Studying diversity in recommender systems

Augustin Godinot
Département EEA
Ecole Normale Supérieure Paris-Saclay
4 Avenue des Sciences, 91190 Gif-Sur-Yvette, France
augustin.godinot@ens-paris-saclay.fr

Avant-propos

Ce rapport résume cinq mois de travail sur le thème de la diversité dans les systèmes de recommendation. Ce travail s'inscrit dans le cadre du premier semestre de l'année ARIA du parcours IA de l'ENS Paris-Saclay. En plus de ce travail de recherche, j'ai pu suivre deux cours du master MVA. L'interaction entre ces cours et le projet ont été bénéfiques à bien des égards. Le cours d'*Optimisation Convexe* a fourni un éclairage théorique et technique sur le modèle de filtrage collaboratif exploré dans ce rapport. Le cours de Transport Optimal a, quant à lui, offert une autre méthode pour mesurer la dissimilarité entre utilisateurs en se basant sur la même structure de graphe triparti introduite dans ce rapport. Enfin, l'étude interdisciplinaire de l'Intelligence artificielle, objectif de ce parcours IA, ne serait complète sans aborder les enjeux sociétaux associés. Dans cette optique, le cours *Enjeux Numériques du Monde Contemporain* a été l'occasion de réfléchir sur l'impact juridique, social et démocratique des systèmes de recommendations, ce qui a apporté une analyse supplémentaires aux résultats exposés dans ce rapport. Je tiens à remercier chaleureusement Fabien Tarissan pour son implication tout au long du projet, son aide pour donner un sens aux résultats et sa relecture.

1 Introduction

With the ever growing quantity of information and data available on online plateforms, comes the need to build efficient and reliable methods to filter this information. Recommender systems were introduced to achieve this very task. Applications of recommender systems range from email filtering [9], news recommendation [15] to music recommendation [10]. Recommender systems' popularity is due to their ability to filter a great number of items to present only a few relevant ones to the user. However, these algorithms face challenges due to overpersonalization of their results. Recently, news recommender systems have been pointed at for the apparition of echo chambers (or filter bubbles) and their role in fake-news spread [12]. In this context, diversity was introduced as property able to mitigate the side effects of recommender systems [13]. Before the study of recommender systems, the notion of diversity has been widly studied in other fields such as ecology or economics. As we will see in section 2, there is no consensus on the method to apply to evaluate the diversity of a recommender system.

This work is based upon a framework for measuring diversity in heterogeneous networks introduced in [16] and applied to music consumption in [17]. To illustrate our method, we will focus on the case of music recommender systems and their effects on music consumption but it can be transposed to any other problem with similar structure. Most of the data of musical consumption is generated by private content streaming companies, Spotify, Deezer, Pandora to cite a few. Because the content recommendation is at the heart of the service they provide, the listening data produced by their users is very valuable and sensitive (not to mention the protection of their users' privacy) and thus not shared. Moreover, the algorithms used in their recommender systems are not or partially disclosed. Therefore, in this work, we try to provide a framework to evaluate the performance of recommender systems offline (ie without being able to implement an algorithm an observe user reaction). We focus our study on a specific user-based collaborative filtering recommender system.

Paper outline After reviewing the existing litterature on recommender systems and diversity in section 2, we present our method of diversity evaluation in section 3. In section 4 we apply our method on the *Million Songs Dataset*. Finally, we conclude the paper and pave the way for future work in section 5.

1st semester project, ARIA year, Ecole Normale Supérieur Paris Saclay.

2 Background and Related Work

In this section, we review the existing litterature on recommender systems, the study of their impact on society and in particular on music consumption.

Recommender systems can be separated into two groups: content based and user based [1]. Content based recommender systems exploit properties of items to find similarities between them. Such properties are either given by metadata (title, genre ...) or computed via information retrieval algorithms. User based recommendation exploits learned similarities bewteen users to make personalized recommendations. Nowadays, recommender systems usually mix these two approaches with for example Ensemble Learning [18]. Thanks to the high granularity of the data collected by online platforms, the objective of modern recommender systems is to provide truely personalized recommendations, taylored to the user and the listening context. The most common user based recommender is Collaborative Filtering [6]. It is based on the assuption that the similarities between users can be discovered from similarities between the ratings they gave to items. With the recent advances in Deep Neural Networks, deep learning based recommender systems have appeared [11, 4, 22]. An approach based on Neural Collaborative Filtering [11] is to replace the scalar product (see subsection 3.2) by a neural network as the affinity metric between a user and an item. While papers introducing these new recommender systems show an improvement in performance, the authors of [5] highlighted the lack of reproductability of these results. An other approach based on user and song embedding, adapted from *Natural Language Processing*, is introduced in [10]. This approach is also one in many trying to take into account the time and context of listenings for the recommendations.

A lot of studies on the diversity of recommender systems and the impacts it has on the user consumption took the case of news recommender systems as a subject. The authors of [12] studied the role of news recommender in today's democracies. They highlight the fact that following one's conception of democracy (liberal, participaory, deliberative or critical), exposing a user to diverse opinions is not always desirable. This reminds us that diversity is a property of recommender systems and should not be a goal itself. In the case of music recommender systems, the authors of [21] introduce a diversity measure based on community embedding. This method is applied to Spotify consumption data in [2]. They use a version of *word2vec* applied to song embedding to compute the GS-score of each user in order to quantify their diversity. Other methods use the mean of pairwise distance bewteen items listened by a user to derive a diversity index [20]. The issue with these methods is that they all boil down to the measure of a distance very dependent on the item/user embedding method and parameters. The method introduced by [16] and used by [17] eliminates the need for a user/song embedding since it only relies on the graph users $\rightarrow$ items $\rightarrow$ tags (see section 3). The domain of optimal transport also offers a promising framework to compute distance between users's subgraphs. [8] introduces a notion of canonical distance between subgraphs which could be used to compute distance between users.

