Exploiting Surrogate Submodular and Cost-Effective Lazy Forward Algorithms for Calibrated Recommendations

Diego Corrêa da Silva[1], **Joel Machado Pires**[1], **Frederico Araújo Durão**[1]

[1]Institute of Computing – Federal University of Bahia – Salvador, Bahia – Brazil
Avenida Milton Santos, s/n – Campus de Ondina, PAF 2 – Salvador – Bahia, 40.170-110

{diego.correa, joel.pires, fdurao}@ufba.br

Abstract. *Recommendation systems are tools to suggest items that might interest users. These systems rely on the user's preference history to generate a list of suggestions most similar to items in the user's history, aiming for better accuracy and minimal error. Pursuing higher accuracy can lead to side effects such as overspecialization, reduced diversity, and imbalances in categories, genres, or niches. Thus, this work explores algorithms for selecting the items that form a calibrated recommendation list. The hypothesis is that calibration can positively contribute to fairer recommendations based on user preferences. We introduce a variation of an existing algorithm, which performs better than the baselines in a commonly used dataset in two different divergence measurements.*

1. Introduction

Recommender systems (RS) have long been designed to retrieve the top-n most relevant items for users based on their preferences. Traditionally, recommendation algorithms have prioritized accuracy, ensuring that the recommended items closely align with user preferences. However, this focus on accuracy has revealed significant drawbacks, such as unbalanced genre distributions. These challenges have led researchers to explore alternative approaches beyond accuracy to incorporate fairness, diversity, and novelty. Recent research has also explored different contextual factors in RS. For example, geographic embeddings have been incorporated into POI recommendation models to better capture user preferences based on spatial and contextual influences [Leite et al. 2024]. Similarly, our work focuses on calibration as an essential property to ensure recommendations align with user preferences while addressing issues such as fairness and diversity. Additionally, fairness has emerged as a critical concern in machine learning, with studies highlighting the risks of biased models leading to discrimination. Approaches such as fairness-aware machine learning techniques aim to mitigate bias, ensuring that recommendation systems provide equitable outcomes across different user groups [Sena and Machado 2024, Aragão et al. 2024].

Personalized recommendations are provided as an ordered list of items, predicted based on user preferences. Two primary approaches dominate the recommendation landscape: content-based filtering (CBF) and collaborative filtering (CF). CBF suggests items similar to those previously preferred by a user, leveraging item descriptions and user profiles to determine relevance. Conversely, CF recommends items liked by other users with similar tastes, relying on collective user interactions to make predictions [da Silva et al. 2021, Steck 2018].

A particularly pressing issue in recommendation systems is miscalibration, where the distribution of recommended items diverges from a user's original preference distribution. Calibrated RS seek to mitigate this issue by ensuring the proportion of different categories (such as genres or tags) in the recommendation list aligns with the user's historical preferences. Miscalibration occurs when the system fails to respect this distribution, potentially leading to biased recommendations and reduced user satisfaction. For instance, if a user listens to 60% Samba, 30% Rock, and 10% K-Pop, an ideally calibrated system should generate recommendations reflecting similar proportions [Lin et al. 2020].

Similarly to previous works, we focus on calibrating recommendation systems as a post-processing of the recommendation list. However, generating an optimally calibrated recommendation list requires evaluating all possible permutations of m candidate items in n positions, making it an NP-hard problem. Among the methods proposed, submodular optimization has emerged as a promising approach. Specifically, surrogate submodular functions have been explored as efficient strategies to optimize recommendation sets while maintaining computational efficiency [Steck 2018, Kaya and Bridge 2019, da Silva et al. 2025]. In this paper, we examine two algorithms, the surrogate and the cost effective lazy forward (CELF), seeking to enhance calibrated recommendations, particularly in genre calibration.

To achieve this, we present a detailed exploration of surrogate submodular algorithms for calibration in RS, including the formal definitions of data models, probability distributions for genre calibration, and divergence measurements. We outline the selection criteria used for optimizing recommendations and examine two algorithms: the original surrogate submodular method and a variation of CELF that we propose. Our methodology first fits a matrix factorization (MF) model, evaluating it using the mean absolute error (MAE) metric to predict user explicit feedback. The recommendation list is then generated based on these predictions, followed by a post-processing step to apply calibration, which is our focus. To evaluate the effectiveness of these approaches, we conduct a comprehensive experimental setup using three-fold cross-validation and grid search. The evaluation is based on multiple performance metrics, allowing us to assess how well each method balances calibration with recommendation quality.

The work is organized as follows. Section 2 describes the related works with calibrated recommendation subject. Section 3 describes a background in calibrating rank. The proposed solution is shown in Section 4. We describe the experiments in Section 5 and discuss the results in Section 6. Finally, we conclude with insights and future directions in Section 8.

