

# DISENTANGLED MULTIDIMENSIONAL METRIC LEARNING FOR MUSIC SIMILARITY

Jongpil Lee<sup>1\*</sup>    Nicholas J. Bryan<sup>2</sup>    Justin Salamon<sup>2</sup>    Zeyu Jin<sup>2</sup>    Juhan Nam<sup>1</sup>

<sup>1</sup>Graduate School of Culture Technology, KAIST, Daejeon, South Korea

<sup>2</sup>Adobe Research, San Francisco, CA, USA

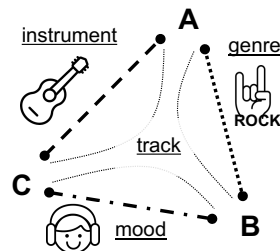
## ABSTRACT

Music similarity search is useful for a variety of creative tasks such as replacing one music recording with another recording with a similar “feel”, a common task in video editing. For this task, it is typically necessary to define a similarity metric to compare one recording to another. Music similarity, however, is hard to define and depends on multiple simultaneous notions of similarity (i.e. genre, mood, instrument, tempo). While prior work ignore this issue, we embrace this idea and introduce the concept of multidimensional similarity and unify both global and specialized similarity metrics into a single, semantically disentangled multidimensional similarity metric. To do so, we adapt a variant of deep metric learning called conditional similarity networks to the audio domain and extend it using track-based information to control the specificity of our model. We evaluate our method and show that our single, multidimensional model outperforms both specialized similarity spaces and alternative baselines. We also run a user-study and show that our approach is favored by human annotators as well.

**Index Terms**— multidimensional music similarity, metric learning, disentangled representation, query-by-example.

## 1. INTRODUCTION

Traditional music *search* methods such as those available on streaming services and online music repositories use text-based metadata (e.g. song, artist, album, and/or semantic tags) for music retrieval. However, there are scenarios where music metadata is either unavailable or insufficient: a concrete example is what we shall refer to as the “music replacement” problem, where a user wishes to replace one music recording with another recording that has a similar “feel”, a common use case e.g. in video editing. Describing the desired musical traits may be extremely hard to do with text, but the user has an example of what they are searching for, and so query-by-example, and more specifically content-based music similarity and retrieval, is an attractive solution. While content-based music similarity has been studied extensively [1], it has found limited application in music recommendation platforms, which rely most heavily on interaction and metadata based collaborative filtering [2].



**Fig. 1.** An illustration of multiple dimensions of music similarity. Letters (A, B, C) denote different music recordings, while lines denote different dimensions of similarity.

Such techniques are not applicable, however to the music replacement scenario, where there may be little-to-no interaction data, and a user’s past music replacement selections can have little correlation with future replacement needs. From a retrieval specificity perspective, music replacement is less specific than music identification (fingerprinting), but more specific than tag-based retrieval (e.g. genre) or than finding similar-sounding music for listening purposes [1, 3], since the pragmatic goal of music replacement is to find songs which sound as close as possible to a query without being identical.

Content-based music similarity typically involves extracting a feature representation from audio recordings and computing the similarity (or distance) between them using a metric or score function. Previous approaches include vector quantization [4], linear metric learning [5, 6, 7], and, more recently, deep metric learning [8, 9, 10] using human similarity labels [11], artist labels [12], track labels [13], or tags in the context of zero-shot learning [14]. A common limitation of these approaches is that similarity is modeled as uni-dimensional, i.e. songs are modelled as similar or dissimilar along a single global axis. In actuality, music is a multidimensional phenomenon, and consequently there are various *different* dimensions along which songs can be compared (e.g. timbre, rhythm, genre, mood, etc.), and songs can be simultaneously similar along some dimensions, while different along others, as illustrated in Figure 1. It is also hard to determine precisely which dimensions people take into account when rating songs for similarity, or how they weight the importance of these dimensions. For this reason, from an application standpoint it can be beneficial to allow the user to specify which musical dimensions they care about when searching-by-example and how to weight their importance.