3 Measuring the diversity of a recommender system

In this section, we introduce the formalism used to quantify user diversity and a particular type of recommender system that will be the focus of our study : *Collaborative Filtering via Matrix Factorization for Implicit Datasets* [14].

3.1 A formalism to analyse diversity

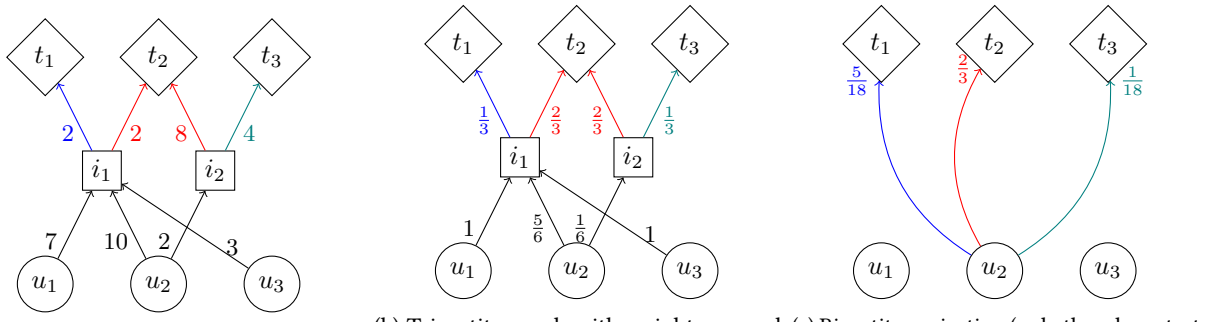
In this subsection, we quickly describe the formalism introduced by [16, 17].

3.1.1 Tripartite graph

A bipartite graph is a graph with two disjoint sets of nodes such that there is no link between nodes in the same set. Formally, it is defined by a triplet $\mathbb{B} = \left(\top, \bot, E_{\top}^{\bot}\right)$. $\top$ is the set of *top* nodes (e.g. items), $\bot$ is the set of *bottom* nodes (e.g. users) and $E \subset \top \times \bot$ is the set of edges between the *top* and *bottom* nodes. Bipartite graphs can be used to express the relation between users and items or items and tags. In order to capture the complete structure of users' activity, [17] suggested the use of a tripartite graph. A tripartite graph is composed of two bipartites graphs with a common node set and is formally defined in Equation 1

$$\mathbb{T} = \left(\top, \vdash, \bot, E_{\bot}^{\vdash}, E_{\vdash}^{\top}\right) \tag{1}$$

As for bipartite graphs, $\top$, $\bot$ and $\vdash$ are disjoint sets of nodes. $E_{\bot}^{\vdash}$ is the set of edges between the nodes in $\top$ and $\bot$, and $E_{\vdash}^{\top}$ is the set of edges between the nodes in $\vdash$ and $\bot$. In addition, weights can be assigned to edges by defining the associated weight functions: $w_{\bot}^{\vdash} : E_{\bot}^{\vdash} \rightarrow \mathbb{R}_{+}$ and $w_{\vdash}^{\top} : E_{\vdash}^{\top} \rightarrow \mathbb{R}_{+}$. An example of such tripartite graph is given



(a) Tripartite graph with original weight

(b) Tripartite graph with weights normalized by node

(c) Bipartite projection (only the edges starting from the user u_2)

Figure 1: Example of a tripartite graph and the random walk bipartite projection. The users u_i listen to items i_j which are tagged by category t_k

in Figure 1a. In the following work, for any vertice v we will note $N(v)$ its set of neighbors, $d(v)$ its degree and d_w its weighted degree.

$$d(v) = |N(v)| \quad \text{and} \quad d_w(v) = \sum_{u \in N(v)} w(u, v) \tag{2}$$

3.1.2 Random walk and bipartite projection

As discussed before, the diversity measure is based on the extraction of a probability distribution from the tripartite graph. This distribution is obtained via a random walk. We take here the case of the computation of the diversity of a bottom node but it can be mirrored to obtain the diversity of a top node.

For each vertice v, we define in the probability to reach a neighbour $z \in N(v)$ as $p_{v \to z} = \frac{w(v,z)}{d_w(v)}$ Therefore, for each bottom node $u \in \bot$ and top node $t \in \top$, we define in Equation 3 the probability $p_{u \to t}$ to reach t from u through $\vdash$. An example of this normalization is given in Figure 1b.

$$p_{u \to t} = \sum_{i \in N(u) \cap N(t)} p_{u \to i} p_{i \to t} \tag{3}$$

If we repeat this process for each bottom vertice, we obtain the *bipartite projection* $\Pr(\mathbb{T}) = \left(\top, \bot, E_{\top}^{\bot}, w_{E_{\top}^{\bot}}\right)$ of the tripartite graph into a bipartite graph. $\top$ and $\bot$ are the *top* and *bottom* nodes of the tripartite graph $\mathbb{T}$. As described in Equation 4, $E_{\top}^{\bot}$ is the set of edges between top t and bottom b nodes such that there exists a *meta-path* from b to t trough a node m in the set of "middle" nodes $\vdash$.

$$E_{\top}^{\bot} = \left\{(t, b) \in \top \mid \exists\, m \in \vdash \; s.t. \; (b, m) \in E_{\bot}^{\vdash} \text{ and } (m, t) \in E_{\vdash}^{\top}\right\} \tag{4}$$

The weights of the resulting edges between each top t and bottom u nodes are the transition probabilities $p_{u \to t}$. An example of the resulting bipartite graph is given in Figure 1c.

