TAG-AWARE SPECTRAL CLUSTERING OF MUSIC ITEMS

Ioannis Karydis[i] **Alexandros Nanopoulos**[ii] **Hans-Henning Gabriel**[iii] **Myra Spiliopoulou**[iii]
[i]Department of Informatics, Ionian University, Greece
[ii]Institute of Informatics, Hildesheim University, Germany
[iii]Faculty of Computer Science, Otto-von-Guericke-University Magdeburg, Germany
karydis@ionio.gr, nanopoulos@ismll.de, {hgabriel, myra}@iti.cs.uni-magdeburg.de

ABSTRACT

Social tagging is an increasingly popular phenomenon with substantial impact on Music Information Retrieval (MIR). Tags express the personal perspectives of the user on the music items (such as songs, artists, or albums) they tagged. These personal perspectives should be taken into account in MIR tasks that assess the similarity between music items. In this paper, we propose an novel approach for clustering music items represented in social tagging systems. Its characteristic is that it determines similarity between items by preserving the 3-way relationships among the inherent dimensions of the data, i.e., users, items, and tags. Conversely to existing approaches that use reductions to 2-way relationships (between items-users or items-tags), this characteristic allows the proposed algorithm to consider the personal perspectives of tags and to improve the clustering quality. Due to the complexity of social tagging data, we focus on spectral clustering that has been proven effective in addressing complex data. However, existing spectral clustering algorithms work with 2-way relationships. To overcome this problem, we develop a novel data-modeling scheme and a tag-aware spectral clustering procedure that uses tensors (high-dimensional arrays) to store the multigraph structures that capture the personalised aspects of similarity. Experimental results with data from Last.fm indicate the superiority of the proposed method in terms of clustering quality over conventional spectral clustering approaches that consider only 2-way relationships.

1. INTRODUCTION

Music Information Retrieval (MIR) is highly interdisciplinary a field that, due to the nature of music, requires an increased amount of contextual information for most of its processes [1]. One popular method that supplies this contextual information is the practice of *social-tagging*. Social tags are shared, free-text keywords that web users can assign to music items, such as artists, albums, songs, playlists, genres, etc. The popularity of music tagging rests with the easy and effective organisation it produces, in contrast to the obscure and ambiguous hierarchical classification (in terms of genre, mood, etc). Social tagging assists the retrieval of items and social expression of taste [2]. Therefore, tags over music items reflect conveniently the personalised opinion of users for these items.

Social tagging attracts increasing attention and MIR systems like Last.fm [3] and MyStrands [4] contain a body of collected data in which data mining is challenging and promising. One of the most essential data mining tasks is the clustering of music data to assist their organisation, the creation of playlists and the model-based music recommendation. However, several existing MIR approaches consider clustering of data based solely on features extracted directly from the audio. In contrast, the proposed approach is based on user-generated content, in the form of tags, in order to include contextual information that would be otherwise non-extractable from the content of items.

Data from social-tagging systems have 3 inherent dimensions: the users, the music items, and the tags. Moreover, they contain 3-way relationships of the form items–users–tags between these dimensions. Thus, there is a clear difference between just knowing that a tag has been applied to an item regardless by which users, and knowing the specific users that applied this tag to the item. The reason is that in the latter case the tag expresses the personalised perspective of the specific users on the item. Clustering of music items with existing algorithms requires the suppression of the 3 dimensions and the reduction of their 3-way relationships into 2-way of the form items–users or items–tags. This is because most existing clustering algorithms model the data in 2-dimensional arrays whose rows correspond to items and columns to features. Thus, clustering can be performed over items-users or items-tags arrays, but not without breaking the original 3-way relationships between items-users-tags. However, such approach may incur loss of valuable information contained in the 3-way relationships.

To address the complexity of data from social tagging systems, we focus on the popular family of spectral clustering algorithms. This type of clustering algorithms work on a similarity graph that connects every item to its k nearest-

THE SECOND AUTHOR GRATEFULLY ACKNOWLEDGES THE PARTIAL CO-FUNDING OF HIS WORK THROUGH THE EUROPEAN COMMISSION FP7 PROJECT MYMEDIA (WWW.MYMEDIAPROJECT.ORG) UNDER THE GRANT AGREEMENT NO. 215006.