2. Related Works

Steck (2018) presents a movie recommendation system using the collaborative filtering method, which calibrates the recommendation list based on the user's previous genre preferences. In his paper, Steck argues that the problem of constructing a fair recommendation list while maximizing user utility is NP-hard. The divergence between the user's preferences and the recommendation list is captured using Kullback-Leibler divergence (Section 4.3) [Steck 2018]. To find the calibrated recommendation list, the author employs the greedy Surrogate Submodular algorithm (Section 4.5.1). Kaya and Bridge (2019) compared calibration with the intent-aware context, a concept in Information Re-

trieval that accounts for multiple possible interpretations of user queries. They presented variations of the calibration implementation and used Subprofile-Aware Diversification and Query Aspect Diversification, algorithms designed to provide diversity in the recommendation list [Kaya and Bridge 2019]. Still, they followed Steck's (2018) approach, using the Surrogate Submodular algorithm in their calibration implementation variations without introducing new algorithms or modifications. These related works apply a post-processing approach to achieve calibration, using the same item reordering/selection algorithm to generate a fair recommendation list. This algorithm addresses the problem studied, which is combinatorial and NP-hard. Using a similar approach, Da Silva *et al.* (2025) have shifted the attention to calibrated recommendations that also consider the time importance of ordering items [da Silva et al. 2025]. However, a state-of-the-art gap exists regarding exploring alternative algorithms that solve the same problem whilst we employ a variation of CELF.

Zhao *et al.* (2020) replaced the Surrogate Submodular algorithm with the Top-Z re-ranking algorithm to examine distortions in various traditional recommender algorithms. Their experiments are conducted using the Movielens 1M dataset. The evaluation relies on miscalibration and F1-Score metrics. The authors analyze recommendations based on user gender and age attributes. They found that F1-Score is not correlated with the calibration of these attributes. Additionally, as they have used maximal marginal relevance (MMR), their results confirm that miscalibration decreases as the λ parameter increases, aligning with findings from Stack (2018) and Kaya and Bridge (2019) [Zhao et al. 2020]. The λ is the trade-off between the original and the target recommendation distribution. Abdollahpouri *et al.* (2023) investigate calibrated recommendations as a minimum-cost flow (MCF) problem. They modify the linear trade-off proposed by Steck (2018), replacing the relevance equation with the Maximum Weight Assignment and the calibration with the Penalized Miscalibration Score. The authors implement the MCF optimization algorithm as the item selector, modeling the problem as a graph. They argue that the MCF approach guarantees an optimal solution for calibrated recommendations [Abdollahpouri et al. 2023]. Abdollahpouri *et al.* (2023) found that miscalibration also increases as the list's relevance increases. Among the optimization algorithms, MCF achieves the best precision and recall, and normalized discounted cumulative gain (NDCG). The authors categorize users into three groups based on their preference for popularity or diversity. They conclude that, depending on the trade-off weight, the metric values can either be lower or higher than the baseline [Abdollahpouri et al. 2023].

Silva and Durão (2021) introduce a calibrated recommendation system and evaluation metrics suite. Like Steck (2018) and Kaya and Bridge (2019), they employ the Kullback-Leibler divergence measure while proposing the Hellinger distance and Pearson Chi-Square as alternative calibration measures. Additionally, they introduce two novel metrics specifically for the calibrated recommendation context: miscalibration average calibration error (MACE) and mean rank miscalibration (MRMC) [da Silva et al. 2021]. The evaluation using six well-known collaborative filtering algorithms demonstrates that the Pearson Chi-Square measure outperforms other divergence-based calibration methods. Silva and Durão (2023) outline a framework for calibrated recommendation systems, breaking it into three steps and twelve components. This framework and the proposed trade-off balance drive the implementation of this paper. Their decision protocol evaluates 546 systems using three metrics and two coefficients to identify the optimal system. The

results align with Silva and Durão (2021), confirming that calibrated recommendations can achieve up to twice the precision of non-calibrated ones [da Silva and Durão 2023b]. Table 1 compares our proposal with the state-of-the-art implementations.