3.1.3 Diversity score

We base our diversity index on the *true diversity of order* d (or Hill Number) $D_d(p)$ [16]. For any probability vector p ($p_i \in [0, 1]$, $\sum_i p_i = 1$), and positive d the expression of $D_d(p)$ is given in Equation 5

$$D_d(p) = \left(\sum_{i=1}^{k} p_i^d\right)^{\frac{1}{1-d}} \text{ if } d \neq 1 \quad \text{and} \quad D_1(p) = \left(\prod_{i=1}^{k} p_i^{p_i}\right)^{-1} \text{ if } d = 1 \tag{5}$$

Following the values of d, this diversity score can express well-known diversity measures. $d = 0$ gives the richness diversity, $d = 1$ is the exponential of the Shannon entropy, $d = 2$ gives the Herfindal diversity and $d = \infty$ gives the Berger diversity (see [16] for a comprehensive review).

This work will mainly be based on Herfindal diversity. With $(p_{u\to t})_{t\in\top}$ the distribution of tags reachable from user u, we will call $D_2((p_{u\to t})_{t\in\top})$ the diversity of user's attention. With $(p_{t\to u})_{u\in\bot}$ the distribution of users reachable from tag t, we will call $D_2((p_{t\to u})_{u\in\bot})$ the diversity of tag's audience.

3.2 Recommender system

Throughout this work, we will focus on a particular type of recommender system : *Collaborative Filtering For Implicit Dataset* [14]. After training, the recommendation process is divided in two steps: prediction and selection. In the prediction stage, score is computed from the trained model whereas in the selection stage, a fixed number of items are selected based on these scores.

3.2.1 Dealing with implicit data

We assume that we have access to a list of triplet (user u, item i, number of listenings r_{ui}), all described by the matrix $R = (r_{ui})_{u,i\in U\times I}$. The number of listenings r_{ui} is also called the rating of item i by user u and R is called the rating matrix. When an item i has never been listened by a user u, we assume $r_{ui} = 0$. This number of listenings is an *implicit* feedback from the user. Unlike situations where we have access to a score given by a user to an item, the fact that a user did not listen to an item doesn't mean that they did not like it. Therefore, the authors of [14] introduced a notion of *confidence*. According to their model, the number of listenings only indicates our confidence c_{ui} in the proposition "the user u likes the item i". Because there is no canonical relation between r_{ui} and c_{ui} we will use a simple model suggested in [14]. This model is defined in Equation 6, with $\alpha \geq 0$. The variable p_{ui} is introduced to encode the fact that a user likes an item. We consider that as soon a user listens to a song, they are prone to like it, therefore $p_{ui} = 1$.

$$c_{ui} = 1 + \alpha r_{ui} \quad \text{and} \quad p_{ui} = \begin{cases} 1 \text{ if } r_{ui} > 0 \\ 0 \text{ else} \end{cases} \tag{6}$$

3.2.2 Prediction stage

As with classical matrix factorization, we assume that each user u (resp. each item i) can be represented by a column vector of *latent factors* x_u (resp. y_i). We note $x = (x_{u_1} \dots x_{u_n})$ (resp. $y = (y_{i_1} \dots y_{i_m})$) the matrix of user factors (resp. item factors). The estimator of p_{ui} is then $\widehat{p}_{ui} = x_u^T y_i$. The values x^* and y^* of the *latent factors* are computed by minimizing the squared difference between p_{ui} and its estimator $\widehat{p}_{ui}$, weighted by the confidence c_{ui}.

$$x^*, y^* = \min_{x,y} \sum_{u,i} c_{ui} \left(p_{ui} - x_u^T y_i\right)^2 + \lambda \left(\|x\|_2^2 + \|y\|_2^2\right) \tag{7}$$

In order to prevent the model from overfitting the training data, we add a regularization term with $\lambda \geq 0$ and $\|x\|_2 = \sqrt{\sum_k \sum_l x_{kl}^2}$ the Frobenius norm. The value of λ is dataset-dependent and is usually determined by cross-validation. To solve this problem, we use the conjugate gradient method described in [19].

3.2.3 Selection stage

We assume now that the optimal latent factors x^* and y^* have been computed. For the sake of simplicity we will drop the star for the computed optimal latent factors.

To recommend a set of items to a user u we first select a set of candidates I_u. Usually, this set contains all the items i a user dit not listened to : $I_u = \{i \in I \mid r_{ui} = 0\}$. Then, these items are sorted by descending score $\widehat{p}_{ui}$. Finally, the recommended items are the k-best items in the sorted candidates set I_u.

In order to give certain properties to the recommender system, other selection strategies can be used. For example, in order to increase the *novelty* of the recommendations, we could randomly add items in the recommendations or select items from an under represented item category.

3.2.4 Training and evaluation

The model in Equation 7 is optimized with a *Regularized Alternative Least Squares* method as described in [19]. Each least squares problem (which is equivalent to solving a linear system) is solved via *Conjugate Gradient Descent*.

To train and characterize the perfomance of the studied model, we split the dataset into two subsets: a *training* and a *test* dataset. Consider U to be the set of users, I the set of items and, $\mathcal{R} = \{r_{ui} \mid u \text{ rated } i\}$ the set of known user-item pairs. There are many ways to split $\mathcal{R}$ into training and test datasets [6], our method is as follow :

1. Randomly select a proportion β of users in U, we name the resulting set U_{test} (note that this is not the test set).
2. For each user u_{test} in U_{test}, randomly select a portion γ of the items listened by this user and put them in the test user-item set $\mathcal{R}_{\text{test}}$. We make sure that while sampling listened items from a user u, there is at least a $u \to i$ pair in the train set.

This results in a testing set $\mathcal{R}_{\text{test}}$ containing users that will all have been encountered during training, therefore all having a user factor, eliminating the cold start problem.

3.2.5 Evaluation metrics

There are typically three classes of metrics to evaluate recommender systems. In the following discussion, z is a vector of real numbers and $\widehat{z}$ is an estimator of z.