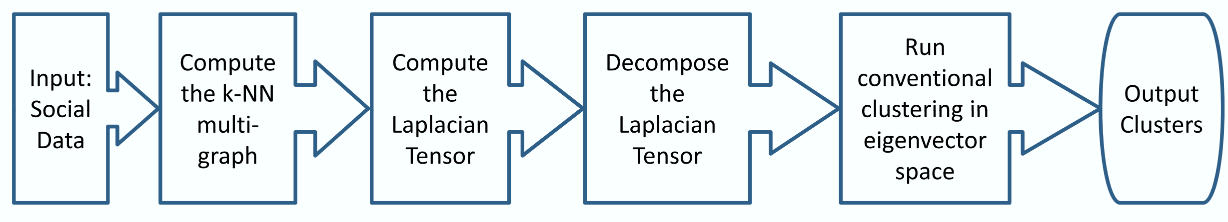

Figure 1. The steps followed by the proposed approach.

neighbors (k-NN) and map each item to a feature space defined by eigenvectors of the similarity graph. Spectral clustering algorithms have been proven effective in addressing complex data [5]. However, existing spectral clustering algorithms cannot be used directly for data from social tagging systems without suppressing the 3 dimensions in order to consider only either items-users or items-tags relationships. The reason is that existing spectral clustering algorithms form the k-NN similarity graph based on the *single value* of similarity between each pair of items.

To overcome the problems of existing approaches and avoid breaking the original 3-way relationships existing in social-tagged data, we propose the extension of spectral clustering in order to become tag-aware and directly handle all present dimensions. Our technical contributions towards this objective are the following: (i) We provide the insight that *multiple similarity values* between each pair of items should be used to account for the fact that when all 3 dimensions are considered, then similarity between two items depends both on the users who tagged them and the tags they assigned, a fact that leads to several similarity values between them. (ii) To support multiple similarity values, we extend the modeling based on k-NN similarity graphs by using k-NN similarity multigraphs, which allow the existence of multiple edges between two nodes. (iii) We extend existing spectral clustering algorithms to consider the k-NN similarity multigraphs by extracting information about eigenvectors from *tensors* (i.e., multidimensional arrays). (iv) We perform experiments with real data crawled from Last.fm and compare the proposed method against conventional spectral clustering that suppresses the original data and consider only 2-way relationships (either items-users or items-tags) in terms of quality of the final clustering.

The rest of this article is organised as follows. Section 2 reviews related work. Section 3 presents an overview of the proposed approach, whereas Section 4 describes the proposed data modeling and Section 5 the proposed clustering algorithm. Experimental results are detailed in Section 6. Finally, Section 7 concludes the article.

2. RELATED WORK

Clustering tagged music data, as well as their visualisation, has also been the focus of the research of Lehwark et al. [6]. In the interest of discovering new music based on the semantic organisation provided by tags on music data, they propose the use of the Emergent-Self-Organising-Map (ESOM) for the clustering of tagged data. Additionally, they also utilise U-Map in order to provide a visually appealing user interface and an intuitive way of exploring new content. Differently from this approach, we apply spectral clustering in contrast to ESOM while our focus is on multiple pairwise similarities in contrast to visualisation of the produced clusters.

Levy et al. [7], investigate the performance of models for varying latent dimensions examining the alteration of low-dimensional semantic representations discriminative capability in searching music collections. This approach is different than the one presented in our work, as we focus on multiple pairwise similarities on the music data for the purpose of clustering the music items, in contrast to [7] where different models are tested in order to uncover emergent semantics from social tags for music.

The clustering of music data has received extensive attention from the MIR community. Most research aims in genre classification (readers are suggested [8] for a detailed survey of the area) as the classification emerging is based on objective similarity measures from the data, thus avoiding the constraints possed by fixed taxonomies, which may be difficult to define as well as suffer from ambiguities and inconsistencies. Using a set of extracted features from the content of the music data, and a similarity measure for the comparison of the data, clustering algorithms organise music data in clusters of similar objects.

Symeonidis et al. [9] proposed dimensionality reduction using higher order SVD for the purposes of personalised music recommendation. That is, given a user and a tag, their purpose is to predict how likely is the user to label a specific music item with this tag. However, conversely from [9] we use tensor factorisation for extracting spectral information and performing spectral clustering, not for predicting recommendations.