Table 1. Related Works Comparison

Authors / Year	Explores alternative algorithms in surrogate
[Steck 2018]	No
[Kaya and Bridge 2019]	No
[Zhao et al. 2020]	Yes - Top-Z
[da Silva et al. 2021]	No
[Abdollahpouri et al. 2023]	Yes - MCF
[da Silva and Durão 2023b]	No
[da Silva et al. 2025]	No
Our work	**Yes (variation of CELF)**

3. The Item Reordering/Selection Problem for Recommendation

The goal of calibrated recommendation is to generate a list of top-n items that is as fair as possible, i.e., items represented as a distribution of classes/genres that achieve the lowest divergence with the distribution based on the user's preferences. Thus, the calibrated recommendation list arranges items selected from a base set. To select these items, it is necessary to test all m candidate items in the n positions, making it a problem of permutation without repetition, which belongs to the NP-hard class. Equation 1 presents the permutation formulation, which defines our problem.

$$A_{m,n} = \frac{m!}{(m-n)!} \tag{1}$$

Figure 1 demonstrates an example of item selection, where a set of $m = 15$ candidate items and $n = 5$ positions in the recommendation list are considered. In this example, it would be necessary to test 360,360 possible arrangements to select the best list ultimately. This becomes unfeasible when the number of items for selection and positions increases, for example, $m = 21,000$ and $n = 30$.

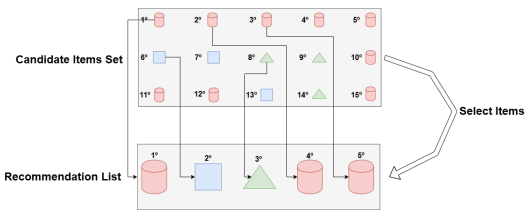


Figure 1. Example of item selection/ordering.

There are no existing methods to solve combinatorial optimization (permutation) problems in a satisfactory time frame. Thus, as shown in Figure 2, it is necessary to consider values other than the best value (global maximum) in exchange for obtaining results in a satisfactory time. The search is expanded to explore solutions beyond the

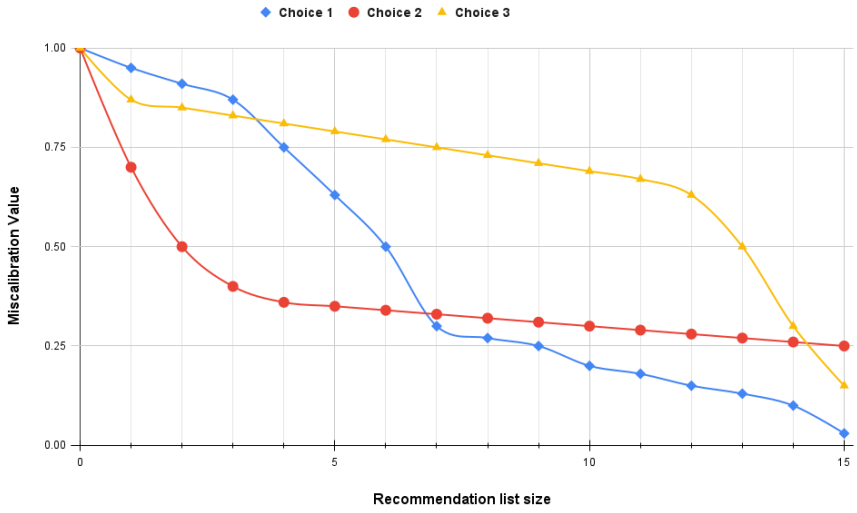


Figure 2. Example of different states of the objective function: the miscalibration level. This figure illustrates the miscalibration value as a function of recommendation list size for three distinct choices. The curves demonstrate that different initial choices lead to significantly varying miscalibration trajectories.

global maximum, often leading to a locally optimal value. Greedy algorithms can help find such solutions efficiently, though they do not guarantee an optimal local maximum.

In creating the optimal recommendation list, RL^*, it is necessary to check all combinations/arrangements of the list, seeking the best one. This task is a combinatorial/arrangement optimization problem, and it is NP-hard, as argued by [Steck 2018]. Moreover, testing all possible recommendation lists in a real dataset with millions of users' profiles becomes a hardware limitation. To avoid it, the Select Item Algorithm can work to find a local optimal list and not the optimal global list.

Figure 2 shows possible results. The x-axis represents a hypothetical recommendation list size, the y-axis is the miscalibration value, and the curves are three possibilities of a recommendation list. Selecting the best recommendation list is a combinatorial/arrangement optimization problem. In the figure example, the optimal list depends on the list size and the choices made from the first item to be included on the list. For instance, if the recommendation list has size 5, the best choice is Choice 2. If the recommendation list has size 15, the best choice is Choice 1. Thus, to select the best list, it is necessary to create the three lists and select the best one. However, it demands too many combinations and trials.

4. Proposed System

The proposed system is divided into three stages [da Silva and Durão 2023b]:

- **Pre-processing**: Aims to model the data to serve as input for the next stage.
- **Processing**: Seeks to filter a set of maximized candidate items through a recommendation algorithm, selecting items as similar as possible to the user's preferences. The items selected in this stage serve as input for the next stage.

- **Post-processing**: Aims to generate calibrated recommendations based on the user's history. Our approach focuses on this stage.