\* This work was performed during an internship at Adobe Research.

In this paper, we propose a deep *disentangled* metric learning method for learning a multidimensional music similarity space (embedding). We adapt Conditional Similarity Networks [15], previously only applied to images, to the audio domain, and employ a combination of user-generated tags and algorithmic estimates (i.e. tempo) to train a disentangled embedding space composed of sub-spaces corresponding to similarity along different musical dimensions: genre, mood, instrumentation and tempo. Further, we propose a track-regularization technique to increase overall perceptual similarity across all dimensions as judged by humans. We evaluate our approach against several baselines, showing our proposed approach outperforms them both in terms of global similarity and similarity along specific dimensions. To validate our quantitative results, we run a user-study and show that our proposed approach is favored by human annotators as well.

## 2. LEARNING MODEL

### 2.1. Metric learning with triplet loss

We use deep metric learning with a triplet loss as the basis for our learning model [8, 9]. On a high level, our model is presented with a triplet of samples, where one is considered the “anchor” and the other two consist of a “positive” and a “negative”, and the model is trained to map the samples into an embedding space where the “positive” is closer to the “anchor” than the “negative”, as illustrated in Figure 2(A).

Formally, we define training triplets as a set  $T = \{t^i\}_{i=1}^N$ , where each triplet  $t^i = \{x_a^i, x_p^i, x_n^i | s(x_a, x_p) > s(x_a, x_n)\}$ ,  $x_a$  is the anchor sample,  $x_p$  is the positive sample,  $x_n$  is the negative sample, and  $s$  is the musical dimension along which similarity is measured. Then, we define the triplet loss as:

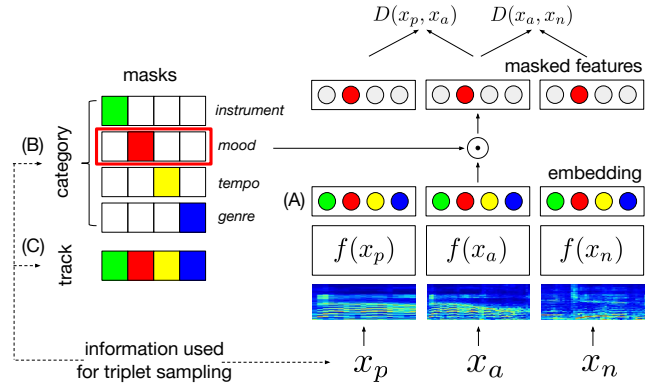
$$L(t) = \max\{0, D(x_a, x_p) - D(x_a, x_n) + \Delta\}, \quad (1)$$

where  $D(x_i, x_j) = \|f(x_i) - f(x_j)\|_2$  is the euclidean distance between two audio embeddings,  $\Delta$  is a margin value to prevent trivial solutions, and  $f(\cdot)$  is a nonlinear embedding function or deep neural network that maps the audio input to the embedding space. For a given set  $T$  and embedding function  $f(\cdot)$ , we use stochastic gradient descent to update the network weights and minimize the loss.

### 2.2. Disentangling the embedding features

To jointly model multiple semantic dimensions of similarity within a single network, we adapt the work of Veit et al. [15], which proposed the use of Conditional Similarity Networks (CSN) [15] for attribute-based image retrieval. The method introduces masking functions  $m_s \in \mathbb{R}^d$ , which are applied to the embedding space of size  $d$ . Each mask corresponds to a certain similarity dimension  $s$  (denoted “condition” in [15]), e.g. mood or tempo, and is used to activate or block disjoint regions of the embedding space, as illustrated in Figure 2(B).