3. OVERVIEW OF PROPOSED APPROACH

This section outlines the proposed approach. The steps that will be described in the following are depicted (for reference) in Figure 1.

Existing (non tag-aware) spectral clustering algorithms [5] first compute the k-NN similarity graph, which connects every item with its k-NN. Next, the Laplacian graph of the k-NN similarity graph is used instead, because of the benefits it offers, i.e., it is always positive-semidefinite (allowing its eigenvector decomposition) and the number of times 0 appears as its eigenvalue is the number of connected components in the k-NN similarity graph. Due to these convenient properties, if c clusters are required to be found, spectral clustering algorithms proceed by computing the c eigenvectors that correspond to the c smallest eigenvalues, and represent each original item as a c-dimensional vector whose coordinates are the corresponding values within the c eigenvectors. With this representation, they cluster the c-dimensional vectors using simple algorithms, like k-means or hierarchical agglomerative.

As described in Introduction, differently from conventional spectral clustering algorithms, our proposed approach considers multiple similarity values between each pair of items. In particular, let U be the set of all users. For a given tag t, let $U_1 \subseteq U$ be the set of users that tagged an item i_1 with t, whereas $U_2 \subseteq U$ be the set of users that tagged an item i_2 with t too. We can define a similarity value between i_1 and i_2 as follows. We form two vectors v_1 and v_2, both with $|U|$ elements that are set to 1 at positions that correspond to the users contained in U_1 and U_2, respectively, whereas all rest positions are set to 0. Therefore, the similarity between i_1 and i_2 is given by the cosine measure between the two vectors v_1 and v_2. Since the above process can be repeated for all tags, the result is several similarity values between each pair of items i_1 and i_2. The set of all multiple similarity values are tag-aware and reflect the personalised aspect of similarity perceived by the users (e.g., two users may tag the same item but using entirely different tags).

To account for the various similarity values between each pair of items, we extend (Section 4) the k-NN similarity graph to a k-NN multidigraph that is the union of multiple simple k-NN graphs, one for each distinct tag. The adjacency matrix of a k-NN multidigraph forms a tensor, i.e., a multidimensional array. In order to attain the aforementioned advantages of the Laplacian graph, we propose a method (Section 5.1) to extent towards the construction of the Laplacian multidigraph, whose adjacency matrix is again represented as a tensor. To map each item to a feature space comprised from spectral information extracted from the Laplacian tensor, we describe (Section 5.2) how to use tensor factorisation that extends SVD to multidimensional arrays. Finally, based on the computed features, we describe (Section 5.3) how the clustering is performed. To help comprehension, we use the data from the following example.

Example 1 (Data representation). We assume 3 users that assign tags to 4 music items (henceforth 'items' for simplicity) from a tag-set with 3 tags. Each assignment comprises a triple of the form (user, item, tag). The 9 triples of the example are given in Table 1, whereas we additionally denote (in the first column) the ID of the triple. The corresponding view of the data as tripartite graph is depicted in Figure 2. In this figure, the numbered edges correspond to the triple IDs in Figure 2a. For instance, the first triple (ID = 1) is: Alice tagged Elvis as Classic. In Figure 2 this corresponds to the path consisting of all edges labelled as 1. To avoid cluttering the figure, parallel edges (i.e., edges between the same two nodes) with different labels are depicted as one with different labels separated by comma. In this example, we assume that Elvis and Beatles form one cluster, whereas Mozart and Bach form a second cluster. This follows by observing in Figure 2 that, although users tag items from both clusters, they assign different tags to the first cluster than the second. Therefore, the relationships between users-items alone are not able to determine a clustering structure among the items. In contrast, when considering the multi-way relationships between users-items-tags, we are able to better detect the clustering of items. Although this simple example highlights the advantage of preserving the multi-way relationships compared to considering only item-user relationships, our experimental results show the advantages compared to the consideration of only item-tag relationships, as well.□

ID	User	Item	Tag
1	Alice	Elvis	Classic
2	Bob	Beatles	Classic
3	Bob	Elvis	Classic
4	Bob	Mozart	Symphonic
5	Joe	Mozart	Symphonic
6	Joe	Bach	Symphonic
7	Alice	Mozart	Orchestral
8	Joe	Mozart	Orchestral
9	Joe	Bach	Orchestral

