CLAM!: Inferring Genres in the Discogs Collaboration Network

John Burke
jcburke@stanford.edu

Rasmus Rygaard
rygaard@stanford.edu

Zachary Yellin-Flaherty
zachyf@stanford.edu

December 10, 2014

Abstract

Labelling musical artists with genres is difficult and time consuming. Artists resist labels, experts have a hard time assigning genres, and automatic approaches to assigning labels mostly rely on identifying complex musical patterns mostly without much success. We propose an alternative mechanism for predicting genre labels using collaborations, which are much easier to record and quantify. We build a network of hundreds of thousands of artists and collaborations based on data from discogs.com and use scalable community detection algorithms to identify clusters. Because of the way we construct our network, these clusters map closely to genre labels and we show that our approach outperforms a randomized baseline by a wide margin.

1 Introduction

Musicians tend to resist genre labels. They argue that this imprecise categorization derives more from the innate human need to classify than from the actual music. In this paper we offer the musicians a chance to define their genres themselves. We infer genre groupings from a network of musical collaborations and collective releases. We analyze album release data provided by discogs.com and build a collaboration network on top of it. We then apply community detection techniques to infer genres from community structure based on the assumption that artists working in the same genre are more likely to collaborate with one another. We evaluate our inferred genre groupings by comparing with the user-supplied genre tags applied to the artists' releases on discogs.com.

We believe accurate collaboration-based genre labels will be useful for two reasons. First, as previously mentioned, genre classification is a hard task even for experts. Genres often blend together and distinctions become difficult to draw even for people with deep knowledge of the field. Tagging is also a time consuming task and accurately tagging large amounts of data is very challenging. For example, Discogs has clear guidelines for how to label artist collaborations, but there are no clear guidelines for how to attach genre labels. Collaborations, however, are easy to quantify and can often be automatically extracted from record release information. There is therefore little to no manual labor involved, especially if we can rely on a small set of labels to label each cluster we discover.

Second, a scalable approach to genre-labels would be valuable for music recommendation engines. For instance, online stores like iTunes or Amazon could identify fine-grained genres associated with users' purchasing habits and recommend artists that fall within the same genre given the collaboration patterns. While most recommendation systems have a bias towards popular artists, collaboration patterns are more or less independent of artist popularity and could therefore provide a much more even set of recommendations.

2 Prior Work

The general problem of community detection is tackled by Girvan and Newman in [1]. They introduce the property of modularity, a quantity that intuitively represents, for a given division into communities, how much more likely an edge is to be internal to a community as opposed to connecting two different communities. Their algorithm performs a bottom-up hierarchical clustering based on another property, edge betweenness (roughly, the number of shortest paths including a given edge), and then chooses the level of the hierarchy that maximizes modularity.

Clauset, Newman and Moore describe another algorithm based around modularity in [2]. Their method works by starting with each node in its own community and, at each step, combining the two that will most increase the overall modularity, therefore greedily optimizing that quantity. Unlike the original Girvan-Newman algorithm, the CNM method is designed to scale to large networks. But our graph is so large that CNM runs out of memory before completion. We looked to more scalable techniques.

In [3], Yang and Leskovec describe such an algorithm, based on a model in which nodes have weighted affiliations to communities (possibly to more than one of them), and the probability of two nodes being connected is a function of their shared affiliations. Their algorithm works by finding the most likely parameters for this model to generate the observed graph, computing the maximum likelihood estimate using row-by-row gradient ascent that, with a bit of memoization, only takes time linear in the degree of the node whose row we are working on. This algorithm has been our preferred technique so far because it is very fast and fits our memory constraints. Meanwhile we continue to explore other methods too.

Rovall and Bergstrom take a somewhat different approach to community detection in [4]. Using ideas from information theory, the authors use random walks and Huffman codes to identify communities in network as efficiently as possible. The algorithm first uses power method simulations of random walks to efficiently identify the important nodes in the network. After calculating the importance vector, node names can be calculated easily by assigning Huffman codes that reflect the relative importance of each node. These names, however, do not identify communities. The final algorithm therefore partitions the network into communities and searches for the partitioning that yields the shortest description of the random walks across the clusters. Rosvall and Bergstrom suggest that this method and similar flow-based algorithms are superior to modularity-based approaches in networks "where links represent patterns of movement among nodes."