Given a specific similarity dimension  $s$ , training triplets are defined as  $T_s = \{t_s^i\}_{i=1}^N$ , with each triplet given by:



**Fig. 2.** Our proposed approach. (A) Standard triplet-based deep metric model, (B) conditional similarity masking, and (C) track regularization.

$$t_s^i = (x_a^i, x_p^i, x_n^i; s), \quad (2)$$

and the training set combining triplets sampled from all similarity dimensions is defined as  $T_S = \{T_s\}_{s=1}^S$ . Consequently, we update the distance function to:

$$D(x_i, x_j; s) = \|f(x_i) \circ m_s - f(x_j) \circ m_s\|_2, \quad (3)$$

such that the mask  $m_s$  only passes through the subspace of embedding features corresponding to similarity dimension  $s$  during training and  $\circ$  denotes Hadamard product. Accordingly, the loss is updated to:

$$L(t_s) = \max\{0, D(x_a, x_p; m_s) - D(x_a, x_n; m_s) + \Delta\}. \quad (4)$$

### 2.3. Track regularization

As noted earlier, music replacement requires retrieved songs to sound as close as possible to the query example. To this end, we propose to complement the aforementioned multi-dimensional metric learning approach with a regularization technique we refer to as “track regularization”. The approach involves sampling an additional set of triplets solely based on the track (song) information: the anchor and positive are both sampled from the same song, while the negative is sampled from a different song. While this sampling was used previously to learn high-specificity music similarity directly [13], here we use it as a “similarity regularization” technique to enforce a certain degree of consistency across the entire (multi-dimensional) embedding space. With this regularization, our final loss is given by:

$$L(t_c, t_t) = L(t_c) + \lambda L(t_t), \quad (5)$$

where  $t_c$  are all triplets sampled from the various music similarity dimensions corresponding to disjoint sub-embedding spaces,  $t_t$  are triplets sampled using track information, and  $\lambda$  allows us to control the trade-off between semantic similarity (low-specificity) and overall track-based similarity (high specificity). Importantly, for track-based triplets, we use a mask with a value of one for all feature dimensions, meaning the regularization is applied to the complete embedding space to capture track similarity across all musical dimensions. Alternatively, this can be thought of as not applying any masking on the embedding space.

### 3. EXPERIMENTAL DESIGN

#### 3.1. Dataset and input features

For our experiments, we use the Million Song Dataset (MSD) [16]. Based on preliminary user studies on music replacement, we identify four relevant musical dimensions to consider: *genre*, *mood*, *instrumentation*, and *tempo*. To determine whether two songs are similar along these dimensions, we use Last.FM tag annotations associated with MSD tracks which have been previously grouped into different categories [17], resulting in 28 genre tags, 12 mood tags, and 5 instrument tags. Since the annotations lack tempo tags, we extract an algorithmic tempo estimate per track using the Madmom Python library [18, 19]. Two tracks are considered similar along a certain musical dimension (genre, mood, instruments) if they share at least one tag in that category, or are within 5 BPM of each other in the case of tempo. For track-based triplets, we ensure there is no more than 50% overlap between the anchor and positive samples. We split the data following [20], giving 201680, 11774, and 28435 samples for the train, validation, and test sets, respectively.

For training, we use 3-second excerpts represented as a log-scaled mel-spectrogram  $S$ , extracted with librosa [21]. We use a window size of 23 ms with 50% overlap and compute 128 mel-bands per frame with the following log-compression:  $\log_{10}(1 + 10 * S)$ , resulting in input dimensions of  $129 \times 128$  as in [12]. The representation is z-score standardized using fixed mean and standard deviation values of 0.2 and 0.25, respectively.

#### 3.2. Model architecture and training parameters

For choosing the triplet network architecture, we ran preliminary experiments with several state-of-the-art convolutional building blocks [22], including a basic conv-batchnorm-maxpool block, ResNet [23], Squeeze-and-Excitation [24], and Inception [25]. Having identified the Inception block as the best option, we use the following model architecture: we start with 64 convolutional filters with a  $5 \times 5$  kernel followed by  $2 \times 2$  strided max-pooling, followed by six Inception blocks each comprising a “naïve” inception module with stride 2 followed by another inception module with a final output dimensionality of 256 [25]. We use ReLU nonlinearities for all layers, and apply  $L2$  normalization to the embedding features prior to computing the distance [10].

