a comparison of random, temporal, and structure-aware reduction strategies across three recommendation models. Second, it demonstrates the importance of user coverage and selection overhead when assessing whether data reduction constitutes an effective sustainability strategy. More broadly, this work supports the development of sustainability-aware recommender systems in which recommendation quality, coverage, and environmental cost are evaluated jointly rather than in isolation.

2 Related Work

Measuring environmental footprint of recommender systems. Recent research provides a broad view on quantifying and analyzing the environmental footprint of recommender systems. Spillo et al. [17] investigate the trade-off between accuracy and training-induced carbon footprint of several SotA recommenders. Their results show that the substantial increases in carbon emissions connected to training more sophisticated recommenders are often followed by only marginal improvements in accuracy, compared to simpler, less demanding approaches. Vente et al. [21] report a drastic increase in the carbon footprint produced by the recommender systems research between 2013 and 2023. Reproducing experimental pipelines from a number of research papers they confirm that energy-intensive deep learning based models often bring no gains in accuracy over traditional approaches. Their paper calls for careful design of experimental pipelines to minimize unnecessary environmental impact. Schodl et al. [16] argue for more holistic investigation of the environmental footprint produced by recommender

systems. They show that a thoughtful design of the recommendation pipeline, e.g. limiting the number of metrics computed during validation, can help notably decrease associated energy consumption and, consequentially, carbon footprint. Additionally, their break-even analysis reveals that choosing a more energy-intensive model may be justified if it is efficient at inference time, given a sufficient number of inference cycles.

Data reduction in recommender systems. Data reduction has been studied in recommender systems for several purposes. Some works discuss data reduction for user privacy and point out challenges of balancing it with recommendation fairness and accuracy [13, 2, 10]. A large body of research is dedicated to data reduction for improving model efficiency. The works mainly focus on developing coreset building techniques, aiming to preserve properties essential for the recommendation task in a smaller version of the original dataset, while considering more straightforward approaches, such as data sampling, as baselines. Most of this literature targets training efficiency optimization [9, 20, 3], with fewer works aimed at optimizing inference-time computation [8]. Recently, researches have begun investigating data reduction as a means to optimize environmental footprint of recommender systems. Existing studies in this area, however, have mainly considered relatively simple reduction techniques. Spillo et al. [18] investigate random sampling of ratings from user and item perspectives and temporal interaction history splits, with global and user-specific cut-off points. Arabzadeh et al. [1] consider the accuracy-footprint trade-off while sampling different proportions of the training set.

To the best of our knowledge, prior work has not explicitly accounted for the energy required to construct the reduced dataset, nor has it evaluated structure-aware coreset reduction methods in the context of sustainable recommender systems. We address this gap by comparing random and temporal methods with SVD-based coreset strategies, while measuring preprocessing and training energy separately.

3 Methodology

Our experimental methodology is aimed at evaluating data reduction strategies from both environmental and recommendation-quality perspectives[3]. Our analysis is guided by the following research questions:

- **RQ1. Impact on energy consumption.** To what extent do different data reduction strategies affect the energy consumed during data preprocessing and model training?
- **RQ2. Impact on recommendation quality.** How do different data reduction strategies affect the recommendation performance of different recommendation algorithms?
- **RQ3. Energy-quality trade-off.** What trade-offs between energy consumption and recommendation quality emerge when data reduction is applied, and which model–reduction combinations offer the most favorable balance?

Dataset and Experimental Split. We use MovieLens-1M [7], which contains 1,000,209 ratings provided by 6,040 users for 3,706 movies. MovieLens-1M provides a practical starting point because it is a well-established and publicly available recommendation benchmark, enabling reproducible experiments and comparison with prior work. Its timestamped interactions support a realistic chronological evaluation, while its scale is large enough to expose meaningful differences in computational cost without making repeated energy measurements impractical. We treat each observed rating as a positive user–item interaction and do not use its numerical rating value.

[3] Code is available in the companion repository https://github.com/herbakmarcell/herbak-recsogood2026.

We perform a chronological leave-one-out split once on the complete, unreduced dataset and reuse it across all reduction strategies and recommendation models. This ensures that every experimental configuration is evaluated using the same validation and test interactions. For each user, the most recent interaction is assigned to the test set, the second most recent to the validation set, and all remaining interactions constitute the training pool. Data reduction is applied exclusively to this training pool, leaving the validation and test sets unchanged across reduction strategies and preventing information leakage.

Data Reduction Strategies. We evaluate eleven reduction strategies plus the unreduced baseline in the main comparison.

