SONG RECOMMENDATION WITH NON-NEGATIVE MATRIX FACTORIZATION AND GRAPH TOTAL VARIATION

Kirell Benzi, Vassilis Kalofolias, Xavier Bresson and Pierre Vandergheynst

Signal Processing Laboratory 2 (LTS2), Swiss Federal Institute of Technology (EPFL)

ABSTRACT

This work formulates a novel song recommender system[1] as a matrix completion problem that benefits from collaborative filtering through Non-negative Matrix Factorization (NMF) and content-based filtering via total variation (TV) on graphs. The graphs encode both playlist proximity information and song similarity, using a rich combination of audio, meta-data and social features. As we demonstrate, our hybrid recommendation system is very versatile and incorporates several well-known methods while outperforming them. Particularly, we show on real-world data that our model overcomes w.r.t. two evaluation metrics the recommendation of models solely based on low-rank information, graph-based information or a combination of both.

***Index Terms*—** Recommender system, graphs, NMF, total variation, audio features

1. INTRODUCTION

Recommending movies on Netflix, friends on Facebook, or jobs on LinkedIn are tasks gaining an increasing interest over the last years. Low-rank matrix factorization techniques [1] where amongst the winners of the famous Netflix prize, involving explicit user ratings as input. Similar techniques were soon used in order to solve implicit feedback problems, where item preferences were implied for example by the actions of a user [2, 3]. Specifically regarding songs and playlists recommendation, various techniques have been proposed, ranging from pure content-based methods [4] to hybrid models [5]. A comprehensive review of related algorithms can be found in [6, 7]. Recently, graph regularization was proposed in order to enhance the quality of matrix completion problems [8–10]. The contributions of this paper are as follows:

- A mathematically sound hybrid system that benefits from collaborative and content-based filtering.
- The introduction of a new graph regularization term (TV) [11] in the context of recommendation that outperforms the widely used Tikhonov regularization. [9, 10],
- A well-defined iterative optimization scheme based on proximal splitting methods [12].

[1] The code is available at: https://github.com/kikohs/recog

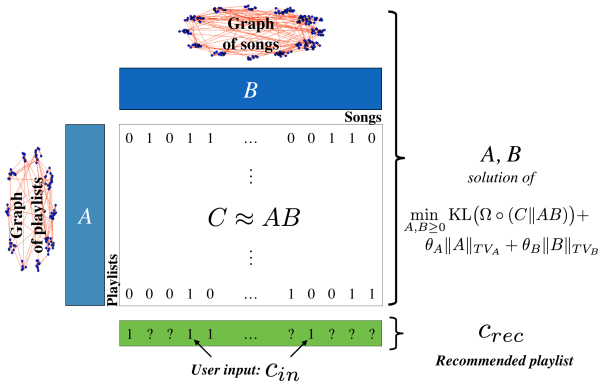

Fig. 1. The architecture of our playlist recommender system.

Numerical experiments demonstrate the performance of our proposed recommender system.

2. OUR RECOMMENDATION ALGORITHM

Suppose we are given n playlists, each containing some of m songs. We define matrix $C \in \{0,1\}^{n\times m}$ as in [3,13], that has a value $C_{ij} = 1$ if playlist i contains song j, 0 otherwise. We also define a weight mask $\Omega \in \{\varepsilon, 1\}^{n\times m}$ that has a "confidence" value $\Omega_{ij} = 1$ one if the entry C_{ij} is 1, and a small value ε, otherwise (we use $\varepsilon = 0.1$). This follows the example of implicit feedback problems [2], since a zero in matrix C does not mean that the corresponding song is irrelevant to the playlist, but that it is less probably relevant.

The goal of the training step is to find an approximate low-rank representation $AB \approx C$, where $A \in \mathbb{R}_+^{n\times r}$, $B \in \mathbb{R}_+^{r\times m}$ non-negative and with small r. This problem is known as Non-Negative Matrix Factorization (NMF) and has drawn a lot of attention after the seminal work [14]. The advantage of NMF over other factorization techniques is that the approximation is only based on adding factors, a property explained as *learning the parts of objects* [14], in this case the playlists. NMF comes to the cost of being NP-hard [15], so sophisticated regularization is important for finding a good local minimum. In our problem we use outside information given by the songs and playlists graphs to give structure to the factors