Since our total embedding size is 256 and we consider four music similarity dimensions (genre, mood, instruments, tempo), each with a disjoint subspace of size 64. We also experimented with a trainable masking layer [15] (as opposed to fixed disjoint masks), but found it did not lead to any significant improvement. Moreover, using fixed masks has the added benefit of allowing us to weight each musical dimension independently post-hoc which, as noted earlier, is a desirable user interaction paradigm. We use the Adam optimizer [26] for training. We initialize the learning rate to 0.01

and reduce it by a factor of 5 when the validation loss does not decrease for 4 epochs, up to 5 times, after which we apply early stopping. The margin for the triplet loss is set to 0.1. And, after empirically hearing the properties of similarity space,  $\lambda$  was set to 0.5 when track regularization is applied.

#### 3.3. Evaluation metrics and user-study

For evaluation we use a set of held-out triplets sampled from the test set. We sample 40,000 triplets per music dimension (genre, mood, instruments, tempo) as well as 40,000 triplets based on track information. To simulate our application scenario, we use triplets of full songs for evaluation, the only exception being track-based triplets, where we stick to 3 second excerpts since the anchor and positive are sampled from the same song and should not overlap by more than 50%. The embedding for a full song is obtained by computing embedding frames from 3-second non-overlapping windows and averaging them over the time dimension. Given a test triplet, a model is evaluated by testing whether the embedding distance between the anchor and positive samples is smaller than the distance between the anchor and negative (score of 1), or greater (score of 0). The scores for all triplets are averaged to obtain a final score between 0 (worst) and 1 (best).

To determine whether human subjects concur with the above quantitative evaluation, we also randomly sampled 4,000 triplets from the test set and asked people to annotate which track sounded more similar to the anchor (positive or negative) without showing which was which. Each triplet was annotated by 5-12 people, resulting in 39,440 human annotations. We then calculated the annotator agreement per triplet, defined as the ratio between the majority vote and total number of annotations, and filtered out triplets where the agreement was below 0.9, resulting in 879 high-agreement human-annotated triplets. Since similarity judgements have a high degree of subjectivity, in this way, we can limit the scope of our human evaluation to triplets where there is broad annotator agreement. Models are evaluated against these triplets as described earlier, obtaining a score between 0–1 in terms of consistency with user ratings. For reproducibility, we share our dataset of user similarity ratings, *dim-sim*, online, along with audio similarity examples for the proposed approach<sup>1</sup>.

#### 3.4. Baseline method

As a strong baseline, we implement a vector quantization method that has been used for both similarity-based music retrieval and auto-tagging [5, 27]. We compute 13 MFCC coefficients and their first and second derivatives per frame for each track, randomly select 2,500,000 frames from all tracks and cluster them using K-means with  $K = 1024$  to produce a dictionary [5]. Given the dictionary, a track embedding is obtained by assigning each MFCC frame to its closest cluster and computing a normalized histogram of cluster

<sup>1</sup><https://jongpillee.github.io/multi-dim-music-sim/>

Used space	Embedding Features	Genre	Mood	Instruments	Tempo	Overall
All-dimensions	MFCC-VQ	0.563	0.481	0.495	0.516	0.514
	Track	0.611	0.595	0.531	0.534	0.568
	Category	0.647	0.633	0.562	0.875	0.679
	Category + track regularization	0.647	0.627	0.561	0.891	0.681
	Category + disentanglement	0.708	0.717	0.657	0.783	0.716
	Category + disentanglement + track regularization	0.693	0.704	0.626	0.836	0.715
Sub-dimensions	Set of specialized networks	0.708	0.619	0.603	0.942	0.718
	Category + disentanglement	0.785	0.790	0.798	0.955	0.832
	Category + disentanglement + track regularization	0.765	0.743	0.700	0.953	0.790

**Table 1.** Prediction accuracy of category-based (genre, mood, instruments, tempo) triplets.

Embedding Features	Track	User
MFCC-VQ	0.833	0.654
Track	0.950	0.763
Category	0.975	0.766
Category + track regularization	0.980	0.740
Category + disentanglement	0.985	0.763
Category + disentanglement + track regularization	0.988	0.792

**Table 2.** Results on track-based and user-based triplets.

assignments. The distance between any two tracks is then given by the Euclidean distance between their normalized histograms [5].