- **Baseline**. The complete unreduced training pool.
- **User-based (**-20%**)**. Uniformly random removal of 20% of the interactions associated with each user, following the random reduction family considered in [18].
- **Item-based (**-20%**)**. Uniformly random removal of 20% of the interactions associated with each item, following [18].
- **User-temporal (last** 300**)**. Retention of each user's 300 most recent interactions, following the user-level temporal strategy considered in [18].
- **Global-temporal (Sep 2000)**. Retention of training interactions occurring on or after a fixed global cutoff date, following the global temporal strategy considered in [18].
- **Global-temporal min-**k **(Sep 2000)**.This variant applies the same global temporal cutoff while ensuring that each user retains at least three training interactions. It allows us to separate genuine ranking degradation from losses caused by users becoming unevaluable because either the user or the held-out item is not represented in the reduced training data.
- **Coreset-Leverage (**$-20\%/-30\%$**)**. A coreset is a carefully selected subset of the original data that aims to preserve the information most relevant to the learning task, allowing a model trained on the subset to approximate the behavior of one trained on the full dataset. In our implementation, all coreset strategies first embed the user-item interaction matrix into a lower-dimensional latent space using Truncated SVD (k=50). In this strategy, interactions are then sampled without replacement with probability proportional to their leverage scores computed from these singular vectors. This aims to retain interactions that contribute most strongly to the dataset's low-rank structure, approximately preserving its spectral norm [4].
- **Coreset-Cluster (**$-20\%/-30\%$**)**. Interactions are embedded in the same SVD latent space by summing the corresponding user and item embeddings, and then partitioned into 500 K-Means clusters. Within each cluster, interactions are sorted by distance to the centroid, and those nearest to the center are retained to form a representative subset of average behavior [5].
- **Coreset-Cluster-Outlier (**$-20\%/-30\%$**)**. This strategy applies the same SVD embedding and K-Means clustering as Coreset-Cluster but reverses the selection rule. Instead of retaining the nearest points, it retains the interactions farthest from each cluster centroid. This tests whether atypical interactions carry complementary collaborative signal that is not represented by centroid-nearest observations.

Recommendation Models. For each data reduction strategy, we train three recommendation models using `Cornac` [15]. `BPR` [14] is configured with a latent dimension of k=64, 50 training epochs, a learning rate of 10^{-3} and ℓ_2 regularization of 10^{-4}. `MultiDAE` [11] is implemented as Cornac's VAECF model with β=0, using k=64, a hidden-layer size of 600, 50 epochs and a batch size of 256. Finally, `EASE` [19] is a closed-form linear item–item model with regularization parameter λ=500.

We adopt `Cornac` rather than RecBole, which has been used in several Green RecSys studies, to broaden the empirical evidence beyond a single recommendation framework. Cornac provides a lightweight and transparent experimental interface, facilitating the consistent integration and evaluation of the selected models and data reduction strategies. Since recommendation libraries may differ in implementations, default hyper-parameters, negative-sampling procedures and evaluation bookkeeping, absolute energy and performance values should not be compared directly with RecBole-based results. Instead, our experiments provide complementary evidence on whether previously observed sustainability patterns also hold within a different implementation ecosystem.

Hardware and Energy Measurement. All experiments were conducted on a single workstation equipped with an AMD Ryzen 9 5900X 12-core CPU (24 threads), 32 GB RAM and an NVIDIA GeForce RTX 4060 GPU, running Windows 11 and Python 3.12.7. The three recommendation models used in the study are CPU-bound, consequently, the GPU was not involved in model training or inference.

Energy consumption was estimated using CodeCarbon 3.2.7 [6] in `machine` tracking mode. Measurements were collected separately for two phases of each strategy–model configuration: *preprocessing* (corresponding to the application of the data reduction strategy), and *training*. For operations shorter than CodeCarbon's power-sampling interval, which commonly occurred for inexpensive reduction strategies, energy was estimated from wall-clock duration using a fixed CPU Thermal Design Power of 65 W. We report the raw estimated energy consumption rather than converting it into CO_2-equivalent emissions, thereby avoiding assumptions about the carbon intensity of the electricity supply [16].

These measurements should be interpreted as estimates of computational footprint rather than precise device-level energy consumption. First, CodeCarbon's `machine` mode attributes the estimated consumption of the entire workstation to the running experiment and therefore does not isolate background processes. Second, 93% of the estimated energy was attributed to RAM through CodeCarbon's constant 20 W capacity-based model rather than direct measurement. As a result, the reported values are strongly correlated with wall-clock duration and primarily reflect differences in computational overhead across experimental configurations.

Evaluation Protocol. Recommendation quality is evaluated using sampled-negative Recall@10 and nDCG@10. For each test user, the held-out interaction is ranked against 100 items sampled uniformly from the corresponding model's training-item set, excluding items previously observed by that user. As some reduction strategies remove items from the training data, the candidate set may differ across configurations, and the resulting ranking tasks are therefore not perfectly identical. Moreover, users for whom either the user profile or the held-out item is absent from the reduced training data cannot be scored directly. To preserve comparability, these users are assigned a score of zero, and all metrics are averaged over the same fixed set of test users.