Error metrics are based on the loss of the model being trained. Classical examples include Root Mean Square Error ($\text{RMSE}(\widehat{z}, z) = \frac{1}{K}\sum_k (\widehat{z}_k - z_k)^2$) or Mean Absolute Error ($\text{MAE}(\widehat{z}, z) = \frac{1}{K}\sum_k |\widehat{z}_k - z_k|$). In our situation, we consider the model's loss as our error metric. These metrics are very close to the model but are not well suited to the case of recommender systems because they are not easily interpretable.

Precision metrics are based on the capacity of the model to predict items that the user likes. Usually, these metrics are defined with respect to the confusion matrix which gathers the number of true positives P_+, true negatives N_+, false positives P_- and false negatives N_-. The two metrics we will be using are Precision (number of true positives over the number of measured positives) and Recall (number of true positives over the number of real positives).

$$\text{Precision} = \frac{P_+}{P_+ + P_-} \quad \text{Recall} = \frac{P_+}{P_+ + N_-} \tag{8}$$

Other metrics such as the Area Under the ROC Curve can be derived from the confusion matrix. These metrics catpure the ability of the model to classify the items as liked or not.

Ranking metrics capture the relevance of the ranking issued by the algorithm with respect to the user's tastes. The authors of [14] used a metric called *percentile ranking*. Another wildely adopted metric that we will use is the Normalized Discounted Cumulative Gain (NDCG). These metric evaluate if items that a user is likely to listen are placed on top of the recommendation list.

The *error* metrics are based on the predicted values $\widehat{p}_{ui}$ for each user-item pair in the test set. To compute the *precision* and *ranking* metrics, we first have to generate recommendations. As described in subsubsection 3.2.3, for each user, we rank the items (taken from the whole dataset) not listened to (ie not paired to the user in the train set), order them by descending value of predicted value $\widehat{p}_{ui}$ and take the k-best.

3.3 Diversity of the recommender system

Now that we have a formalism to measure diversity in user-item-tag graphs and a method to generate personalized recommendations for users, we introduce a method to measure the diversity of the recommender system. We first adapt the method from subsection 3.1 to work with recommendations. Then we introduce a method to study the effect of recommendations on user diversity based on simple modelling of these users.

3.3.1 Diversity of the recommendations

We construct a new tripatite graph $\mathbb{T}_r$. The edges between the top $\top$ (tags) and middle $\vdash$ layers (items) are the same as in the dataset. For each recommendation of item $i \in \vdash$ to user $u \in \perp$, we create an edge $u \to i$. There are many options to assign a weight to this edge. One could choose for example the inverse of the recommendation rank: $\frac{1}{\text{rank}_{ui}}$, the linear "inverse": $\max_i rank_{ui} - rank_{ui}$, the prediction score: $x_u^T y_i$ ... We chose a linear relation between the weight and the rank (n_r is the number of recommended items per user): $w(u,i) = a_u \text{rank}_{ui} + b_u$. We assume that the user will listen to as many recommendations as the number u_v of items they have listened to (ie u_v is the volume of items listened to by user u). With $(i_{uk})_{k \in [\![1, n_r]\!]}$ the n_r items recommended to user u, this yields the constraint

$\sum_{k=1}^{n} aw(u, i_k) + b = u_v$. Finally, assuming that the last item recommended is not listened to ($w(u, i_{n_r}) = 0$), we obtain the values in Equation 9.

$$w(u, i) = \frac{2u_v}{n_r(n_r - 1)}(n_r - \text{rank}_{ui}) \tag{9}$$

With this formulation, the weights of the edges between users and items have the dimention of a number of listenings as in the dataset graph. Moreover as we could expect, the further an item is in the ranking, the less it is listened to. It is important to note that this choice of weight function hides a model of the user. This is neccessary to evaluate the recommender system in an offline context. The authors of [17] proposed other, more advanced, user models that capture the saturation of users' attention when increasing the listening volume.

3.3.2 Effect of recommendations on user diversity

We assume that we have the graph $\mathbb{T}_d$ associated to the entire dataset. We compute the individual Herfindal diversity $\text{hd}(\mathbb{T}_d, u)$ of each user in the test set (ie for whom we have computed recommendations). Then, for each item i_r recommended to user u, we add an edge $u \to i_r$ with weight $w(u, i_r)$ as in Equation 9. If the edge $u \to i_r$ already existed, we simply sum the existing weight with $w(u, i_r)$. After these operations, we obtain a new triparite graph $\mathbb{T}_{d,r}$. With this graph, we compute the individual Herfindal diversity after recommendations $\text{h}(\mathbb{T}_{d,r}, u)$ of each user in the test set. Finally, we can compute the diversity difference $\Delta_{\text{hd}}(u)$ before and after recommendations for each user in the test set.

$$\Delta_{\text{hd}}(u) = \text{h}(\mathbb{T}_{d,r}, u) - \text{hd}(\mathbb{T}_d, u) \tag{10}$$

4 Numerical results

In this section, we present the results obtained on the *Million Songs Dataset*. Particular attention attention has been paid to the reproducibility of our results. All the parameters are specified within the figures' captions or in the result introduction.