Table 1. Example of input data

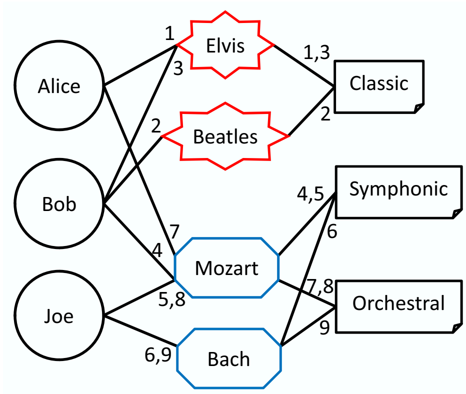

Figure 2. Illustration of the tripartite graph.

4. DATA MODELLING

In this section, we describe the modelling of multiple similarity values with a k-nearest-neighbor multidigraph. A multidigraph is a directed graph permitted to have multiple directed edges (henceforth, simply called edges), i.e., edges with the same source and target nodes.

A tripartite graph (like in the example of Figure 2b) can be partitioned according to the tags. For each tag t, we get the corresponding underlying subgraph B_t, by keeping users and items that participate in triples with this tag.

Each bipartite subgraph is represented with its adjacency matrix B_t ($1 \leq t \leq |T|$), whose size is $|I| \times |U|$; that is, its rows correspond to items and its columns to users. (Henceforth, wherever there is no ambiguity, we use interchangeably the same symbol for a graph and its adjacency matrix.) Each element $B_t(i, u)$ is equal to 1, if there is an edge between the item i and user u, or 0 otherwise. Therefore, from each adjacency matrix B_t we can compute for every pair of items i, j ($1 \leq i, j \leq |I|$), a similarity measure according to the values in their corresponding rows $B_t(i, :)$ and $B_t(j, :)$. Following the widely used approach for 2 dimensional matrixes (like document-term in information retrieval or user-item in CF), we consider the cosine similarity measure between every pair of items.

Having defined a similarity measure, from each subgraph B_t ($1 \leq t \leq |T|$), we can compute the corresponding k-nearest neighbor (k-NN) graph, N_t, which is a labelled and directed graph (digraph). The nodes of each N_t correspond to the items. There is an directed edge between items i and j ($1 \leq i, j \leq |I|$), if j is among the k nearest neighbors of i. Each edge is labelled with the corresponding similarity value.

Considering all k-NN digraphs together, we form the k-NN labelled multidigraph, $\mathcal{N}$, that summarises all multiple similarities. The nodes of $\mathcal{N}$ correspond to the items. The labelled edges of $\mathcal{N}$ is a multiset resulting from the union of the labelled edges of all N_t for $1 \leq t \leq |T|$.

Example 2 (k-NN multidigraph). For the data in Figure 2, the resulting k-NN multidigraph $\mathcal{N}$, for $k = 1$, is depicted in Figure 3a. The multiple edges between the nodes of $\mathcal{N}$ denote the different similarities between the items, according to the different tags. To assist notation, we assume that T_1 denotes the first tag, i.e., Classic, T_2 the second, i.e., Symphonic, and T_3 the third, i.e., Orchestral. In Figure 3a, the edges representing similarities according to tag T_i ($1 \leq i \leq 3$) are annotated with T_i and then follows the corresponding similarity value.[1] Notice that $\mathcal{N}$ correctly captures the clustering structure: edges exist only between items of the same cluster, i.e., between Elvis and Beatles for the first cluster and between Mozart and Bach for the second. Conversely, in Figure 3b, which depicts the k-NN digraph when only user-item relationships are considered, the separation of clusters is not clear. □

5. THE PROPOSED CLUSTERING ALGORITHM

5.1 Constructing the Laplacian Tensor

For each k-NN digraph N_t ($1 \leq t \leq |T|$) of $\mathcal{N}$, compute D_t as a diagonal matrix the diagonal elements of which are defined as follows:

[1] In this small example, to avoid numerical problems, we assign similarity equal to 0 when at least one item has no edge at all in the corresponding bipartite graphs. Moreover, to avoid cluttering the graph, only the non-zero similarities are depicted.