4 Results

4.1 RQ1. Impact on energy consumption

We first examine how the different data reduction strategies affect the size and composition of the training data. As shown in Table 1, the random and −20% coreset strategies retain approximately 800,000 ratings, while the −30% coreset variants retain approximately 700,000 ratings. User-temporal reduction removes 21.5% of the ratings, whereas the global temporal cutoff produces the largest reduction, removing 40.6%. The strategies also differ in their effects on users and items. Most preserve all 6,040

Table 1. Statistics of the leave-one-out training pool after applying each data reduction strategy. The table reports the retained numbers of users, items, and interactions, together with the percentage reduction in interactions relative to the unreduced training-pool baseline.

Strategy	Users	Items	Interactions	Δ Ratings	Evaluable users (%)
Baseline	6,040	3,706	988,129	-	99.99%
User-based (−20%)	6,040	3,677	792,921	−19.75%	99.99%
Item-based (−20%)	6,040	3,706	791,977	−19.85%	99.99%
User-temporal (last 300)	6,040	3,671	774,591	−21.61%	99.99%
Global-temporal (Sep 2000)	3,912	3,659	586,646	−40.63%	64.17%
Global-temporal min-k (Sep 2000)	6,040	3,660	593,177	−39.96%	99.99%
Coreset-Leverage (−20%)	6,040	3,623	790,503	−20.00%	99.99%
Coreset-Leverage (−30%)	6,040	3,583	691,690	−30.00%	99.99%
Coreset-Cluster (−20%)	6,040	3,696	790,502	−20%	99.99%
Coreset-Cluster (−30%)	6,040	3,691	691,669	−30.00%	99.99%
Coreset-Cluster-Outlier (−20%)	6,040	3,668	790,503	−20.00%	99.99%
Coreset-Cluster-Outlier (−30%)	6,040	3,638	691,665	−30.00%	99.99%

users, but the unprotected global-temporal cutoff removes the complete history of 2,128 users whose interactions precede September 2000. Accordingly, all strategies retain an evaluable-user rate of approximately 99.9%, except Global-temporal (Sep. 2000), for which only 64.2% of test users remain evaluable. Global-temporal min-k (Sep. 2000) restores all users by retaining only 6,531 additional ratings, while still reducing the dataset by approximately 40%. Strategies with equivalent rating-reduction targets also preserve the item catalog differently. For example, at −20%, Coreset-Cluster reduction retains 3,696 items, compared with 3,623 for Coreset-Leverage. Thus, the percentage of removed ratings alone does not fully characterize the effects of a reduction strategy.

Figure 1 reports the energy consumed by data processing and a model-training run for each strategy–model combination. The stacked bars distinguish the energy required to load and reduce the data from the energy required to train the recommender. The results show that data reduction cannot be assumed to be cost-free or necessarily energy-saving. Its overall benefit depends on whether the decrease in training energy compensates for the cost of constructing the reduced dataset.

For BPR, all user- and item-random, temporal, and Coreset-Leverage strategies consume less total energy than the full-data baseline. Global-temporal reduction produces the largest saving, decreasing total energy by 40.2%. For Global-temporal, applying the minimum-history safeguard increases preprocessing energy, but the resulting configuration still consumes 27.6% less than the baseline. The remaining user- and item- random, user-temporal, and Coreset-Leverage strategies provide savings ranging from 9.0% to 20.2%.

Results are less uniform for MultiDAE. Global-temporal reduction decreases total energy by approximately 37.06%, while user-based and Coreset-Leverage at −30% reductions provide savings of approximately 19.11% and 18.63%, respectively. Item-based and user-temporal reduction yield smaller savings of approximately 6.07% and 9.25%, while the Global-temporal with minimum-history safeguard reduces total energy by only 2.66%. By contrast, Coreset-Leverage at −20% consumes approximately 0.91% more energy than the baseline, indicating that a reduction in dataset size does not necessarily translate into an immediate reduction in total energy.