3 Data Collection

Our dataset comes from discogs.com. It provides a list of albums, artists, and metadata, including performers, writers, producers, record labels, and numerous other attributes. Discogs provides free and unlimited access to their data through their API (`api.discogs.com`) and through periodic database dumps (`discogs.com/data/`). We used a snapshot of the database from October 1 2014. The dataset contains information on more than 5,000,000 releases, 3,500,000 artists, and 650,000 labels.

To create a network of collaborations from this data, we constructed a graph with the following edges:

1. Two artists who release a single or album together have an edge between them. For example, John Lennon and Yoko Ono would have an edge between them since they released "Happy Xmas (War Is Over)" (and many other songs) together.

2. Two artists who form a group together have an edge between them. For example,

John Lennon and Paul McCartney would have an edge between them since they were both members of The Beatles.

3. We add an edge between two artists if one is credited on a song by the other. For example, George Martin produced a large number of songs for the Beatles and would therefore have an edge to John Lennon.

Edges are undirected and unweighted, since all we wish to capture is whether or not two people have collaborated in some way; one could imagine extending this graph by weighting different kinds of collaboration differently or keeping track of repeated collaborations, but we have not explored this possibility yet. Newman et. al. suggest in [1] that adding weights tends not to affect the community structure significantly.

It is important to note that the the above criteria, if translated literally, produces a very large graph. For example, consider the reissue of The Beatles' 1966 album *Revolver* from 2012. In addition to the key artists we would expect to be credited (George Harrison, John Lennon, George Martin, Paul McCartney, and Richard Starkey), the release credits 6 designers, engineers, photographers, reissue producers, and remastering engineers. While these people were without a doubt important when making the reissue, they were most likely not critical for the original release. Perhaps more importantly, the non-musical collaborators tell us next to nothing about genres. For example, Geoff Emerick, principal engineer on the reissue, has engineered over 200 releases for artists spanning a wide range of genres.

4 Techniques Used

As outlined in our Prior Work section, we have mostly used BigCLAM, the method introduced in [3]. We experimented with other methods but without luck. In particular, the SNAP implementations of Infomap as described in [4] and Clauset-Girvin-Newman [2] proved too slow to run in practice. We therefore focused our work on more heavily optimized algorithms like a parallel OpenMP version of Infomap and BigCLAM.

After exploring more scalable options, we decided to take advantage of BigCLAM's ability to detect overlapping communitites. This requires using a slightly different approach from the other algorithms outlined previously since it detects overlapping communities by using maximum likelihood estimates rather than using modularity and matrix eigenvalues. The underlying assumption in BigCLAM is that each node has ties to several communities and that these ties can be represented probabilistically using a single value for each community per node. We can add edges between nodes if they belong to the same community. Since each node knows its probability of belonging to a certain community, two nodes have an edge between them if they both belong to at least one common community.

If the probability of node u belonging to community C is F_{uA}, BigClam models the probability of a link between them through C as $P_C(u,v) = 1 - \exp(-F_{uA} \cdot F_{vA})$. Given that u and v can belong to many communities, the probability of having an edge between them has to take into account all possible communities, which tells us that $P(u,v) = 1 - \prod_C (1 - P_C(u,v))$. Since the edges are generated independently, we can rewrite this as $P(u,v) = 1 - exp(-\sum_C F_u C \cdot F_v C) = 1 - exp(-F_u \cdot F_v)$ where F_u is the vector of membership strengths for u.

Given the above equation, BigCLAM estimates community memberships for node u by fixing membership probabilities for all other nodes and estimating F_u using a maximum likelihood estimate. This process can be further optimized through clever caching techniques to avoid repeatedly paying for calculating the objective function for non-neighbor nodes of u. This gives us a much faster algorithm whose

running time for each row is proportional only to the number of neighbors of a given node, not the total number of nodes.