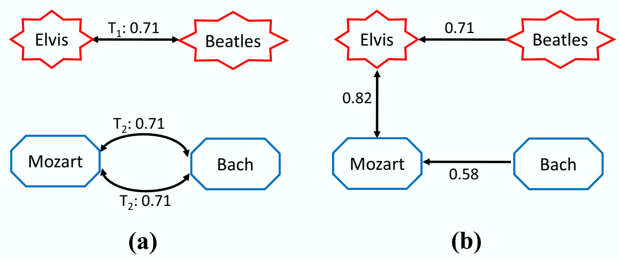

Figure 3. The k-NN multidigraph for the running example.

$$D_t(i, i) = \sum_{j=1}^{|I|} N_t(i, j) \tag{1}$$

The Laplacian matrix, L_t, of each N_t is computed as follows [10]:

$$L_t = \mathbb{1} - D_t^{-1/2} N_t D_t^{-1/2} \tag{2}$$

where $\mathbb{1}$ is the identity matrix.

The Laplacian tensor of $\mathcal{N}$ is therefore defined as $\mathcal{L} \in \mathbb{R}^{|I| \times |I| \times |T|}$, whose elements are given as follows:

$$\mathcal{L}(i, j, t) = L_t(i, j) \tag{3}$$

Thus, each matrix L_t, for $1 \leq t \leq |T|$, comprises a frontal slice in $\mathcal{L}$.

The Laplacian tensor $\mathcal{L}$ has 3 *modes*: the first mode corresponds to the items, the second mode to the neighboring items, and the third mode to the tags. To perform spectral clustering, we are interested in extracting the spectrum of $\mathcal{L}$ for the first mode. This procedure is explained in the section to follow.

5.2 Factorising the Laplacian Tensor

In this subsection, we summarise the factorisation of the Laplacian tensor using Tucker decomposition [11], which is the high-order analogue of the Singular Value Decomposition (SVD) for tensors. The factorisation of the Laplacian tensor will produce the required spectrum of its first (corresponding to items) mode.

First, we define the n-mode product $\mathcal{T} \times_n M$ between a general N-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ and a matrix $M \in \mathbb{R}^{J_n \times I_n}$. The result is an $(I_1 \times I_2 \times \ldots \times I_{n-1} \times J_n \times I_{n+1} \times \ldots \times I_N)$-tensor, whose entries are defined as follows (elements are denoted through their subscript indexes):

$$(\mathcal{T} \times_n M)_{i_1 i_2 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} T_{i_1 i_2 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} M_{j_n i_n} \tag{4}$$

Since $\mathcal{L}$ is a 3-order tensor, we henceforth focus only on 1-mode, 2-mode and 3-mode products.

The Tucker decomposition of the 3-order tensor $\mathcal{L}$ can be written as follows [12]:

$$\mathcal{L} \approx \mathcal{C} \times_1 P_1 \times_2 P_2 \times_3 P_3 \quad (5)$$

The $P_1 \in \mathbb{R}^{|I|\times|I|}, P_2 \in \mathbb{R}^{|I|\times|I|}, P_3 \in \mathbb{R}^{|T|\times|T|}$ are called the mode-1 (items), mode-2 (neighboring items), and mode-3 (tags) projection matrixes, respectively. The 3 projection matrixes contain the orthonormal vectors for each mode, called the mode-1, mode-2 and mode-3 singular vectors, respectively. $\mathcal{C}$ is called the core tensor and has the property of all orthogonality. Nevertheless, unlike SVD for matrixes, $\mathcal{C}$ is not diagonal. Recently, several algorithms have bee proposed to efficiently compute the components of the Tucker decomposition. Due to lack of space, more details about the algorithms and their complexity can be found in a recent survey on tensor factorisation [11].

Having already performed the Tucker decomposition of the Laplacian tensor $\mathcal{L}$, we are interested in the mode-1 singular vectors that are stored in P_1. A frequently followed approach in spectral clustering, when c clusters are required, is to select the c eigenvectors associated to the c smallest eigenvalues [5]. Similarly, we select the c mode-1 singular vectors in P_1 associated to the smallest singular values in the core tensor $\mathcal{C}$.