The post-processing stage utilizes the maximized candidate item model to generate a fair recommendation list. This list seeks to balance similarity and divergence, aiming to maintain the proportionality of genres in all users' preferences.

4.1. Data Models

Regardless of the dataset used by the system, the data must be formally modeled and defined. Each model determines how its data is handled during system execution. During pre-processing, the primary data is modeled for input in the processing stage, which, in turn, models data for use as input in the post-processing stage. Consider the item set, the user model, and the candidate item as the data model sets.

The items are the suggested objects in our system, which make up the recommendation list. The letter i represents an item, and I represents a set of items, such as

$$I = \{i_1, ..., i_z | 1 \leq z \leq \mathbb{N}\} . \tag{2}$$

Each $i \in I$ in our system is represented by a tuple with two metadata fields, $i = \langle UID, G \rangle$, where UID represents the unique identifier of the item, and G represents the genres to which the item belongs. The item set I is divided into two parts: the user model MU_u and the candidate item model MIC_u, for each $u \in U$.

In any recommendation system, users must somehow interact with the system's items. These interactions are called transactions and are selected to be modeled as user feedback. Thus, our user model, MU_u, represents all valid transactions between user u and a subset of items belonging to I:

$$MU_u = \{\langle i_1, w_{u,1} \rangle, ..., \langle i_z, w_{u,z} \rangle | 1 \leq z \leq |I|\} , \tag{3}$$

where $w_{u,i} \in \mathbb{R}$ is the weight/*rating/feedback*.

The candidate item model, MIC_u, consists of all items $i \in I$ that do not belong to the user model MU_u.

This is the processing stage, which filters the items from the user's candidate item model, MIC_u, creating a maximized subset,

$$MIC_u^{max} = \{\langle i_1, w_{r(u,1)} \rangle, ..., \langle i_m, w_{r(u,m)} \rangle | 1 \leq m \leq |MIC_u|\} . \tag{4}$$

In this subset, items are ranked based on the recommended algorithm's predicted weight. This weight is used to select items from MIC_u, removing any items that may not appeal to the user.

4.2. Probability Distribution of Genres

The genres belonging to the user model MU_u and the maximized candidate items MIC_u^{max} can be represented as probability distributions. To derive the target distribution based on the user model, we use the function p, which determines the importance of genres in the user model MU_u, as

$$p(g|u) = \frac{\sum_i^{MU_u} w_{u,i} \cdot p(g|i)}{\sum_i^{MU_u} w_{u,i}} , \tag{5}$$

where $w_{u,i}$ is the item weight given by the user u to item i, and $p(g|i)$ is the probability of observing g given i. This equation can be understood as a normalized weighted probability of observing g given u. Similarly, the realized distribution q for the recommendation list is defined to measure the distribution over the recommendation list S_u:

$$q(g|u) = \frac{\sum_i^{S_u} w_{r(u,i)} \cdot p(g|i)}{\sum_i^{S_u} w_{r(u,i)}}, \tag{6}$$

where $w_{r(u,i)}$ is the model-predicted weight of the item in the recommendation list.

There is a possibility that the value of $q(g|u)$ could be zero, which could lead to numerical instability. We introduce a balancing mechanism between the probabilities to prevent division by zero. A small value of α ensures that $q \approx \tilde{q}$, preserving the original result as much as possible. The adjusted probability distribution is given by:

$$\tilde{q}(g|u) = (1 - \alpha) \cdot q(g|u) + \alpha \cdot p(g|u)\,. \tag{7}$$

This formulation follows the framework introduced by [Steck 2018] and later adopted by [Kaya and Bridge 2019, da Silva and Durão 2023a, Abdollahpouri et al. 2020, da Silva et al. 2025].

4.3. Divergence Measures

Divergence and distance are related. Distance is defined as the degree to which two objects are far apart. There are numerous measures of distance. Measures that satisfy the metric properties are called metrics, while measures that do not satisfy these metric properties are occasionally referred to as divergences [Cha 2007, da Silva and Durão 2025]. Our system uses the Kullback-Leibler and Hellinger divergences introduced in [Steck 2018].

Kullback-Leibler is a measure of divergence between sets and can take values within the range $[0, \infty] \in \mathbb{R}$, where the smaller the value, the less divergent the sets are. It is defined as:

$$F_{KL} = KL(p||q) = \sum_{g}^{G} p(g|u) \log_2 \frac{p(g|u)}{\tilde{q}(g|u)}\,. \tag{8}$$

Hellinger is a measure of divergence between two sets, where $[0, \infty] \in \mathbb{R}$ represents the value of the divergence between the probability sets p and q. The Hellinger divergence has multiple formulations [Cha 2007, Jolad et al. 2016, da Silva and Durão 2025], we consider the following:

$$F_{HE} = HE(p, q) = \sqrt{2 \cdot \sum_{g}^{G} (\sqrt{p(g|u)} - \sqrt{q(g|u)})^2}\,. \tag{9}$$