For EASE, whose full-data training cost is substantially lower, preprocessing constitutes a larger share of the total energy consumption. User-temporal and global-temporal reduction decrease total energy by approximately 20.12% and 14.26%, respectively, while user-based reduction is approximately equivalent to the baseline, with a small increase of 2.59%. Item-based reduction, the global-temporal with minimum-history temporal variant, and both Coreset-Leverage strategies consume more energy than training EASE on the full dataset. Consequently, even a relatively small prepro-

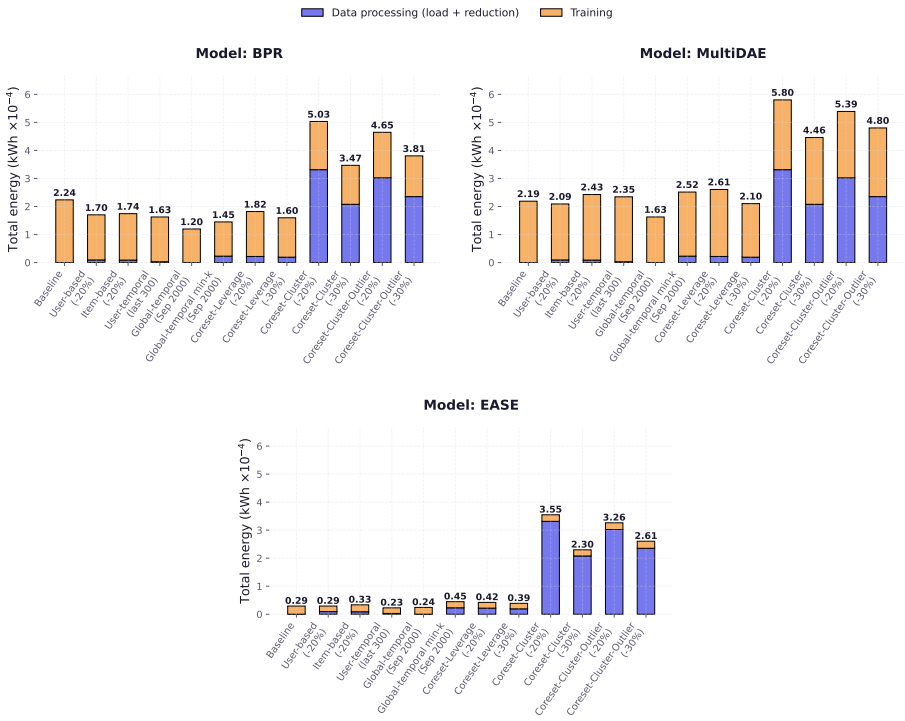

Fig. 1. Energy consumption of data processing and a model-training run for each reduction strategy across the three recommendation models. Stacked bars separate the energy required to load and reduce the data from the energy consumed during training.

cessing overhead can offset the training savings of an already efficient recommendation model.

Coreset-Cluster strategies show the clearest effect of reduction overhead, with data processing dominating their total energy consumption. They increase the total energy consumed by approximately 73.34%–151.53% for BPR and 72.59%–124.57% for MultiDAE. Their relative impact is even greater for EASE, for which total energy consumption is approximately 8.1 to 12.5 times that of the baseline. Coreset-Cluster-Outlier, which retains cluster outliers rather than centroid-nearest interactions, does not eliminate this overhead and produces similarly high energy requirements.

Overall, results answer **RQ1** by showing that *the impact of data reduction on energy consumption is jointly determined by the selection procedure and the recommendation model*. Inexpensive random and temporal strategies can translate reduced dataset size into lower total energy, while more computationally demanding coreset construction may offset or exceed the savings obtained during training. This is particularly important for efficient recommendation models, for which preprocessing can represent a substantial portion of the overall footprint.

4.2 RQ2. Impact on recommendation quality

Table 2 reports Recall@10 and nDCG@10 for each reduction strategy and recommendation model. Overall, the effect of data reduction is broadly consistent across model families. Coreset-Leverage preserves recommendation quality most effectively, whereas item-based random reduction causes the largest degradation. Nevertheless, the magnitude of the effect varies by model and reduction level, indicating that the number of retained interactions alone does not determine recommendation performance.

Table 2. Training energy and recommendation quality for each strategy–model configuration. Δ Train reports the percentage change in training energy relative to the corresponding full-data baseline, while Δ Total includes both preprocessing and a training run. Negative values indicate energy savings and positive values indicate increased consumption. An asterisk marks a statistically significant difference in Recall@10 or nDCG@10 relative to the baseline according to a paired Wilcoxon signed-rank test ($p < 0.05$).