5 Initial Findings

Our initial approach involved establishing to what extent communities exist in our graph. As we've mentioned before, community detection algorithms based on modularity maximization have been too slow for our dataset. As a result, we've limited the Clauset-Newman-Moore algorithm to run on small subsets of our data. The vast majority of edges in our graph come from the third type of collaboration as outlined above. Stripping out those edges give us a graph with hundreds of thousands of nodes and edges which we can analyze using CNM. For this particular dataset, the modularity achieved by CNM is 0.880, which suggests that even a stripped down version of the graph has high community structure. It appears that the resulting clusters have relatively high precision, since Mozart is in a cluster with chorales and other classical composers, but poor recall, since a lot of other artists share that cluster.

We focused our initial analysis on the properties of the graph that we could more easily analyze. Figure 1 shows the sizes of the connected components and the degree distributions for the network. Of particular interest is the fact that 94% of nodes fall in the largest strongly connected component, suggesting that collaboration across genres is fairly common. Depending on the extent of this, the cross-genre collaboration could be an issue for our analysis that depends on the assumption that collaborations mostly take place within genres.

The community structure in the graph is even more evident when looking at Figure 1. The average artist has 33 collaborations, and for nodes of that degree, clustering is around 0.70. In the graph as a whole, the clustering coefficient is 0.758 on average, which is fairly high. This again suggests a high level of collaboration between artists.

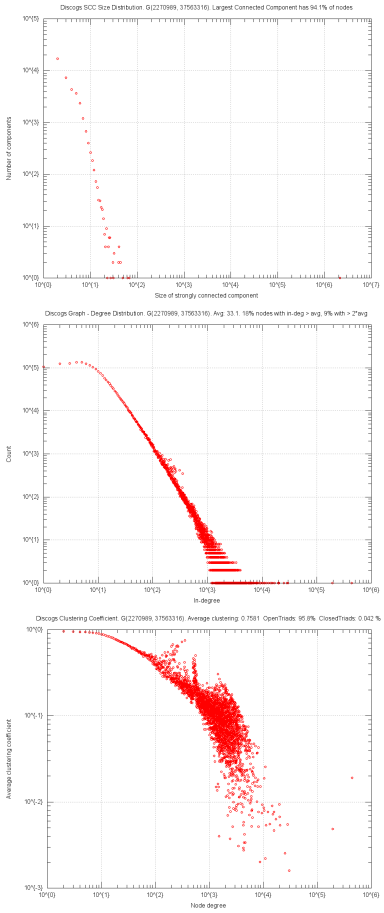

Figure 1: SCC, degree, clustering coefficient, and shortest path distributions for the Discogs network

6 Results

Our network proved difficult to analyze in the form described above since a naive parsing of the data yields a graph with millions of nodes and tens of millions of edges. We therefore tried two approaches to reducing the size of the network to better test our hypothesis.

6.1 Analyzing Recent Releases

We found that filtering releases by year was one effective way to lower the noise in our data, so

Communities	Collaboration	Baseline
1,000	77.3%	18.6%
10,000	85.3%	20.1%
25,000	87.5%	20.1%

Table 1: Genre Jaccard Similarity averaged across communities (2013 only)

Communities	Collaboration	Baseline
1,000	6.3%	41.2%
10,000	11.7%	59.9%
25,000	15.3%	61.6%

Table 2: Percentage of outliers for different numbers of communities (2013 only)

we combined this strategy with allowing only title artists, featured artists, writers, and producers to have relationships resulting from releases. The resulting graph of 2013-only data has 43,000 nodes and 77,000 edges. We ran BigCLAM with 1,000, 10,000, and 25,000 communities and found community membership distribution followed a power-law as seen in Figure 2.

In order to analyze how well clusters in the collaboration network reflected genres, we found the ground truth genre labels from Discogs, calculated various metrics (enumerated in the next paragraphs), and compared them against a baseline. The baseline we considered was a set of communities with identical distribution to BigCLAM output but with random nodes assigned to communities. We achieved this baseline by mapping each node to a random node and replacing all of its occurrences with the random node in the BigCLAM output. This keep all network properties including edge distribution, community distribution, and community memberships per node the same while ignoring the underlying real-world network.