4.4. Maximal Marginal Relevance

Since our goal is to create a recommendation list that is as fair as possible, we can use the MMR function introduced in [Carbonell and Goldstein 1998] and utilized in the works

presented in [Steck 2018] and [Kaya and Bridge 2019]. MMR represents the ranking algorithm designed to select the items that optimize the recommendation list. The formulation is presented as follows:

$$S_u^* = \arg\max_{S^*}(1-\lambda)\cdot \text{Rank}(S^*) - \lambda \cdot F(p, q(S^*))\,, \quad |S^*| = n, S^* \subseteq \text{MIC}_u^{max}\,, \quad (10)$$

where $F(p, q(S^*))$ is the divergence measurement used, which can be Kullback-Leibler or Hellinger. The term function Rank() is the sum of the predicted weight for each item belonging to MIC:

$$Rank(S_u) = \sum_{i}^{S_u} w_{r(i)}\,. \quad (11)$$

4.5. Item Reordering/Selection Algorithms

State-of-the-art works typically use the Surrogate Submodular to reorder/select items for the fair recommendation list. In our work, we propose a variation of a calibration algorithm for the fair recommendation problem.

4.5.1. Surrogate Submodular

The Surrogate Submodular algorithm, proposed by Nemhauser et al. (1978), finds a fair list with an optimality guarantee of $1 - \frac{1}{\mathrm{e}}$, where e is Euler's number [Nemhauser et al. 1978]. We will use this algorithm similarly to Steck (2018) and Kaya and Bridge (2019). Below are the steps describing how the reordering process works [Steck 2018, Kaya and Bridge 2019]:

- S^* starts with an empty recommendation list {}, S^* is the calibrated recommendation list;
- For each position in the recommendation list, $k \leq n$, an item i is selected from MIC_u^{max} if it is the item that maximizes the fairness value between S_u^* and MU_u;
- At the end, the recommendation list contains $n-1$ items, and the last item is added to obtain a fairer recommendation list.

The time complexity of the Surrogate Submodular algorithm is $\Theta(n \cdot m)$, where n is the number of items to be selected and m is the number of items to be tested for the n positions. This complexity is due to the algorithm testing all m items for each of the n positions. The algorithm always executes the same number of steps, meaning its complexity has no variations.

4.5.2. Cost Effective Lazy Forward

Developed by Leskovec et al. (2007), the CELF algorithm, known as the Lazy Greedy algorithm, is used to find the k nodes with the highest spread in a graph. To achieve this, it exploits the submodularity property of the spread function, implying that the marginal spread of a specific node in one iteration of the greedy algorithm cannot be greater than the marginal spread in the previous iteration [Leskovec et al. 2007]. Below are the steps describing how this algorithm works:

1. Calculate the spread of all nodes and store them in a list/heap, performing the ranking at the end.
2. Only the spread for the top node is calculated (the one at the top of the list from the previous step).
3. If the node from the previous step remains at the top of the list, it must have the highest marginal gain among all nodes.
4. This iterative process continues, finding the node that remains at the top after calculating its marginal spread and then adding it to the seed set.
5. At the end, the algorithm returns the optimal seed set, the resulting spread, and the time required to compute each iteration.

The original formulation of CELF uses the independent cascade (IC) as the cost function. In our proposed modification, we replace the IC formulation with Equation 10, so the spread values are determined through the balancing calculation. The complexity of CELF is given as:

- Best case: $\Omega(n \log n)$, achieved when the algorithm always finds the best item during step 3, executing only once. Thus, the time is bounded below by sorting items $n \log n$, as the loop for selecting items in steps 3 and 4 runs in $(n-1) \cdot 1$ steps.
- Worst case: $O(n \cdot m)$, achieved when the algorithm finds the best item at the end of the spread list, meaning steps 3 and 4 are executed $(n-1) \cdot m$ times.

5. Experiment Setup

5.1. Methodology

The proposed system uses the collaborative filtering technique and employs item genres as a balancing class, inspired by previous works. Due to hardware limitations, we selected only one recommender algorithm for the processing stage, namely *Singular Value Decomposition (SVD)*. The formulation described by [Koren and Bell 2015] is implemented by the Surprise library [Hug 2017] and is used in this work. We use grid search to determine the best recommender hyperparameter values among 81 input hyperparameter configurations. The decision metric used to find the best hyperparameter set is the MAE [Desrosiers and Karypis 2011], aiming to minimize the error between the user's actual feedback (w_{ui}) and the prediction made by the recommender ($w_{r(u,i)}$). The validation of the best hyperparameters is performed using cross-validation, dividing the dataset into three partitions. The optimal hyperparameter values found are: number of epochs, $ne = 50$, number of factors, $f = 100$, item learning rate, $\gamma_i = 0.05$, user learning rate, $\gamma_u = 0.05$, item regularization term, $\lambda_i = 0.03$, user regularization term, $\lambda_u = 0.03$, and activation of user and item bias. After this model fitting, we use the predicted weights to generate the rank list for each user.