Model	Strategy	Train (kWh)	Δ Train (%)	Δ Total (%)	Recall@10	nDCG@10
BPR	Baseline	0.00020	–	–	0.3300	0.1693
	User-based (−20%)	0.00016	−18.78	−14.92	0.3114	0.1543*
	Item-based (−20%)	0.00017	−16.37	−12.81	0.1286*	0.0617*
	User-temporal (last 300)	0.00016	−19.38	−18.75	0.2800*	0.1399*
	Global-temporal (Sep. 2000)	0.00012	−39.67	−40.15	0.2308*	0.1178*
	Global-temporal min-k (Sep. 2000)	0.00012	−38.39	−27.56	0.3119	0.1583*
	Coreset-Leverage (−20%)	0.00016	−19.10	−9.04	0.3273*	0.1662*
	Coreset-Leverage (−30%)	0.00014	−29.07	−20.19	0.3232	0.1557*
	Coreset-Cluster (−20%)	0.00017	−13.17	+151.53	0.3040	0.1543*
	Coreset-Cluster (−30%)	0.00014	−29.84	+73.34	0.3010	0.1531*
	Coreset-Cluster-Outlier (−20%)	0.00016	−18.00	+132.23	0.3172	0.1580*
	Coreset-Cluster-Outlier (−30%)	0.00015	−26.62	+90.21	0.2950	0.1441*
MultiDAE	Baseline	0.00026	–	–	0.2962	0.1686
	User-based (−20%)	0.00020	−22.13	−19.11	0.2651*	0.1439*
	Item-based (−20%)	0.00023	−8.77	−6.07	0.1331*	0.0682*
	User-temporal (last 300)	0.00023	−9.67	−9.25	0.2570*	0.1495*
	Global-temporal (Sep. 2000)	0.00016	−36.66	−37.06	0.1944*	0.1079*
	Global-temporal min-k (Sep. 2000)	0.00023	−10.84	−2.67	0.2821	0.1507*
	Coreset-Leverage (−20%)	0.00024	−6.79	+0.91	0.3056*	0.1693*
	Coreset-Leverage (−30%)	0.00019	−25.48	−18.63	0.2957	0.1620*
	Coreset-Cluster (−20%)	0.00025	−2.90	+124.57	0.2613*	0.1447*
	Coreset-Cluster (−30%)	0.00024	−7.13	+72.59	0.2530*	0.1417*
	Coreset-Cluster-Outlier (−20%)	0.00024	−7.65	+108.61	0.2795	0.1523*
	Coreset-Cluster-Outlier (−30%)	0.00024	−4.47	+85.83	0.2702	0.1442*
EASE	Baseline	$2.6e{-5}$	–	–	0.3305	0.1831
	User-based (−20%)	$2.0e{-5}$	−25.00	+2.59	0.3152	0.1607*
	Item-based (−20%)	$2.5e{-5}$	−6.62	+17.71	0.1356*	0.0687*
	User-temporal (last 300)	$2.0e{-5}$	−24.93	−20.12	0.2820*	0.1496*
	Global-temporal (Sep. 2000)	$2.4e{-5}$	−8.71	−14.26	0.2329*	0.1292*
	Global-temporal min-k (Sep. 2000)	$2.2e{-5}$	−17.30	+57.38	0.3051	0.1664*
	Coreset-Leverage (−20%)	$2.0e{-5}$	−22.55	+48.49	0.3341*	0.1769*
	Coreset-Leverage (−30%)	$2.0e{-5}$	−25.58	+36.70	0.3132	0.1626*
	Coreset-Cluster (−20%)	$2.3e{-5}$	−11.32	+1147.78	0.3094	0.1654*
	Coreset-Cluster (−30%)	$2.2e{-5}$	−17.15	+708.17	0.2992	0.1578*
	Coreset-Cluster-Outlier (−20%)	$2.4e{-5}$	−9.59	+1047.21	0.3136	0.1656*
	Coreset-Cluster-Outlier (−30%)	$2.5e{-5}$	−4.46	+816.27	0.2801	0.1418*

Among the simple strategies, user-based random reduction produces relatively moderate losses. Recall@10 decreases by 4.62% for EASE, 5.63% for BPR and 10.49% for MultiDAE relative to their respective baselines. The corresponding nDCG@10 reductions range from 8.9% to 14.7%. By contrast, item-based reduction removes approximately the same proportion of ratings, but decreases Recall@10 by 55.06%–61.03% and nDCG@10 by 59.54%–63.55%. These differences are statistically significant. This suggests that uniformly removing interactions from each item damages the collaborative signal throughout the user population, rather than affecting only users with sparse histories.

User-temporal reduction also leads to consistent performance degradation, reducing Recall@10 by approximately 13.23%–15.15% across the three models. Differences are statistically significant. Global-temporal cutoff produces larger losses of approximately 29.53%–34.36% in Recall@10. However, this result is strongly influenced by the removal of the complete training histories of 2,128 users. Introducing the minimum-history safeguard substantially narrows the Recall@10 gap to −4.76% for MultiDAE, −5.48% for BPR and −7.68% for EASE. nDCG@10 similarly improves, showing that much of the degradation observed for the original global cutoff is attributable to coverage loss rather than to poorer rankings for users who remain evaluable.