5.3 Performing the Final Clustering

To find c clusters of items using the c mode-1 singular vectors that were computed and selected during the factorisation of the Laplacian tensor, we apply the following steps: (1) Normalise the c selected mode-1 singular vectors to have norm equal to 1. (2) Form a matrix $X \in \mathbb{R}^{|I|\times k}$, whose columns are the normalised c selected mode-1 singular vectors. (3) Associate each item i to a point x_i whose coordinates are the contents of the i-th row of X. (4) Choose a distance metric for the $(x_i)_{i=1,\ldots,|I|}$ points. (5) Cluster the points $(x_i)_{i=1,\ldots,|I|}$ into c clusters using a conventional clustering algorithm, according to the chosen distance metric. (6) Assign each item to the cluster of its associated point.

Due to the properties of the Laplacian tensor, in practice, the points in X can be easily clustered (Step 5) using simple conventional algorithms, like the K-Means or the hierarchical agglomerative algorithms. In the sequel we consider hierarchical agglomerative algorithms for this purpose based on Euclidean distance (Step 4).

Therefore, the proposed approach can better detect the clustering as it fully exploits all users-items-tags relationships. This is verified with the experimental results in the following section.

6. PERFORMANCE EVALUATION

6.1 Experimental setting

In our experiments we tested the proposed method, denoted as Tag-aware Spectral Clustering (TSC). For comparison purposes we tested two baseline Spectral Clustering methods, denoted as SC(U) and SC(T), that apply spectral clustering on a 2-dimensional item-user and item-tag matrix, respectively. In the former matrix an element is set to 1 when the corresponding item has been tagged at least once by the corresponding user (otherwise set to 0), whereas in the second matrix, when the corresponding item has been assigned at least once the corresponding tag (otherwise set to 0). All methods have been implemented in Matlab using the same components. Tensor factorisation was computed using the Tensor toolbox [2].

We used a real data set crawled from Last.fm (June 2008) by using Last.fm web services. The music items correspond to song titles. There are 64,025 triplets in the form user–tag–song. These triplets correspond 732 users, 2,527 tags and 991 songs.

Social-tagging data present problems like tag polysemy and sparsity. To address them, we applied the widely used technique of Latent Semantic Indexing (LSI) and reduced the number of dimensions in the modes of users and tags, by maintaining a percentage of them. This reduction was performed by modelling the original triples as a 3-mode tensor and applying Tucker decomposition [11]. The item mode is left unchanged, whereas the number of maintained users and tags after this process is expressed as a percentage (default value 30%) of the original number of users and tags (for simplicity we use the same percentage for both). Both SC(U) and SC(T) also utilise this technique by maintaining the same percentage for users or tags.

To form the k-NN similarity graphs and multidigraphs, we used the cosine distance, which is commonly applied for 0-1 sparse data like in our case. We tested several values of k and found that all examined methods are not sensitive in this parameter (default value $k = 10$). For the fifth step of the spectral clustering algorithm, we examined the Unweighted Pair Group Method with Arithmetic mean (UPGMA) hierarchical agglomerative clustering algorithm over the Euclidean distance (in the spectral feature space). Following the approach of conventional spectral clustering algorithms [5], we considered the number of clusters as a user-defined parameter. The quality of the final clustering result is measured with the popular Silhouette coefficient (the higher the better) that expresses both the coherency within clusters and the separation between clusters. For an item that is mapped to a vector x in the spectral feature space and is assigned to cluster C, its silhouette coefficient $s(x)$ is calculated as follows: a_x is the mean distance of x from all other vectors in C, whereas b_x is the minimum mean distance from vectors in all other clusters except C. Then, $s(x) = (b_x - a_x)/\max(a_x, b_x)$. The overall silhouette coefficient is the mean of all $s(x)$ for each x. [3]

6.2 Experimental results

We experimentally compare TSC against SC(U) and SC(T). The mean Silhouette coefficients for varying number of clusters is depicted in Figure 4. Due to its ability to consider 3-way relationships, TSC clearly outperforms the two baseline methods, which suppress the 3-way relationships

[2] http://csmr.ca.sandia.gov/∼tgkolda/TensorToolbox/

[3] For all compared method the silhouette coefficients are computed based on the Euclidean distance in the resulting feature space.

into 2-way, thus loosing information that is valuable for the clustering.