The user models MU serve as the foundation for training the recommender. To validate the system before training, one-third of the transactions ($\sim 33\%$) were randomly removed from each user's model and added to their respective candidate item models MIC_u. This data separation allows us to determine which items within each MIC_u the users prefer, as these are the expected items in the final recommendation list. After the recommender algorithm predicts the weight $w_{r(u,i)}$ for all items in MIC_u, a cutoff is applied, limiting the maximized candidate item models to $|MIC_u^{max}| = 100$. These 100

items have the highest $w_{r(u,i)}$ values, indicating a higher likelihood of user preference. Additionally, the study by Kaya and Bridge (2019) uses the same cutoff value. Also following them, we use eleven fixed values of $\lambda \in [0.0, 0.1, 0.2, 0.3, ..., 1.0]$. A value of $\lambda = 0.0$ means that ranking is given full importance while divergence is ignored, resulting in a recommendation list identical to that produced by the recommender algorithms without post-processing. Conversely, a value of $\lambda = 1.0$ means that ranking is ignored and divergence takes full importance, recommending the least divergent items from each user's model.

In Section 4, we describe the use of α to prevent division by zero. We set $\alpha = 0.01$, a value also applied in the works of [Steck 2018, Abdollahpouri et al. 2020, Kaya and Bridge 2019]. In the same section, we define $p(g|i)$ as part of Equations 5 and 6. In our work, we apply the rule $\frac{1}{|i_G|}$, meaning that each genre receives an equal probability for each item.

To validate our system and ensure statistical stability in the results, we run our system three times for each dataset independently. That is, in each execution, the training data MU and the expected recommendations differ. The presented results are the average values obtained across the three runs for each dataset. Recommendation lists are generated with sizes $[1; 10] \in N$, and evaluations consider each position in the lists.

5.2. Datasets

The *MovieLens* dataset is publicly available and free-access, used in state-of-the-art works by [Steck 2018] and [Kaya and Bridge 2019]. It contains metadata describing the movies: title, genre, and *tags*, as well as links to additional information in other databases; it also includes information about user preferences and demographic data about the users. The dataset has several subsets, and in particular, we use *MovieLens 1M*[1].

The *MovieLens 1M* subset contains 18 genres, 6040 users, 3883 movies, and 1000209 *ratings* (transactions) with values $w_{u,i}$ between $[1, 5] \in \mathbb{R}$. These values $w_{u,i}$ represent the rating that user $u \in U$ assigned to movie $i \in I$ (explicit feedback). We applied a sequence of steps to clean the dataset, similar to other studies [Liang et al. 2016, Steck 2018, Kaya and Bridge 2019]: i) we discarded any transaction with $w_{u,i}$ less than 4, keeping all higher values; ii) we removed all movies without genre information; and iii) we eliminated any movie without user transactions, i.e., movies that were never watched. However, unlike [Steck 2018, Kaya and Bridge 2019] and similar to [Liang et al. 2016], we performed two additional steps in the following order: iv) we removed all users with fewer than 50 total preferences, and v) we eliminated all items with fewer than 5 transactions. Steps (iv) and (v) help us avoid the cold-start problem, which was not addressed in this study, and assist with hardware resource management. After applying all steps, the resulting dataset contains: 3379 users (i.e., $|U| = 3379$), 2330 movies (i.e., $|I| = 2330$), 495354 *ratings* with $w_{u,i}$ between $[4; 5] \in \mathbb{R}$ (i.e., $|MU| = 495354$), and 18 genres (i.e., $|G| = 18$).

5.3. Metrics

Recommendation Systems provide users with a list of size n containing the top-n most recommended items for a given user. Thus, a ranking metric is applied to quantify the

[1]https://grouplens.org/datasets/movielens/1m/

quality of the list. To obtain a single metric that contributes to the accuracy of the recommendation method across the entire user set, the Mean Average Precision (MAP) [Parra and Sahebi 2013] is used.