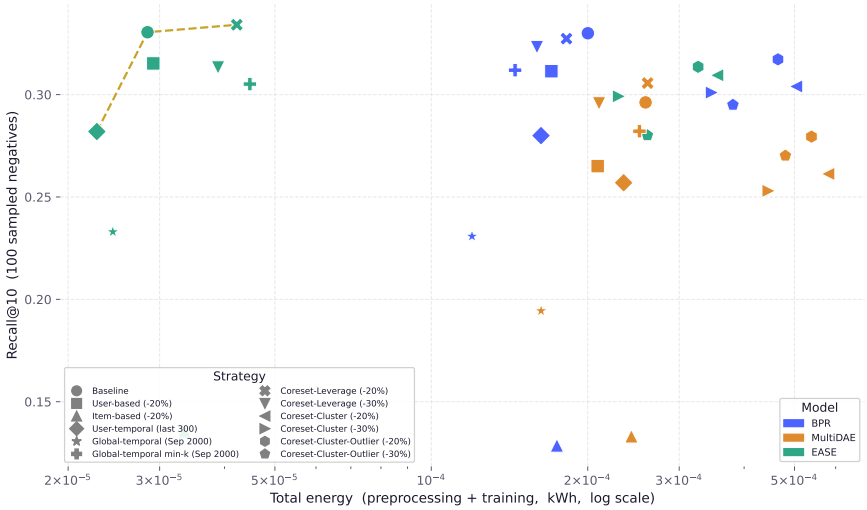

Fig. 2. Energy–quality trade-off across recommendation models and data reduction strategies. The x-axis reports total energy consumption for preprocessing and a single training run on a logarithmic scale, while the y-axis reports Recall@10. Colors identify recommendation models, marker shapes identify reduction strategies, and the dashed line indicates the global Pareto frontier. Configurations closer to the upper-left provide a more favorable balance.

Coreset-Leverage preserves the baseline most closely. At −20%, Recall@10 differs from the baseline by only −0.81% for BPR, +3.17% for MultiDAE and +1.09% for EASE. nDCG@10 remains within 3.36% of the baseline for all three models. Even at −30%, Recall@10 remains within −5.23% of full-data performance. These results suggest that leverage-score sampling preserves the collaborative structure of the original data more effectively than the other reduction strategies considered.

Coreset-Cluster is less effective than Coreset-Leverage. Coreset-Cluster variants reduce Recall@10 by approximately 6.38%–11.78% at −20% and by 8.78%–14.58% at −30%. Coreset-Cluster-Outlier improves the −20% results for all three models, but the pattern is not consistent at −30%, as Coreset-Cluster-Outlier improves MultiDAE while reducing performance for BPR and EASE. Reversing the cluster-selection rule therefore provides no systematic advantage.

Overall, **RQ2** shows that *recommendation quality depends more strongly on which interactions are removed than on the reduction rate alone*. Coreset-Leverage can remove up to 30% of the ratings while largely preserving recommendation performance, whereas an equally sized item-based reduction severely degrades both Recall@10 and nDCG@10. Global-temporal reduction can also remain competitive when minimum user coverage is explicitly protected.

4.3 RQ3. Energy-quality trade-off

Figure 2 jointly compares recommendation quality and the total energy required for data preprocessing and a model-training run. As improving one objective may come at the expense of the other, we use Pareto analysis to identify configurations that offer the best attainable compromises without assigning an arbitrary weight to either energy consumption or recommendation quality. A configuration is Pareto-optimal when no alternative achieves equal or higher recommendation quality while consuming equal or less energy.

Results indicate that model choice has a stronger influence on the global energy–quality trade-off than the reduction strategy alone. For Recall@10, every configuration on the global Pareto frontier shown in Figure 2 uses EASE. An analogous analysis based on nDCG@10 leads to the same model-level conclusion. The baseline full-data EASE model achieves the highest nDCG@10 in the study and nearly the highest Recall@10, while consuming substantially less energy than BPR and MultiDAE. Thus, selecting an inherently efficient recommendation model can yield a greater benefit than reducing the training data of a more computationally demanding model.

For Recall@10, the global frontier contains three EASE configurations. User-temporal reduction is the lowest-energy solution, reducing total energy by approximately 20.1% relative to the EASE baseline, but also decreasing Recall@10 by 14.7%. At the system level, the full-data EASE configuration offers a particularly strong balance between energy consumption and recommendation quality, while the other EASE configurations on the Pareto frontier represent alternatives that prioritize either lower energy or slightly higher Recall@10. Coreset-Leverage at -20% obtains the highest observed Recall@10, but improves it by only 1.1% while increasing total energy by approximately 48.5% because of its preprocessing overhead.

Within the more computationally demanding model families, leverage sampling provides useful compromises. For BPR, Coreset-Leverage at -20% reduces total energy by approximately 9.0% while decreasing Recall@10 and nDCG@10 by only 0.8% and 1.8%, respectively. At -30%, it reduces energy by 20.2%, with a 2.1% Recall@10 loss but a larger 8.0% decrease in nDCG@10. For MultiDAE, Coreset-Leverage at -30% offers the most favorable within-model balance, lowering total energy by 18.6% while preserving Recall@10 almost entirely and reducing nDCG@10 by 3.9%.