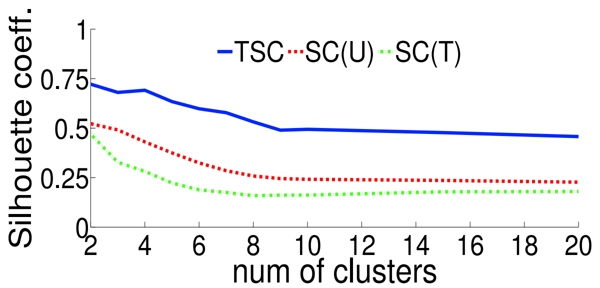

Figure 4. Results for varying number of clusters.

We also tested the sensitivity of the result against the percentage of maintained users/tags after the application of LSI (described in Section 6.1). Figure 5 depicts the resulting Silhouette coefficients for varying values of this percentage (the number of clusters is set to 10). When the percentage of maintained users/tags is severely low, the quality of TSC is reduced, as the resulting information is not adequate to capture the clustering structure. When the percentage is high, quality is again reduced, as the problems in the original data (polysemy, sparsity, noise) cannot be addressed. Therefore, in accordance to most applications of LSI, the best performance is attained with percentages that are in between the two extremes. In all cases, TSC compares favorably against TS(U) and TS(T).

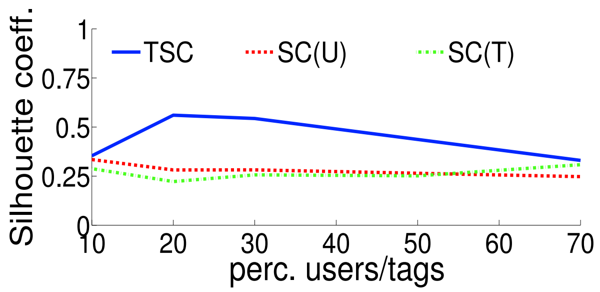

Figure 5. Results for varying perc. maintained users/tags.

7. CONCLUSIONS

We proposed a novel, tag-aware clustering algorithm for music data from social tagging systems. The advantage of the proposed algorithm over conventional clustering algorithms is that it preserves all 3 dimensions in the data and the 3-way relationships among them. The 3-way relationships of the form items–users–tags between these dimensions offers a clear advantage between just knowing that a tag has been applied to an item regardless by which users, and knowing the specific users that applied this tag to the item. To attain its advantages, the proposed algorithm uses tensors to store the underlying data model represented with multigraph structures, and extracts spectral features from them using tensor factorisation. Experimental results with real data showed that the proposed method yields clustering with better quality compared to conventional spectral clustering methods that suppress the dimensions and consider only 2-way relationships.

8. REFERENCES

[1] D. Byrd. Organization and searching of musical information, course syllabus, 2008.

[2] A. Morgan and M. Naaman. Why we tag: motivations for annotation in mobile and online media. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 971–980, 2007.

[3] Last.fm. Listen to free music with internet radio and the largest music catalogue online, 2009.

[4] MyStrands. Social recommendation and discovery, 2009.

[5] U. von Luxburg. A tutorial on spectral clustering. Technical report, (No. TR-149) Max Planck Institute for Biological Cybernetics, 2006.

[6] P. Lehwark, S. Risi, and A. Ultsch. *Data Analysis, Machine Learning and Applications*, chapter Visualization and Clustering of Tagged Music Data, pages 673–680. Springer Berlin Heidelberg, 2008.

[7] M. Levy and M. Sandler. Learning latent semantic models for music from social tags. *Journal of New Music Research*, 37(2):137–150, 2008.

[8] N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: A survey. *IEEE Signal Processing Magazine*, 23(2):133–141, 2006.

[9] P. Symeonidis, M. Ruxanda, A. Nanopoulos, and Y. Manolopoulos. Ternary semantic analysis of social tags for personalized music recommendation. In *Proc. International Conference on Music Information Retrieval (ISMIR08)*, 2008.

[10] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS'01)*, pages 849–856, 2001.

[11] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3), September 2009 (to appear).

[12] L. de Lathauwer, B. de Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal of Matrix Analysis and Applications*, 21(4):1253–1278, 2000.