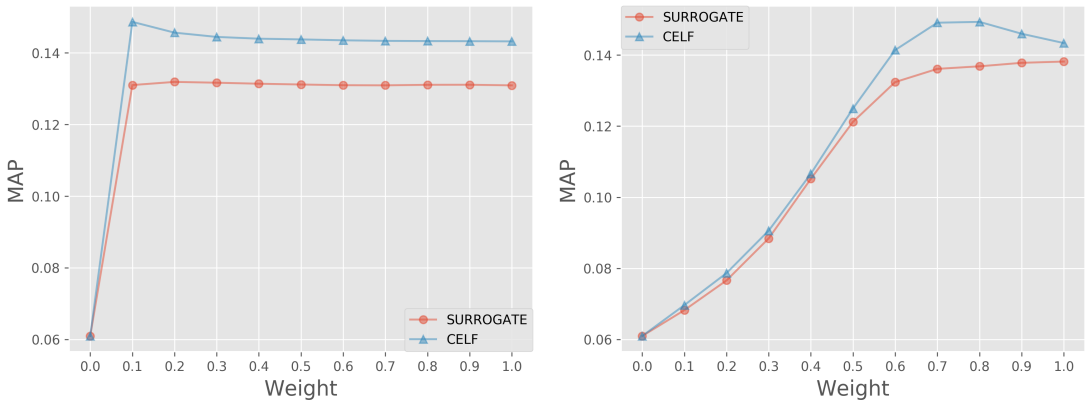
6. Experimental Results

The two figures containing the results (Figures 3 and 4) represent each divergence measure; the lines in the figures represent the item selection algorithms (surrogate and CELF). The x-axis of the graphs displays the weights $\lambda \in [0.0; 0.1; 0.2; 0.3; ...; 1.0]$, while the y-axis represents the metric values.

The results presented in Figure 3a illustrate the observation of the MAP metric over the selection algorithms using Kullback-Leibler. It can be observed that, with a balancing weight of 0.0, both item selectors yield similar results. When the balancing weight starts to increase (0.1), CELF achieves better results than the surrogate, and this difference is maintained up to a weight of 1.0, where only the Kullback-Leibler measure is considered. The most promising result from the cost function composition using Kullback-Leibler is obtained with a weight of 0.1 in the CELF algorithm, achieving an average precision of approximately 0.148.

The results presented in Figure 3b illustrate the observation of the MAP metric over the selection algorithms using Hellinger. It can be observed that, with a balancing weight of 0.0, both item selectors yield similar results, and this similarity is maintained up to a balancing weight of 0.4. From a weight of 0.5 onward, CELF achieves better results than the surrogate, and this difference increases up to a weight of 0.8. For weights 0.9 and 1.0, the difference between CELF and the surrogate decreases, although CELF still maintains an advantage. The most promising result from the cost function composition using Hellinger is obtained with a weight of 0.8 in the CELF algorithm, achieving an average precision of approximately 0.148.

When comparing Figures 3a and 3b, it is possible to observe that the Kullback-Leibler divergence achieves better results than the composition of the function using the Hellinger divergence, when used in the cost function.



(a) Kullback-Leibler. **(b) Hellinger.**

Figure 3. Results of the selection algorithms evaluated using the MAP metric while observing the balancing weights in the Kullback-Leibler and Hellinger divergence measures.

7. Limitations

The proposed method still has some limitations. It does not handle extremely sparse data scenarios or more complex multi-category problems. For instance, we have removed users with fewer than 50 ratings and items with fewer than 5 ratings, simplifying the experiment to avoid dealing with cold-start. This preprocessing step, while beneficial for reducing noise and computational load, limits the method's applicability to real-world environments where user and item interactions are typically sparse and unevenly distributed. Additionally, the system relies on a single collaborative filtering algorithm (FunkSVD), which constrains the generalizability of our findings across different recommender architectures. The exclusive use of the MovieLens 1M dataset further restricts the scope of the evaluation, as domain-specific datasets or those with higher complexity may pose additional challenges. Also, this work does not cover all aspects of calibration. The analysis exclusively focuses on MAP for ranking quality and on Kullback-Leibler and Hellinger divergences for calibration, potentially overlooking other critical dimensions of recommendation performance. For example, diversity-related metrics such as item coverage or intralist diversity are not considered, which could provide insight into how well the system avoids over-concentration on popular items. Similarly, novelty—capturing the system's ability to suggest less obvious or previously unseen items—is not evaluated. Furthermore, fairness measures, such as demographic parity or equal opportunity across user groups, are absent from the evaluation.

8. Conclusion

This work explores the problem of item selection for fair recommendation, based on the distribution of genres of the items that make up the users' preferences. The problem addressed belongs to the class of NP-hard problems due to its permutation nature. The proposal used CELF as a new algorithm beyond the state of the art and also proposed the application of new cost functions in the algorithm. The results indicate that the proposal to use CELF with a new cost function achieves promising results when evaluated using the precision metric. The formulation of the cost function using the Kullback-Leibler divergence measure achieves better results than the formulation using the Hellinger divergence, for miscalibration, both item selection algorithms are used.