We calculated average Jaccard similarity across all combination of artists in a community, and again averaged this community Jaccard similarity across all communities. The sets we considered were the Discogs' genre labels for each artist. The results are in Table 1. As the table confirms, artists sharing a community in the collaboration network are about four times as similar as our baseline, where similarity refers to the similarity of genres as defined by our ground truth, the Discogs genre labels.

Another meaningful metric is the percentage of outlier genres per community. We define an outlying genre to be a genre for which there is only one artist in the community belonging to that genre. The intuition for this measurement is that these genres are errors. If only one artist belongs to that genre, then that genre is noise because it is not representative of what defines the community. We calculate the percentage of outlying genres for each community and average it across all communities for our collaboration and baseline networks. The results are displayed in Table 2. About half of all genre labels in our baseline are outlier genres. Collaboration-based community genres are about 1/5th as likely to be outliers, indicating that the majority of genres appearing the collaboration network clusters have some relationship across artists in the community.

A final meaningful metric is the percentage of nodes that participated in the most popular genre of the community. This measurement measures how strongly a genre defines a community. As seen in Table 3, the collaboration network has an extremely high percentage of artists belonging to the most popular genre in the community. With the randomized baseline, only about half the artists can be expected to belong to the most popular genre in each community.

Notably, the most popular genres (Electronic, Rock, and Hip-hop) were all evenly represented as uniting communities. There was a drop-off to Pop, and another drop-off to less popular genres like Jazz and Classical. The differences between baseline and collaboration network results are significant, and our hypoth-

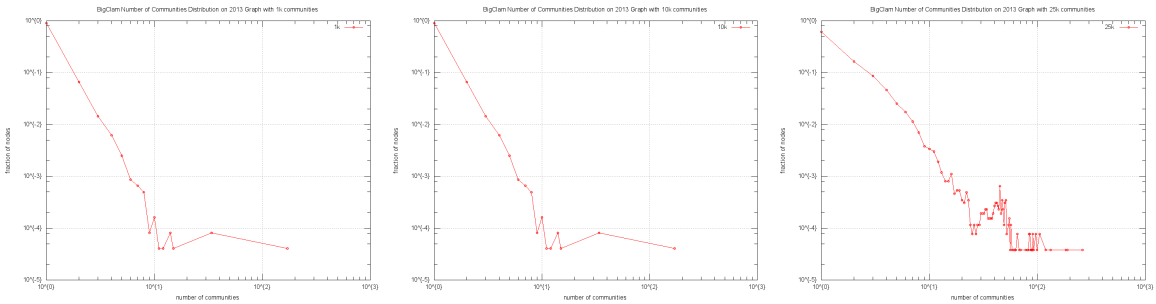

Figure 2: Distribution of community memberships for varying numbers of communities (2013 only)

Communities	Collaboration	Baseline
1,000	93.7%	47.7%
10,000	98.2%	57.7%
25,000	99.6%	57.6%

Table 3: Percentage of artists belonging to the most popular genre in each community (2013 only)

esis that collaboration metadata strongly indicates genre appears to be valid.

The metrics also illustrate a trade-off when we change the number of communities detected. The more communities we detect, the smaller on average the community and the more likely we only have one uniting community. As a result, participation in the most popular community grows as we increase the number communities. However, the trade-off is that outlier percentage also grow, likely because secondary genre members no longer have as much of an effect on community membership.

6.2 High Quality Data

Discogs data is user-generated and subsequently voted on by the existing userbase. The confidence score for each piece of data is exposed in the dataset. We therefore decided to only consider data had that the Discogs users considered complete or near-complete. Besides the reduced size, an advantage of this approach is that the confidence threshold eliminates very obscure artists who would otherwise be attached to the network through very few, low-quality votes. The resulting graph has 310,000 artists and 790,000 collaborations.

Using BigCLAM, we divided this graph into several sets of overlapping communities of different sizes: 5000 communities, 10000, 15000, 20000, 25000, 50000, and 100000. We then explored how similar artists in the same community are. We decided to use a variant of Jaccard similarity to assess this. Suppose $C = \{a_1, ..., a_n\}$ where a_i is the list of genre labels assigned to artist i in the Discogs data set. For a community C, we use the following score

$$sim(C) = avg\{J(a,b) \,|\, \{(a,b) \,|\, C \times C\,\}\}$$

where $J(a,b)$ is the Jaccard similarity between the two vectors. We initially considered calculating the average Jaccard similarity between each artist and the union of all genre labels in the community. *sim* as defined above, however, rewards clusters where many nodes have high similarity without dropping the score significantly if a few outliers exist.