A and B. Our model is formulated as

$$\min_{A,B\geq 0} \mathrm{KL}\big(\Omega \circ (C\|AB)\big) + \theta_A\|A\|_{TV_A} + \theta_B\|B\|_{TV_B}, \quad (1)$$

where $\circ$ is the pointwise multiplication operator and $\theta_A, \theta_B \in \mathbb{R}_+$. We use a weighted Kullback-Leibler (KL) divergence as a distance measure between C and AB, that has been shown to be more accurate than the Frobenius norm for various NMF settings [16]. The second term is the TV of the rows of A on the playlists graph, so penalizing it promotes piecewise constant signals [11]. Similarly with the third term for columns of B. Eventually, the proposed model leverages the works of [9, 16], and extends them to graphs using the TV seminorm.

Graph Regularization with Total Variation. In our NMF-based recommender, each playlist i is represented in a low-dimensional space by a row A_i of the matrix A. In order to learn better low-rank representations A_i of the playlists, we also impose the pairwise similarities of the playlists $w^A_{ii'}$ on their corresponding low-rank representations. We can see this from the definition of the TV regularization term, $\|A\|_{TV_A} = \frac{1}{2}\sum_i \sum_{i'\sim i} w^A_{ii'}\|A_i - A_{i'}\|_1$. Hence, when two playlists i, i' are similar then they are also well-connected on the graph and the weight of the edge connecting these two playlists $w^A_{ii'}$ is large (here $w^A_{ii'} \approx 1$). Moreover, any large distance between the corresponding low-dimensional representation vectors $(A_i, A_{i'})$ is penalized, forcing $(A_i, A_{i'})$ to stay close in the low-dimensional space. In a similar way, each song j is represented in a low-dimensional space by a column B_j of the matrix B. If two songs (j, j') are close ($w^B_{jj'} \approx 1$), so will be $(B_j, B_{j'})$ with the graph regularization $\|B\|_{TV_B}$.

A similar idea has been used in [10] by incorporating the graph information through Tikhonov regularization, i.e. with the Dirichlet energy term $\frac{1}{2}\sum_i \sum_{i'\sim i} w^A_{ii'}\|A_i - A_{i'}\|_2^2$. However, the latter promotes smooth changes between the columns of A, while the graph TV term penalization promotes piecewise constant signals with potentially sharp transitions between columns A_i and $A_{i'}$. This is advantageous in applications where well separated classes are sought, for example in clustering [17], or in our recommendation system where similar playlists might belong to different categories.

As we demonstrate in Sec. 4, the use of the graphs of songs and playlists improve significantly the recommendations, while the results are better when the more forgiving TV term is used instead of Tikhonov regularization.

Primal-dual optimization. Optimization problem (1) is globally non-convex, but separately convex w.r.t. A and B. A standard strategy is thus to optimize B for fixed A, then optimize A for fixed B, and repeat until convergence. We describe here the proposed optimization algorithm w.r.t. B for fixed A based on [12, 16, 18]. The same algorithm can be applied to A for fixed B. Let us rewrite problem (1) as:

$$\min_{B\geq 0} \; F(AB) + G(K_B B), \quad (2)$$

where

$$F(AB) = \mathrm{KL}\big(\Omega \circ (C\|AB)\big) = \quad (3)$$
$$\sum_{i=1}^{m}\sum_{j=1}^{n}\Big(-\Omega_{ij}C_{ij}\Big(\log\frac{(AB)_{ij}}{C_{ij}} + 1\Big) + \Omega_{ij}(AB)_{ij}\Big),$$
$$G(K_B B) = \theta_B\|B\|_{TV_B} = \theta_B\|K_B B\|_1, \quad (4)$$

where $K_B \in \mathbb{R}^{n_e\times m}$ is the graph gradient operator [17], with n_e being the number of edges in the graph of B. Using the conjugate functions $F^\star$ and $G^\star$ of F and G, (2) is equivalent to the saddle-point problem:

$$\min_{B\geq 0}\max_{Y_1,Y_2} \; \mathrm{tr}((AB)^T\cdot Y_1) - F^\star(Y_1) + \mathrm{tr}((KB^T)^T\cdot Y_2) - G^\star(Y_2), \quad (5)$$

where $Y_1 \in \mathbb{R}^{n\times m}$, $Y_2 \in \mathbb{R}^{n_e\times r}$. Let us now introduce the proximal terms and the time steps $\sigma_1, \sigma_2, \tau_1, \tau_2$:

$$\min_{B\geq 0}\max_{Y_1,Y_2} \; \mathrm{tr}((AB)^T\cdot Y_1) - F^\star(Y_1) + \mathrm{tr}((KB^T)^T\cdot Y_2) - G^\star(Y_2) + \frac{\tau_1+\tau_2}{2\tau_1\tau_2}\|B - B^k\|_F^2 - \frac{1}{2\sigma_1}\|Y_1 - Y_1^k\|_F^2 - \frac{1}{2\sigma_2}\|Y_2 - Y_2^k\|_F^2. \quad (6)$$

The iterative scheme is thus for $k \geq 0$:

$$Y_1^{k+1} = \mathrm{prox}_{\sigma_1 F^\star}(Y_1^k + \sigma_1 AB^k), \quad (7)$$
$$Y_2^{k+1} = \mathrm{prox}_{\sigma_2 G^\star}(Y_2^k + \sigma_2 K_B B^k), \quad (8)$$
$$B^{k+1} = (B^k - \tau_1 A^T Y_1^{k+1} - \tau_2(K_B^T Y_2^{k+1})^T)_+, \quad (9)$$

where prox is the proximal operator [12] and $(\cdot)_+ = \max(\cdot, 0)$. For our problem we have chosen the standard Arrow-Hurwicz time steps $\sigma_1 = \tau_1 = 1/\|A\|$ and $\sigma_2 = \tau_2 = 1/\|K\|$, where $\|\cdot\|$ is here the operator norm.

The proximal solutions (7) and (8) are given by:

$$\mathrm{prox}_{\sigma_1 F^\star}(Y) = \frac{1}{2}\Big(Y + \Omega - \sqrt{(Y-\Omega)^2 + 4\sigma_1\Omega\circ C}\Big)$$
$$\mathrm{prox}_{\sigma_2 G^\star}(Y) = Y - \mathrm{shrink}(Y, \theta_B/\sigma_2), \quad (10)$$

where shrink is the soft shrinkage operator [19]. Note that the same algorithm could be used for Tikhonov regularization, i.e. replacing $\|K_B B\|_1$ by $G(K_B B) = \frac{\theta_B}{2}\|K_B B\|_2^2$ by just changing the first proximal (10) to $\mathrm{prox}_{\sigma_2 G^\star}(Y) = \frac{\theta_B}{\sigma_2+\theta_B}Y$. In [10] this regularization is used along with a symmetric version of the KL divergence, however the latter has no analytic solution unlike the one we use in this work. As a result their objective function does not fit an efficient primal dual optimization scheme like the one we propose. We thus choose to keep the non symmetric KL model, denoted as GNMF in this paper, in order to compare the TV versus Tikhonov regularization.

Recommending songs. Once we have learned matrices A

and B by solving (1), we wish to recommend a new playlist c_{rec} given a few songs c_{in} (see Fig. 1). We also want to make real-time recommendations, so we design here a fast recommender function as follows:

Given the songs c_{in}, we first find a good representation of the query on the learned low-rank space of playlists by solving a regularized least squares problem: $a_{in} = \arg\min_{a \in \mathbb{R}^{1\times r}} \|\Omega_{in} \circ (c_{in} - aB)\|_2^2 + \varepsilon \|a\|_2^2$. The latter enjoys an analytic solution $a_{in} = (B^T \Omega_{in} B + \varepsilon I)^{-1} (B^T \Omega_{in} c_{in})$ that is cheap to compute as r is small (we use $\varepsilon = 0.01$).

The recommended playlist can benefit from the playlists that have similar representations as the one of the query, thus we use the weighted sum $a_{rec} = \sum_{i=1}^{n} w_i A_i / \sum_{i=1}^{n} w_i$ as the representation of the recommended playlist in the low dimensional space. Here the weights w_i are defined as $w_i = e^{-\|a_{in} - A_i\|_2^2 / \sigma^2}$ and depend on the distance of a_{in} from other playlists representations, while $\sigma = \text{mean}_i(\{\|a_{in} - A_i\|_2\}_{i=1}^{n})/4$. The final recommended playlist uses the low-rank representation a_{rec}:

$$c_{rec} = a_{rec} B. \tag{11}$$

Note finally that the recommended playlist c_{rec} is not binary, but with continued values that serve as song rankings.