4.1 Implementation

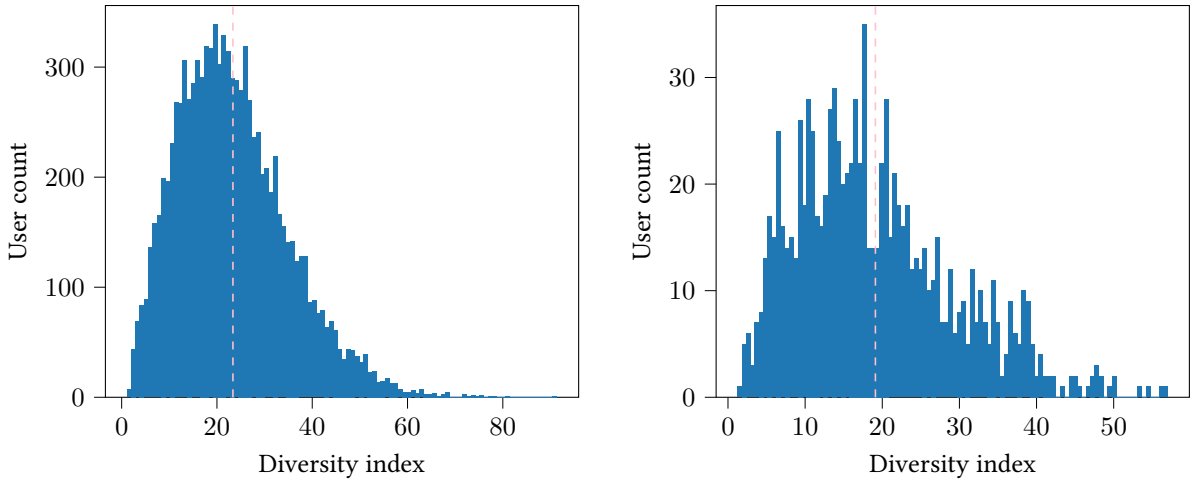
We run all of the following experiments using the Python programming language on a 40 cores `Intel Xeon` server, equipped with 256 GB of RAM and our code is available on GitHub [1]. We use the package `lenskit` [7] to instanciate, train and evaluate the recommender system. To quantify diversity, the code from [17] was adapted to our situation. In order to orchestrate all the different models (with differents hyperparameters) and the following analysis, we use a task scheduler (or pipeline builder): the package `luigi` [2]. Thanks to this package, all the figures outputed by our script can be easily redrawn in one command.

4.2 Dataset

The dataset we used for the experiments stems from the *Million Song Dataset (MSD)* project [3]. To populate the first two layers of the triparite graph and create the user-item links, we used the *Echo Nest user tast profile* dataset. It contains triplets (user, item, play count) that describe how many times a user has listened to a given song. In order to reduce the training time of our models, we randomly sampled $10,000$ users and their played items. To populate the third layer of the triparite graph and create the item-tag links, we used the *last.fm* dataset. It contains (item, tag, strength) triplets that describe the tags associated to each song and how representative is each tag.

Moreover, for the results to be comparable with those of [17], we applied the same filters to the original dataset: we selected only the $1,000$ most popular tags, deleted songs with no tag and users with no item and deleted items with ambiguous identifiers [3]. The relevant statistics of the dataset are summurized in table Table 1 and the users' diversity histogram of the train and test sets are plotted in Figure 2.

[1] https://github.com/grodino/recodiv
[2] see https://github.com/spotify/luigi
[3] see http://millionsongdataset.com/blog/12-2-12-fixing-matching-errors/



(a) Users in the train set. mean = 23.36

(b) Users in the test set. mean = 19.11

Figure 2: Diversity of users' attention. The mean is represented by a pink hatched line.

	number of users	number of items	number of tags	number of user-item links
full dataset	$10,000$	$77,254$	$1,000$	$374,329$
train set	$10,000$	$75,585$	$1,000$	$355,692$
test set	$1,000$	$12,403$	$1,000$	$18,637$

Table 1: Statistics of the dataset, the training set and the testing sets

We note $\mathbb{T}_d$ the tripartite graph derived from the entire dataset, $\mathbb{T}_{\text{test}}$ the graph derived from the test set and $\mathbb{T}_{\text{train}}$ the graph derived from the train set.

4.3 Model training

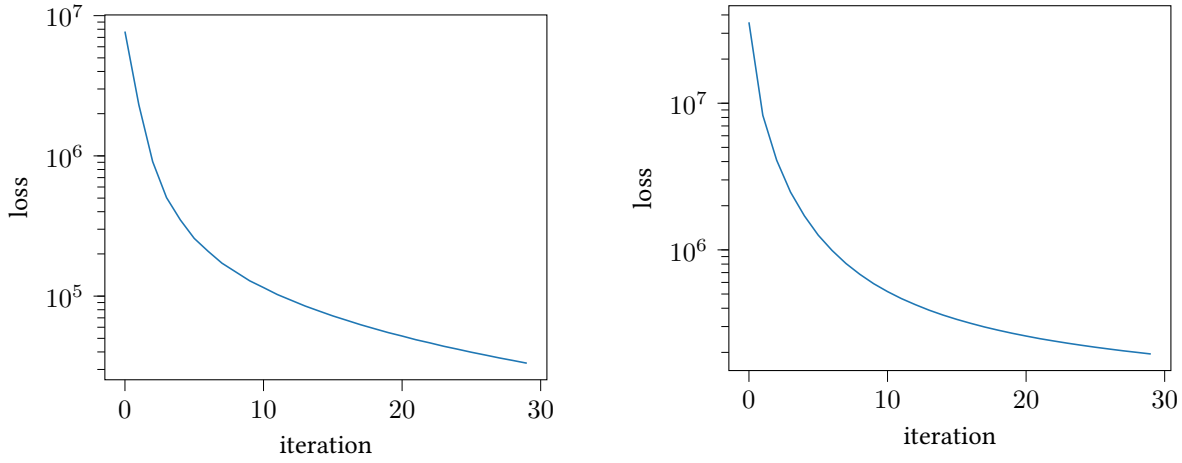
4.3.1 Number of iterations

When optimizing a loss such as Equation 7, the stopping criterion has a significant effect on the execution time. Classical stopping criterion usually rely on the difference of the loss value between two iterations or the norm of the difference between two variable iterates being smaller than a given ϵ. Such stopping criterion is relevant when we want to enforce a certain precision at the expense of execution time. In our case, because of the large amount of data to process, we use a fixed number of iterations in order to control the execution time. Experimentally, we can separate the convergence plot in two regimes (see Figure 3). Between 0 and 7-10 interations, the loss decreases at a high rate (around a decade for each two iterations). From 10 iterations, the loss dercreases at a lower rate (around a decade for 20 iterations). Therefore, we choose to set the number of iterations at 10 as a compromise between precision and execution time. Finally, as suggested by [14], we use a confidence factor α value of 40.