Given this definition, we explored the quality of our cluster. Figure 3 shows plots of community size and average pairwise similarity for our graph partitioned into 25,000 and 100,000 overlapping communities. We used a randomized baseline that rearranged artists into different clusters while keeping the number of clusters, cluster sizes, and cluster membership distribution constant. While the difference is not as pronounced for small communities, larger communities have significantly

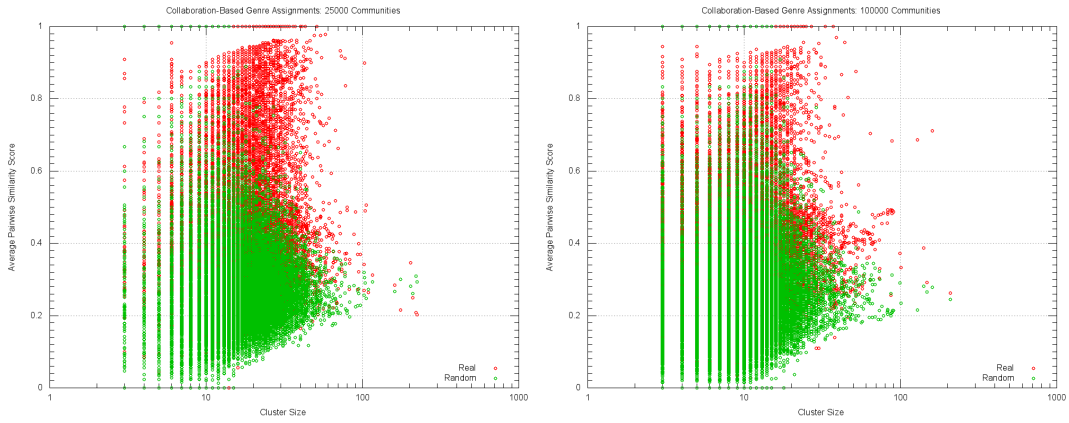

Figure 3: Community size and average pairwise similarity score for 25,000 and 100,000 comunities. Green points are randomized, red points are real.

higher similarity scores than the baseline. For communities with more than 10 artists, the similarity scores for random communities quickly drops below 0.5 while our collaboration-based communities have significantly higher scores overall. Even more importantly, much larger clusters are significantly better labelled using our collaboration-based network. No randomly generated community with more than 75 nodes has average similarity above 0.4, but majority of the collaboration-based communities have a higher similarity. This can also be seen in 4, which shows the sorted similarity scores for our results and a random baseline with both 5000 and 25000 communities.

Clearly in both cases the number of high-quality communities is much greater than in the equivalently sized random set of communities. More than half of communities having a pair-wise Jaccard similarity score over .4 and about 10% of communities scoring a perfect 1.0, meaning all members have the same genre tags. BigCLAM did not generate any singleton communities in either case, so these perfect scores really do indicate clusters of artists sharing exactly the same genre tags.

The number of clusters appears not to have an especially large effect on the overall average similarity score, though there is some difference between them. As show in Table 4, the average similarity increases up to 20,000 communities and then decreases from there. The differences between the different numbers of communities is more visible in Figure 5, which shows the sorted similarity scores for each different set of communities. Increasing the number of communities appears to also increase the number of good communities, unsurprisingly, but the averages are very close together, reflected in the similar shapes of the different lines. Picking a very small or extremely large number of communities, however, might skew the distribution more than we have seen in our experiments above.