3. GRAPHS OF PLAYLISTS AND SONGS

Playlists Graph. The playlists graph naturally encodes pairwise similarities between playlists. The set of nodes of this graph is the set of playlists and the edge weight provides the proximity between two playlists. A large weight (here $w_{ii'}^A \approx 1$) implies a strong proximity between the playlists. In this work, the edge weight of the playlists graph uses both "outside" information, i.e. the meta-data, and "inside" information, i.e. the songs that form the playlists. As meta-data, we use the predefined Art of the Mix playlist categories [20] onto which users label their mixes. The edge weight of the playlists graph is thus defined as follows:

$$w_{ii'}^A = \gamma_1 \delta_{cat\{i\}=cat\{i'\}} + \gamma_2 \, \text{sim}_{\cos}(C_i, C_{i'}),$$

where cat stands for playlist category, C_i is the i^{th} row of matrix C and $\text{sim}_{\cos}(p, q) = p^\top q / (\|p\|.\|q\|)$ is the cosine similarity distance between the vectors of the songs of the two playlists. In our case, the cosine similarity is the ratio between the songs in common and the square root of the product of the lengths of the two playlists. The two positive parameters γ_1, γ_2 with $\gamma_1 + \gamma_2 = 1$ allow to weight the importance of the playlist labels against their element-wise similarity. To control the edge density in each category and to give more flexibility to our recommendation model, we keep a random subset of 20% of the edges between nodes of the same category. As we find experimentally, $\gamma_2 = 0.3$ constitutes a good compromise, see Sec. 4.

The quality of the playlist graph is measured by partitioning the graph using the standard Louvain's method [21]. The number of partitions is automatically given by the modularity dendrogram which is cut where the modularity is maximal. The graph used in Sec. 4 has a modularity of 0.63 when using the cosine similarity ($\gamma_2 = 0$) only. If we add the meta-data information by connecting 20% of all playlist pairs within each category with $\gamma_2 = 0.3$, the modularity increases to 0.82.

High Level Features	
acousticness	Acoustic or electric?
valence	Is the song positive or negative?
energy	How energetic is the song?
liveness	Is it a "live" recording?
speechiness	How many spoken words?
danceability	Is the song danceable?
tempo	Normalized BPM.
instrumentalness	Is the song instrumental?
Social Features	
artist discovery	How unexpectedly popular is the artist?
artist familiarity	How familiar is the artist?
artist hotttnesss	Is the artist currently popular?
song hotttnesss	Is the song currently popular?
song currency	How recently has it become popular?
Temporal Echonest Features	
statistics on echonest segments	Described in [22]
Metadata Features	
genre	ID3 genre extracted from tags given by LastFM api

Table 1. The features used to create graph of songs.

Songs Graph. The second graph used in our model is the graph of song similarity. It is created from a mixture of Echonest features extracted from the audio signal which we combine with meta-data information and social features for the track. Table 1 gives a view of the features used to create the song graph.

In order to improve the quality of our audio features, we trained a Large Margin Nearest Neighbors model [23] on the song genres extracted from the LastFm associated terms (tags). To extract real music genres we use the Levenshtein distance between those terms weighted by their popularity (according to LastFm) and the music genres defined in the ID3 tags.

Eventually, the songs graph is created using the k nearest neighbors (here k = 5) where the edge weight between two songs j, j' is given by $w_{jj'}^B = \exp(-\|x_j - x_{j'}\|_1 / \sigma)$ for j' in the k^{th} nearest neighbors of j. The parameter σ acts as the scale parameter of the graph and is set to be the average distance of the k^{th} neighbors. The obtained graph has a high modularity (0.64) and is quite pure with respect to song genres with around 65% of accuracy using an unsupervised k-NN classifier.

4. EXPERIMENTAL RESULTS

In this section we validate our approach by comparing our model against three different recommender systems on a real world dataset. Our test dataset is extracted from the Art-of-the-Mix corpus created by McFee and al. in [20] onto which we extract the previously described features.

Assessing the quality of any music recommender systems is well-known to be a challenging problem [7]. In this work, we use a typical metric for recommender system with implicit feedback, *Mean Percentage Ranking (MPR)* described in [2] and the *playlist category accuracy*, that is the percentage of the recommended songs that have already been used in playlists from the requested category in the past.