4.3.2 Hyperparameter tuning

The best values of the regularization factor λ and number n of latent factors are determined via grid search. For the computation of the NDCG and in the rest of the article (unless stated otherwise), we set the number of recommendations r, generated for each user to $r = 50$. As shown by Figure 4, following the metric given to the space of hyperparameters, the optimal point varies. Interestingly, evaluating the model with the test loss yields poor results. This could be explained by the increasing value of $x_u^T y_i$ in Equation 7 as the number of factors grow.

We choose to use the NDCG as the metric to optimize since it takes into account the rank of the recommended items. Experimentally, we obtain $\lambda^* = 10$ and $n^* = 3000$. Yet, because the training time of the model grows with the number of factors, we choose to set the number of factors to $n = 500$ and use the corresponding optimal regularization $\lambda = 5,000$.

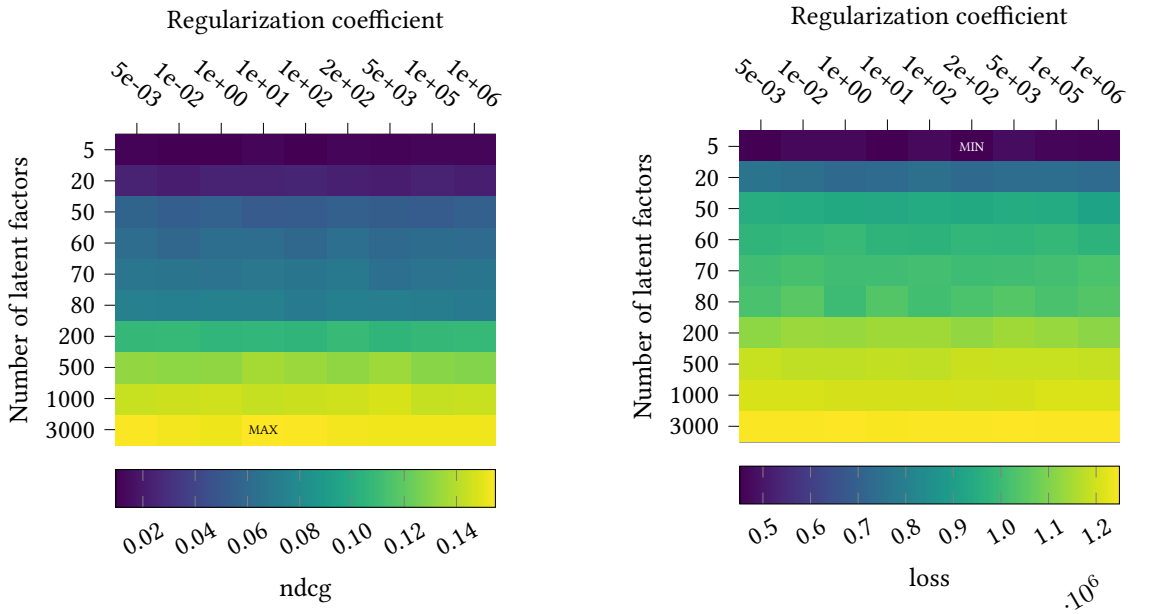


(a) 3, 000 factors, $\lambda = 10^6$

(b) 200 factors, $\lambda = 10^{-3}$

Figure 3: Convergence of the model. The loss of Equation 7 is plotted for each iteration of the alternated least squares. The number of iterations for each *Conjugate Gradient Descent* problem is 3

.



(a) Evaluation with NDCG, evaluated on test set

(b) Evaluation with loss, evaluated on test set

Figure 4: Hyperparameter tuning with grid search.

4.4 Analysis of the model at equilibrium

In this section we evaluate the diversity of the recommender system at equilibrium ($n = 500$, $\lambda = 5,000$).

In Figure 6 we plot the diversity increase (see subsubsection 3.3.2) due to the recommendations made to each user with respect to its *organic* diversity (ie the diversity of this user computed with the dataset). We observe that for the majority of the users, the recommendations lower their diversity. Moreover, the color scale indicates that users who listen to more items seem to be less affected by the diversity deterioration. On the other hands, the model seems to increase the diversity of users who already listened to a diverse set of items. This tends to corroborate the common assumption that the diversity of recommender systems' output is taylored to the user's diversity. Comparing the three different number of recommendations scenarii, we observe that the number of users for whom the diversity increases