The global-temporal configurations illustrate why energy and accuracy values must also be interpreted alongside coverage. The original cutoff has the lowest energy consumption within the BPR and MultiDAE families, but only 64.2% of test users remain evaluable. More than one third of the users therefore receive a score of zero because their complete training history was removed, rather than because the model failed to rank their held-out item correctly. The minimum-history safeguard variant restores user coverage and provides a more meaningful compromise. For BPR, it reduces total energy by 27.6% while decreasing Recall@10 by 5.5% and nDCG@10 by 6.5%.

By contrast, item-based random reduction does not provide a favorable trade-off. Although it lowers training energy, it reduces Recall@10 by more than half across all models. This pattern is consistent with item-based removal potentially weakening the limited collaborative signal available for sparse and long-tail items. The Coreset-Cluster strategies are also dominated, as their preprocessing overhead substantially increases total energy while their recommendation quality remains below both the baseline and leverage-based alternatives.

Overall, **RQ3** shows that *reducing the size of the training data does not automatically improve the energy–quality balance*. The most favorable configurations combine an efficient recommendation model with a data reduction strategy that preserves informative interactions at low construction cost. Coreset-Leverage offers the strongest within-model trade-offs for BPR and MultiDAE, but at the system level the full-data EASE configuration provides the most favorable overall balance between energy consumption and recommendation quality.

5 Conclusions

This work investigated how random, temporal, and coreset structure-aware data reduction strategies affect the energy consumption and recommendation quality of recommender systems. Results show that model choice has a stronger effect on the global energy–quality balance than data reduction alone. Coreset-Leverage preserves full-data recommendation quality more closely than the other reduction strategies, whereas item-based reduction causes substantial performance losses. Global-temporal reduction can

provide meaningful energy savings, but requires a minimum-history safeguard to avoid excluding users. Finally, the high construction cost of the cluster-based coreset strategies demonstrates that data reduction cannot be assumed to be cost-free. Overall, sustainable data reduction requires more than removing interactions. It requires jointly considering which data are retained, which model is trained, whether user coverage remains sufficient, and whether the training-energy savings outweigh the cost of constructing the reduced dataset.

Several limitations define directions for future work. First, the present results rely primarily on single-seed evaluations and are therefore sensitive to run-to-run variations[4]. Future work should conduct multi-seed experiments and report confidence intervals to determine whether the observed Pareto improvements and differences between reduction strategies are statistically reliable. Second, the sampled-negative pool is currently drawn from each strategy's retained item set. Since some reductions remove more items than others, the resulting ranking tasks are not perfectly comparable. Future evaluations should use a shared global negative-item pool so that all strategies are assessed under the same ranking conditions, complemented with a full-ranking approach. Third, energy measurements are based on CodeCarbon in machine-tracking mode and are strongly influenced by runtime, Thermal Design Power assumptions, and constant estimates of RAM consumption. Consequently, the reported values should be interpreted as estimated energy footprints rather than precise device-level measurements. Future studies should complement software-based tracking with hardware-level sensors or with external physical power meters.

Finally, the empirical comparison should be broadened across datasets, hardware environments, recommendation models and recommendation frameworks. While much of the existing Green RecSys literature relies on RecBole, our use of Cornac expands the evidence beyond a single software ecosystem and helps assess whether previously observed patterns generalize across implementations. Future work should conduct a controlled cross-framework comparison between Cornac, RecBole and other recommendation libraries (e.g., Elliot), aligning datasets, splits, hyper-parameters, evaluation protocols and hardware conditions. Such an analysis would help distinguish general properties of recommendation algorithms from effects introduced by particular implementations and library defaults. Further extensions should quantify the energy cost of hyper-parameter tuning and investigate how data reduction affects non-accuracy dimensions such as recommendation diversity, catalog coverage and popularity bias.

References

1. Arabzadeh, A., Vente, T., Beel, J.: Green recommender systems: Optimizing dataset size for energy-efficient algorithm performance. In: Recommender Systems for Sustainability and Social Good - First International Workshop, RecSoGood 2024, Bari, Italy, October 18, 2024, Proceedings. Communications in Computer and Information Science, vol. 2470, pp. 73–82. Springer (2024)
2. Bufi, S., Paparella, V., Anelli, V.W., Noia, T.D.: Legal but unfair: Auditing the impact of data minimization on fairness and accuracy trade-off in recommender systems. In: Proceedings of the 33rd ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2025, New York City, NY, USA, June 16-19, 2025. pp. 114–123. ACM (2025)
3. Choi, S.H., Jeong, Y., Jeong, M.K.: A hybrid recommendation method with reduced data for large-scale application. IEEE Trans. Syst. Man Cybern. Part C **40**(5), 557–566 (2010)

[4] As a preliminary robustness check, we repeated the evaluation over five seeds. Across all configurations, the standard deviation reached at most 0.0176 for Recall@10 and 0.0095 for nDCG@10. These results suggest that the main trends are stable, although the relative ordering of configurations separated by only small performance differences may vary across seeds.