Models. We first compare our model against a graphs-only based approach, labeled as *Cosine only*. For a given input, this model computes the t-closest playlists (here $t = 50$) using cosine similarity. Songs are recommended by computing a histogram of all the songs contained in these playlists weighted by the cosine similarity weight, as defined by eq. (11). The second model is NMF using KL divergence, labeled *NMF [16]*. The last model, *GNMF [10]* described in Sec. 2, is based on the KL divergence with Tikhonov regularization using the graphs of our model.

Queries. We test our model with three different types of queries. In all cases, a query c_{test} contains $s = 3$ input songs, and the system returns the top $k = 30$ output songs as a playlist using eq. (11). The first type of queries, *Random*, contains completely randomly chosen songs from all categories and is solely used as a comparison baseline. The second type of queries, *Test*, picks randomly 3 songs from a playlist of the test set. Lastly, *Sampled*, contains randomly chosen songs from a given category. It simulates a recommender system based on chosen playlist categories input by a user.

Training. We train our model using a randomly selected subset of 70% of the playlists. As our model is not jointly convex, initialization may change the performance of the system, so we use the nowadays standard technique of NNDSVD [24] to get a good approximate solution. In all our experiments a value of the rank $r = 15$ performs well, which is expected as each row has between 5 and 20 non-zero values. The best set of parameters $\theta_A = 18$ and $\theta_B = 1$ is found using a grid search using queries on the validation set. In order to prevent overfitting, we perform *early stopping* as soon as the MPR on the validation set ceases to increase.

Validation set. We create the "playlists" of the validation set by creating artificial queries from the different playlist categories. That is, for each category we randomly pick $s = 3$ songs that have been previously used in user-made playlists labeled by the given category.

Results. The performance in terms of playlist category accuracy and MPR of the different models are reported in Table 2 and Table 3 respectively. As expected, for random category queries all models fail to return playlists from the categories of the input songs. At the same time, the performance of NMF as collaborative filtering without the graphs information is poor. This can be explained by the sparsity of the dataset, that only contains 5 to 20 non-zero elements per row, i.e. only 0.11-0.46% sparsity. Collaborative filtering models are known to perform better as more observed ratings are available [9]. The cosine model performs better in terms of category accuracy, as it directly uses the cosine distance between the input query and playlists from pure categories. However, its high MPR value shows that our model, albeit more complex, achieves better song recommendations.

	Cosine only	NMF [16]	GNMF [10]	$\gamma_1 = 0$ $\gamma_2 = 1$	$\gamma_1 = 0.3$ $\gamma_2 = 0.7$
Random	0.135	0.150	0.167	0.210	0.183
Test	0.530	0.236	0.332	0.544	**0.646**
Sampled	0.822	0.237	0.366	0.598	**0.846**

Table 2. Category accuracy for all models for different types of 3-song queries (higher is better). Results are averaged over 10 train/validation runs with 300 queries each.

	Cosine only	NMF [16]	GNMF [10]	$\gamma_1 = 0$ $\gamma_2 = 1$	$\gamma_1 = 0.3$ $\gamma_2 = 0.7$
Test	0.208	0.248	0.181	0.153	**0.146**
Sampled	0.226	0.319	0.211	0.164	**0.074**

Table 3. Mean percentage ranking (MPR) for all models for different types of 3-song queries (lower is better). Results are averaged over 10 train/validation runs with 300 queries each.

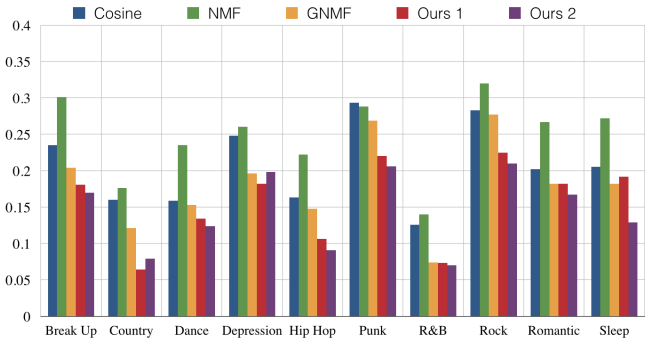

Fig. 2. MPR for each playlist category on the test set. Our models use the same parameters of Table 3. Ambiguous categories such as Rock, Punk have the highest MPR on the test set. Our model outperforms significantly the others methods on those specific categories.