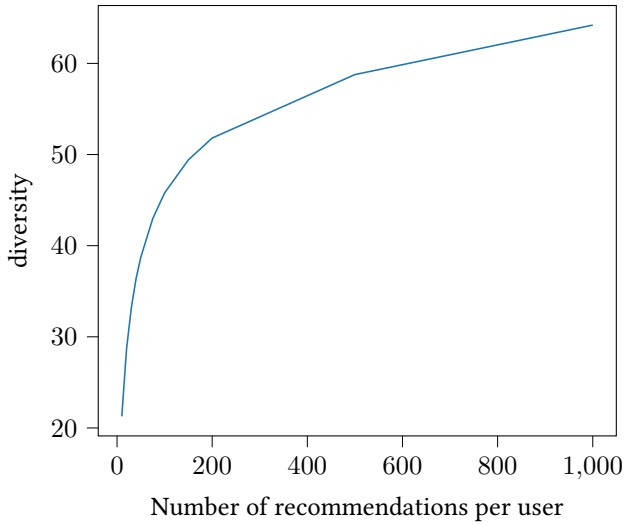
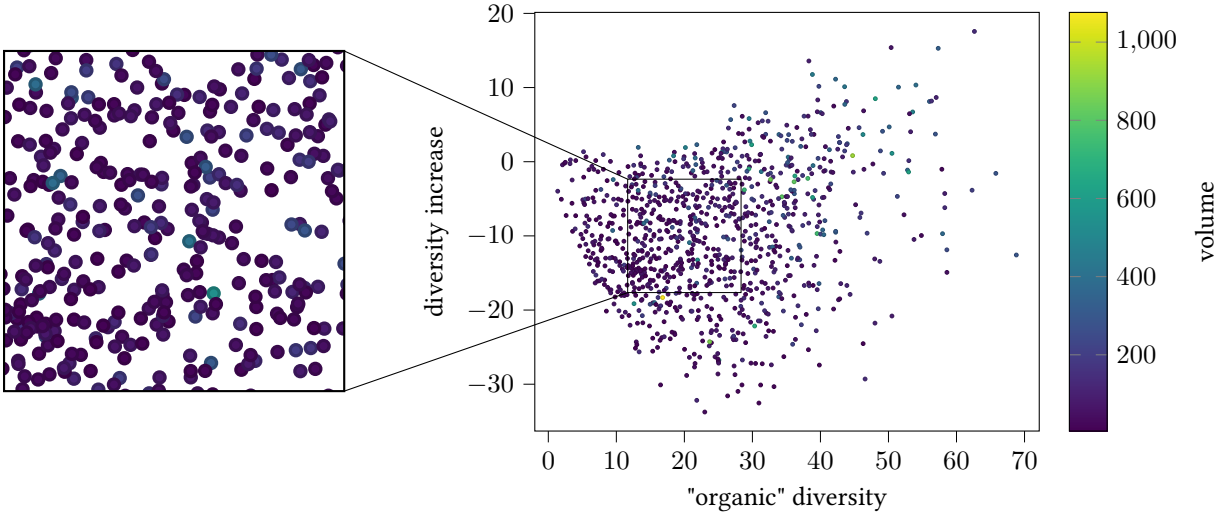


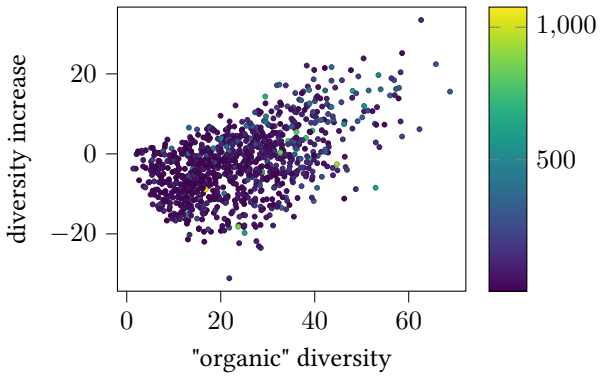
Figure 5: Average recommendations diversity plotted against the number of recommendations generated for each user.

is lower when they listen to more recommendations. This is interesting because our user model (see Equation 9) was designed so that whatever the number of recommendations is, the total user recommendations play count is the same (equal to their the "organic" volume). Therefore, even for the most "organically" diverse users, spreading their listenings to more recommendations does not improve their diversity and can even lower it.

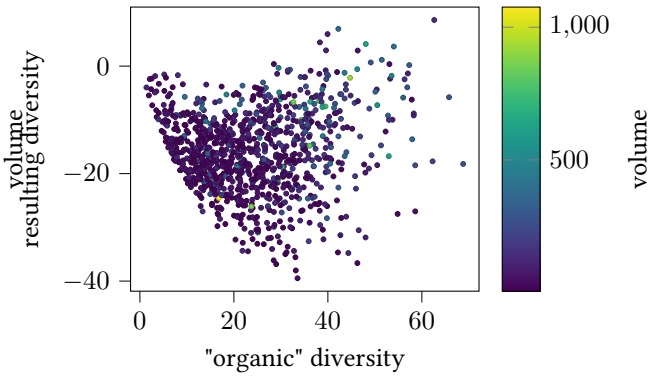
In Figure 5 we plot the average diversity of the recommendations made to users in the test set versus the number of recommendations made. Interestingly, we find a similar result as in [17]. Because of the finite number of tags and songs to recommend, the recommendations' mean diversity cannot increase for ever. This also provides an explanation of the diversity decrease observed in Figure 6 for high numbers of recommendations. In a future work, it would be intersting to compare this model with a baseline model (random or popularity based) and to see if we can act on the selection stage to improve the mean diversity of recommendations.



(a) $r = 50$ recommendations per user



(b) $r = 10$ recommendations per user



(c) $r = 500$ recommendations per user

Figure 6: User diversity increase after recommendation with respect to their original diversity. The color of each user's dot represent the user volume of listening.

5 Conclusion

In this paper we investigated the question of quantifying the diversity of recommender systems, building upon previous work on diversity in heterogeneous networks. We introduced a method to quantify offline the diversity of recommender systems via their recommendations. Focusing on the case of collaborative filtering, we applied our method to study the impact of this recommender system on user diversity. We discovered that the studied recommender system influenced the diversity of users differently depending on their "organic" diversity. Finally, we pointed out that the more recommendations the users listened to, the more their diversity was decreased.

This work has the potential to be expanded on in several directions. The first obvious development would be to apply our method on different models such as the more recent Neural Collaborative Filtering or embedding-based models. Then, it would be interesting to replace the MSD dataset by a more controlled dataset. In fact, this dataset was exctracted from the *Echo Nest* platform which already had the ability to make personalized recommendations. Finally, in order to test the robustness of the method, it would be interesting to study the effect of the user model on the the diversity results.

References

[1] Charu C. Aggarwal. *Recommender Systems*. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-29657-9 978-3-319-29659-3. DOI: 10.1007/978-3-319-29659-3.