As a future work, it is possible to use other algorithms based on cost functions, adapting them to the balancing formulation. Another future work is to use other cost functions based on the similarity-divergence balance. Additionally, evaluating the algorithms beyond precision, using evaluation metrics that verify whether the final generated list is truly fair. A broader evaluation would be necessary to fully capture the system's strengths and weaknesses across different calibration dimensions.

9. Acknowledgments

We want to thank the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES – Brazil) for funding this research under Grant Codes 88887.685243/2022-0 and 001, and 88887.941435/2024-00. This research was also supported in part by the FAPESB INCITE PIE0002/2022 grant.

References

Abdollahpouri, H., Mansoury, M., Burke, R., and Mobasher, B. (2020). The connection between popularity bias, calibration, and fairness in recommendation. In *Fourteenth ACM Conference on Recommender Systems*, RecSys '20, page 726–731, New York, NY, USA. Association for Computing Machinery.

Abdollahpouri, H., Nazari, Z., Gain, A., Gibson, C., Dimakopoulou, M., Anderton, J., Carterette, B., Lalmas, M., and Jebara, T. (2023). Calibrated recommendations as a minimum-cost flow problem. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, WSDM '23, page 571–579, New York, NY, USA. Association for Computing Machinery.

Aragão, L. R., Silva, M., and Machado, J. (2024). The inefficiency of achieving fairness with protected attribute suppression. In *Anais do XXXIX Simpósio Brasileiro de Bancos de Dados*, pages 813–819, Porto Alegre, RS, Brasil. SBC.

Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 335–336, New York, NY, USA. ACM.

Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *Int. J. Math. Model. Meth. Appl. Sci.*, 1.

da Silva, D. C. and Durão, F. A. (2023a). How novel and unexpected the calibrated recommendations are? In *American Conference on Information Systems*, volume 6.

da Silva, D. C. and Durão, F. A. (2023b). Introducing a framework and a decision protocol to calibrated recommender systems. *Applied Intelligence*.

da Silva, D. C. and Durão, F. A. (2025). Benchmarking fairness measures for calibrated recommendation systems on movies domain. *Expert Systems with Applications*, 269:126380.

da Silva, D. C., Jannach, D., and Durão, F. A. (2025). Considering time and feature entropy in calibrated recommendations. *ACM Trans. Intell. Syst. Technol.* Just Accepted.

da Silva, D. C., Manzato, M. G., and Durão, F. A. (2021). Exploiting personalized calibration and metrics for fairness recommendation. *Expert Systems with Applications*, page 115112.

Desrosiers, C. and Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*.

Hug, N. (2017). Surprise, a Python library for recommender systems. `http://surpriselib.com`.

Jolad, S., Roman, A., Shastry, M. C., Gadgil, M., and Basu, A. (2016). A new family of bounded divergence measures and application to signal detection. *Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods*.

Kaya, M. and Bridge, D. (2019). A comparison of calibrated and intent-aware recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 151–159, New York, NY, USA. Association for Computing Machinery.

Koren, Y. and Bell, R. (2015). *Advances in Collaborative Filtering*, pages 77–118. Springer.

Leite, N., Campelo, C. E. C., and Silva, S. D. (2024). Leveraging geographic feature embeddings for enhanced location-based recommendation systems. In *Anais do XXXIX Simpósio Brasileiro de Bancos de Dados*, pages 354–366, Porto Alegre, RS, Brasil. SBC.

Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., and Glance, N. (2007). Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429.

Liang, D., Altosaar, J., Charlin, L., and Blei, D. M. (2016). Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 59–66, New York, NY, USA. Association for Computing Machinery.

Lin, K., Sonboli, N., Mobasher, B., and Burke, R. (2020). Calibration in collaborative filtering recommender systems: A user-centered analysis. In *Proceedings of the 31st ACM Conference on Hypertext and Social Media*, HT '20, page 197–206, New York, NY, USA. Association for Computing Machinery.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions–i. *Math. Program.*, 14(1):265–294.

Parra, D. and Sahebi, S. (2013). Recommender systems: Sources of knowledge and evaluation metrics. In *Advanced techniques in web intelligence-2*, pages 149–175. Springer.

Sena, L. and Machado, J. (2024). Evaluation of fairness in machine learning models using the uci adult dataset. In *Anais do XXXIX Simpósio Brasileiro de Bancos de Dados*, pages 743–749, Porto Alegre, RS, Brasil. SBC.

Steck, H. (2018). Calibrated recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, pages 154–162, New York, NY, USA. ACM.

Zhao, X., Zhu, Z., Alfifi, M., and Caverlee, J. (2020). Addressing the target customer distortion problem in recommender systems. In *Proceedings of The Web Conference 2020*, WWW '20, page 2969–2975, New York, NY, USA. Association for Computing Machinery.