5. CONCLUSION

In this work we introduce a novel flexible song recommender system that combines collaborative filtering with playlist and song proximity information encoded by graphs. We use a primal-dual based optimization scheme to achieve a highly parallelizable algorithm with the potential to scale up to very large datasets. We choose graph TV instead of Tikhonov regularization and demonstrate the model's superiority by comparing our system against three other recommendation models on a real music playlists dataset.

6. REFERENCES

[1] Yehuda Koren, Robert Bell, and Chris Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 8, pp. 30–37, 2009.

[2] Yifan Hu, Yehuda Koren, and Chris Volinsky, "Collaborative filtering for implicit feedback datasets," in *In IEEE International Conference on Data Mining (ICDM 2008*, 2008, pp. 263–272.

[3] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian Personalized Ranking from Implicit Feedback," in *Proceedings of Conference on Uncertainty in Artificial Intelligence*, 2009, pp. 452–461.

[4] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen, "Deep content-based music recommendation," in *Advances in Neural Information Processing Systems*, 2013, pp. 2643–2651.

[5] Bo Shao, Dingding Wang, Tao Li, and Mitsunori Ogihara, "Music recommendation based on acoustic features and user access patterns," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 17, no. 8, pp. 1602–1611, 2009.

[6] Oscar Celma, "Music recommendation," in *Music Recommendation and Discovery*, pp. 43–85. Springer Berlin Heidelberg, 2010.

[7] Geoffray Bonnin and Dietmar Jannach, "Automated Generation of Music Playlists: Survey and Experiments," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, pp. 1–35, nov 2014.

[8] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King, "Recommender systems with social regularization," in *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 2011, pp. 287–296.

[9] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Matrix Completion on Graphs," *arXiv*, vol. preprint arXiv:1408.1717, 2014.

[10] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang, "Graph regularized nonnegative matrix factorization for data representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 8, pp. 1548–1560, 2011.

[11] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear Total Variation Based Noise Removal Algorithms," *Physica D*, vol. 60(1-4), pp. 259 – 268, 1992.

[12] P.L. Combettes and J.C. Pesquet, "Proximal Splitting Methods in Signal Processing," *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pp. 185–212, 2011.

[13] N. Hariri, B. Mobasher, and R. Burke, "Context-Aware Music Recommendation based on Latent Topic Sequential Patterns," in *Proceedings of ACM conference on Recommender systems*, 2012, pp. 131–138.

[14] Daniel D Lee and H Sebastian Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[15] Stephen A Vavasis, "On the complexity of nonnegative matrix factorization," *SIAM Journal on Optimization*, vol. 20, no. 3, pp. 1364–1377, 2009.

[16] F. Yanez and F. Bach, "Primal-Dual Algorithms for Non-negative Matrix Factorization with the Kullback-Leibler Divergence," *arXiv:1412.1788*.

[17] X. Bresson, T. Laurent, D. Uminsky, and J.H. von Brecht, "Multiclass Total Variation Clustering," *Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 1421–1429, 2013.

[18] A. Chambolle and T. Pock, "A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging," *Journal of Mathematical Imaging and Vision*, vol. 40(1), pp. 120–145, 2011.

[19] D. Donoho, "De-Noising by Soft-Thresholding," *IEEE Transactions on Information Theory*, vol. 41(33), pp. 613–627, 1995.

[20] Brian McFee and Gert RG Lanckriet, "Hypergraph Models of Playlist Dialects.," in *ISMIR*. Citeseer, 2012, pp. 343–348.

[21] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, pp. P10008, 2008.

[22] Alexander Schindler and Andreas Rauber, "Capturing the temporal domain in echonest features for improved classification effectiveness," in *Adaptive Multimedia Retrieval: Semantics, Context, and Adaptation*, pp. 214–227. Springer, 2014.

[23] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul, "Distance metric learning for large margin nearest neighbor classification," in *Advances in neural information processing systems*, 2005, pp. 1473–1480.

[24] Christos Boutsidis and Efstratios Gallopoulos, "SVD Based Initialization: A Head Start For Nonnegative Matrix Factorization," *Pattern Recognition*, vol. 41, no. 4, pp. 1350–1362, 2008.