[2] Ashton Anderson et al. "Algorithmic Effects on the Diversity of Consumption on Spotify". In: *Proceedings of The Web Conference 2020*. WWW '20: The Web Conference 2020. Taipei Taiwan: ACM, Apr. 20, 2020, pp. 2155–2165. ISBN: 978-1-4503-7023-3. DOI: 10.1145/3366423.3380281.

[3] Thierry Bertin-Mahieux et al. "The Million Song Dataset". In: (2011), pp. 591–596. DOI: 10.7916/D8NZ8J07.

[4] Paul Covington, Jay Adams, and Emre Sargin. "Deep Neural Networks for YouTube Recommendations". In: *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA, 2016.

[5] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. "Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches". In: *Proceedings of the 13th ACM Conference on Recommender Systems* (Sept. 10, 2019), pp. 101–109. DOI: 10.1145/3298689.3347058. arXiv: 1907.06902.

[6] Michael D. Ekstrand. "Collaborative Filtering Recommender Systems". In: *Foundations and Trends® in Human–Computer Interaction* 4.2 (2011), pp. 81–173. ISSN: 1551-3955, 1551-3963. DOI: 10.1561/1100000009.

[7] Michael D. Ekstrand. "LensKit for Python: Next-Generation Software for Recommender System Experiments". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Oct. 19, 2020), pp. 2999–3006. DOI: 10.1145/3340531.3412778. arXiv: 1809.03125.

[8] Montacer Essid and Justin Solomon. *Quadratically-Regularized Optimal Transport on Graphs*. Mar. 23, 2018. arXiv: 1704.08200 [cs, math]. URL: http://arxiv.org/abs/1704.08200 (visited on 01/26/2021).

[9] David Goldberg et al. "Using Collaborative Filtering to Weave an Information Tapestry". In: *Communications of the ACM* 35.12 (Dec. 1992), pp. 61–70. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/138859.138867.

[10] Casper Hansen et al. "Contextual and Sequential User Embeddings for Large-Scale Music Recommendation". In: *Fourteenth ACM Conference on Recommender Systems*. RecSys '20: Fourteenth ACM Conference on Recommender Systems. Virtual Event Brazil: ACM, Sept. 22, 2020, pp. 53–62. ISBN: 978-1-4503-7583-2. DOI: 10.1145/3383313.3412248.

[11] Xiangnan He et al. *Neural Collaborative Filtering*. Aug. 25, 2017. arXiv: 1708.05031 [cs]. URL: http://arxiv.org/abs/1708.05031 (visited on 09/23/2020).

[12] Natali Helberger. "On the Democratic Role of News Recommenders". In: *Digital Journalism* 7.8 (Sept. 14, 2019), pp. 993–1012. ISSN: 2167-0811, 2167-082X. DOI: 10.1080/21670811.2019.1623700.

[13] Natali Helberger, Kari Karppinen, and Lucia D'Acunto. "Exposure Diversity as a Design Principle for Recommender Systems". In: *Information, Communication & Society* 21.2 (Feb. 2018), pp. 191–207. ISSN: 1369-118X, 1468-4462. DOI: 10.1080/1369118X.2016.1271900.

[14] Yifan Hu, Yehuda Koren, and Chris Volinsky. "Collaborative Filtering for Implicit Feedback Datasets". In: *2008 Eighth IEEE International Conference on Data Mining*. 2008 Eighth IEEE International Conference on Data Mining (ICDM). Pisa, Italy: IEEE, Dec. 2008, pp. 263–272. ISBN: 978-0-7695-3502-9. DOI: 10.1109/ICDM.2008.22.

[15] Mozhgan Karimi, Dietmar Jannach, and Michael Jugovac. "News Recommender Systems – Survey and Roads Ahead". In: *Information Processing & Management* 54.6 (Nov. 2018), pp. 1203–1227. ISSN: 03064573. DOI: 10.1016/j.ipm.2018.04.008.

[16] Pedro Ramaciotti Morales et al. *Measuring Diversity in Heterogeneous Information Networks*. Dec. 16, 2020. arXiv: 2001.01296 [cs, math]. URL: http://arxiv.org/abs/2001.01296 (visited on 01/27/2021).

[17] Rémy Poulain and Fabien Tarissan. "Investigating the Lack of Diversity in User Behavior: The Case of Musical Content on Online Platforms". In: *Information Processing & Management* 57.2 (Mar. 2020), p. 102169. ISSN: 03064573. DOI: 10.1016/j.ipm.2019.102169.

[18] Markus Schedl, Peter Knees, and Fabien Gouyon. "New Paths in Music Recommender Systems Research". In: *Proceedings of the Eleventh ACM Conference on Recommender Systems*. RecSys '17: Eleventh ACM Conference on Recommender Systems. Como Italy: ACM, Aug. 27, 2017, pp. 392–393. ISBN: 978-1-4503-4652-8. DOI: 10.1145/3109859.3109934.

[19] Gábor Takács, István Pilászy, and Domonkos Tikk. "Applications of the Conjugate Gradient Method for Implicit Feedback Collaborative Filtering". In: *Proceedings of the Fifth ACM Conference on Recommender Systems - RecSys '11*. The Fifth ACM Conference. Chicago, Illinois, USA: ACM Press, 2011, p. 297. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043987.

[20] Saúl Vargas and Pablo Castells. "Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems". In: *Proceedings of the Fifth ACM Conference on Recommender Systems - RecSys '11*. The Fifth ACM Conference. Chicago, Illinois, USA: ACM Press, 2011, p. 109. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043955.

[21] Isaac Waller and Ashton Anderson. "Generalists and Specialists: Using Community Embeddings to Quantify Activity Diversity in Online Platforms". In: (2019), p. 11.

[22] Shuai Zhang et al. "Deep Learning Based Recommender System: A Survey and New Perspectives". In: *ACM Computing Surveys* 52.1 (Feb. 28, 2019), pp. 1–38. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3285029. arXiv: 1707.07435.