4. Cohen, M.B., Elder, S., Musco, C., Musco, C., Persu, M.: Dimensionality reduction for k-means clustering and low rank approximation. In: Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015. pp. 163–172. ACM (2015)
5. Cohen-Addad, V., Saulpic, D., Schwiegelshohn, C.: A new coreset framework for clustering. In: STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021. pp. 169–182. ACM (2021)
6. Courty, B., Schmidt, V., Goyal-Kamal, MarionCoutarel, Feld, B., Lecourt, J., LiamConnell, SabAmine, inimaz, supatomic, Léval, M., Blanche, L., Cruveiller, A., ouminasara, Zhao, F., Joshi, A., Bogroff, A., Saboni, A., de Lavoreille, H., Laskaris, N., Abati, E., Blank, D., Wang, Z., Catovic, A., alencon, Stęchły, M., Bauer, C., Lucas-Otavio, JPW, MinervaBooks: mlco2/codecarbon: v2.4.1 (May 2024)
7. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM Trans. Interact. Intell. Syst. **5**(4), 19:1–19:19 (2016)
8. Jiang, J., Chen, P.H., Hsieh, C., Wang, W.: Clustering and constructing user coresets to accelerate large-scale top-k recommender systems. In: WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020. pp. 2177–2187. ACM / IW3C2 (2020)
9. Ju, Z., Du, H., Tragos, E., Hurley, N., Lawlor, A.: Exploring coresets for efficient training and consistent evaluation of recommender systems. In: Proceedings of the 18th ACM Conference on Recommender Systems. p. 1152–1157. RecSys '24, Association for Computing Machinery, New York, NY, USA (2024)
10. Leysen, J., Favier, M., Goethals, B.: What data is really necessary? A feasibility study of inference data minimization for recommender systems. In: Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025, Seoul, Republic of Korea, November 10-14, 2025. pp. 1519–1529. ACM (2025)
11. Liang, D., Krishnan, R.G., Hoffman, M.D., Jebara, T.: Variational autoencoders for collaborative filtering. In: Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018. pp. 689–698. ACM (2018)
12. Mirzasoleiman, B., Bilmes, J.A., Leskovec, J.: Coresets for data-efficient training of machine learning models. In: Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event. Proceedings of Machine Learning Research, vol. 119, pp. 6950–6960. PMLR (2020)
13. Niu, X., Rahman, R., Wu, X., Fu, Z., Xu, D., Qiu, R.: Leveraging uncertainty quantification for reducing data for recommender systems. In: 2023 IEEE International Conference on Big Data (BigData). pp. 352–359 (2023)
14. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: bayesian personalized ranking from implicit feedback. CoRR **abs/1205.2618** (2012)
15. Salah, A., Truong, Q., Lauw, H.W.: Cornac: A comparative framework for multimodal recommender systems. J. Mach. Learn. Res. **21**, 95:1–95:5 (2020)
16. Schodl, J., Lesota, O., Tommasel, A., Schedl, M.: Investigating carbon footprint of recommender systems beyond training time. In: Proceedings of the Nineteenth ACM Conference on Recommender Systems, RecSys 2025, Prague, Czech Republic, September 22-26, 2025. pp. 1206–1211. ACM (2025)
17. Spillo, G., Filippo, A.D., Musto, C., Milano, M., Semeraro, G.: 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, RecSys 2023, Singapore, Singapore, September 18-22, 2023. pp. 856–862. ACM (2023)
18. Spillo, G., Filippo, A.D., Musto, C., Milano, M., Semeraro, G.: Comparing data reduction strategies for energy-efficient green recommender systems. J. Intell. Inf. Syst. **63**(6), 1837–1863 (2025)

19. Steck, H.: Embarrassingly shallow autoencoders for sparse data. In: The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019. pp. 3251–3257. ACM (2019)
20. Tran, H.V., Chen, T., Wen, H., Nguyen, Q.V.H., Cui, B., Yin, H.: Efficient content-based recommendation model training via noise-aware coreset selection. In: Proceedings of the ACM Web Conference 2026, WWW 2026, Dubai, United Arab Emirates, originally scheduled for April 13-17, 2026, rescheduled for June 29 - July 3, 2026. pp. 6841–6852. ACM (2026)
21. Vente, T., Wegmeth, L., Said, A., Beel, J.: From clicks to carbon: The environmental toll of recommender systems. In: Proceedings of the 18th ACM Conference on Recommender Systems, Italy, October 14-18, 2024 (2024)