Communities	Avg Similarity
5,000	0.448
10,000	0.461
15,000	0.456
20,000	0.480
25,000	0.471
50,000	0.460
100,000	0.457
5,000 (random)	0.261
25,000 (random)	0.244

Table 4: Average similarity for different numbers of clusters

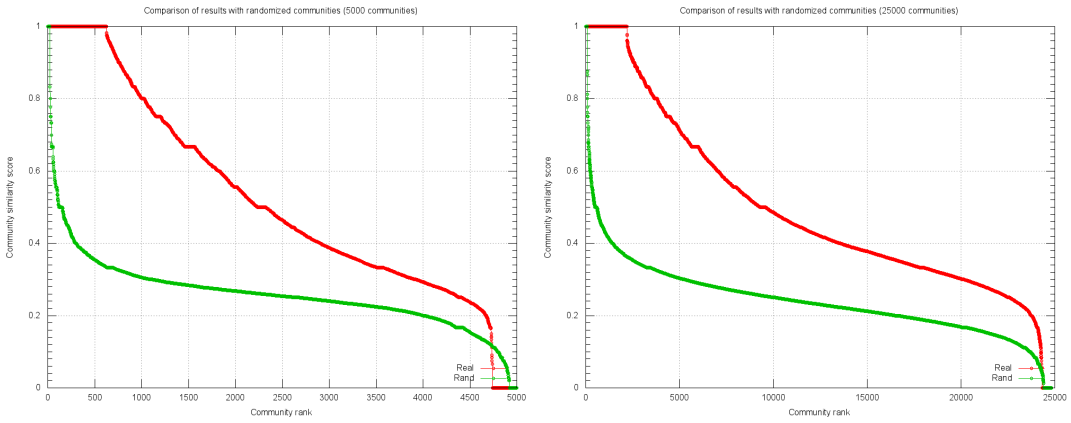

Figure 4: Comparison of similarity scores between real and random clusters. The first plot is for 5000 clusters, the second for 25000.

It is worth mentioning that we also let BigCLAM search for the optimal number of communities. That process, however, ended up taking prohibitively long even for a moderately sized graph as the algorithm started exploring configurations with more than 100,000 communities. Given the results we have outlined above, we do not have any reason to believe that a larger number of communities will increase the accuracy of our method unless the number is several factors larger than what we hav explored. In particular, Figure 3 shows us that increasing the number of communities might give us may smaller communities but not necessarily higher quality communities when seen in conjunction with Table 4.

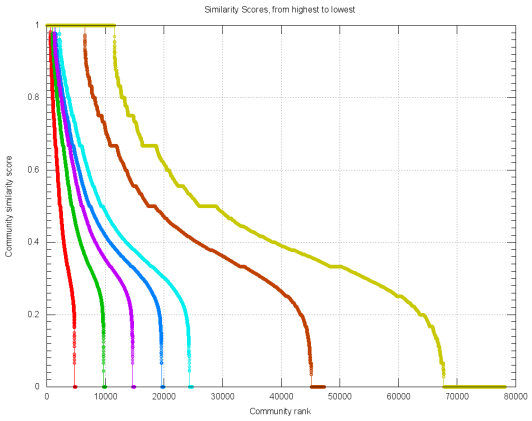

Figure 5: Similarity scores sorted from highest to lowest for each community configuration. The number of communities varies from 5,000 to 100,000 on the far right

7 Conclusion

Our results show that community detection methods can accurately find communities of the same genre in a network of music collaboration, finding clusters that were roughly twice as similar in genre (based on our pairwise Jaccard similarity metric) as randomized networks of the same size as structure. This seems to verify our hypothesis that collaborations mostly take place within genres, and given that the boundaries of genres are some-

what fluid and many artists span multiple genres, and that our "ground truth" is taken from user-submitted tags describing fairly broad categories like "Rock" or "Electronic", the clusters may actually be better than our results indicate.

These results could potentially be applied to suggesting genre tags for artists that have collaboration connections but have not been given a genre. They could also be used as part of a recommendation system, by recommending artists based on shared clusters with artists that we know a user likes. One could even envision avoiding imprecise genre labels altogether and describing an artist's "genre" by assigning representative artists to each community (based on centrality or popularity, or some other metric), and then describing an artist by listing the representatives of their top few most-affiliated communities.

References

[1] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[2] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[3] Jaewon Yang and Jure Leskovec. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 587–596. ACM, 2013.

[4] M Rosvall and C Bergstrom. Maps of information flow reveal community sctructure in complex networks. *arXiv preprint physics.soc-ph/0707.0609*, 2007.