#### 4. RESULTS

In Table 1, we present the numerical results obtained for each of the four held-out triplet sets corresponding to a music similarity dimension, as well as aggregated scores over all four triplet sets (“Overall”). The “Used space” column indicates which subset of the embedding space was used to compute the distance between pairs of tracks, where “all dimensions” means all embedding features were used ( $f(x)$ ), whereas “sub-dimensions” means only the subspace corresponding to the musical dimension ( $f(x) \circ m_s$ ) from which the test triplets were sampled was used. We compare six models plus the baseline, specified in the “embedding features” column. The “Track” model was trained on triplets sampled based on track-information only, the “Category” model was trained on triplets sampled from the four music similarity dimensions (categories) of genre, mood, instruments and tempo, including both with and without disentanglement (subspace masking) and track regularization. For disentangled models, we include an additional baseline, “Set of specialized networks”, which is comprised of four separate triplet-loss networks, each trained exclusively on triplets sampled from one of the four musical dimensions.

We see that all deep metric learning models outperform the MFCC-VQ baseline. More importantly, disentangling the embedding improves performance in almost all cases, with our disentangled model trained on all triplets jointly (Category + disentanglement) even outperforming the specialized networks trained separately on each dimension.

As one might expect, track regularization decreases numerical performance on each of the four triplet test sets, as it enforces all embedding subspaces to respect a global notion of track similarity. The key question is how does it affect

model performance when compared against the human ratings obtained from our user study, presented in Table 2. As a sanity check, we start by evaluating our models against the track-based triplet test-set, presented in the “Track” column. We see that, as expected, track-regularization increases performance on this high-specificity set. Somewhat surprisingly, training on category-sampled triplets outperforms training on track-sampled triplets, with disentanglement increasing performance further. Next, we turn to the results obtained from the user study, presented in the “User” column. We see that our proposed approach outperforms the baseline, and, as per our initial hypothesis, track regularization increases the overall user agreement with our model’s similarity ratings when training on category triplets with disentanglement.

#### 5. CONCLUSION

In this paper, we introduce a novel approach for deep metric learning of a disentangled, *multidimensional*, music similarity space. We use Conditional Similarity Networks trained on a combination of user tags and algorithmic estimates, and introduce track regularization to control for retrieval specificity. Through a series of experiments, including both a quantitative evaluation and a user study, we demonstrate that our proposed approach outperforms several baselines, with per-dimension similarity performance increasing due to the disentangling of the embedding space, and agreement with human annotations increasing as a result of track regularization. Our solution is particularly relevant to the music replacement problem, and opens the door to novel interaction paradigms which permit the user to select which music dimensions they care about for retrieval, how to weight their relative importance, and how to balance subspace similarity versus high-specificity overall similarity. This approach can further be extended to general audio similarity such as voice similarity based on their speaker’s condition, phonation, or prosody. In the future, we plan to conduct further user studies to determine human agreement when considering each musical dimension in isolation, and evaluate the performance of our model against these ratings. We also plan to explore and evaluate our proposed approach for multi-query retrieval (query-by-multiple-examples) and mix-and-match scenarios where the user is interested in finding songs whose characteristics match the subspaces of different songs (e.g. the genre of example A with the tempo of example B).

## 6. REFERENCES

- [1] M.A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, “Content-based music information retrieval: Current directions and future challenges,” *Proc. of the IEEE*, vol. 96, no. 4, pp. 668–696, 2008.
- [2] O. Celma, “Music recommendation,” in *Music recommendation and discovery*, pp. 43–85. Springer, 2010.
- [3] P. Grosche, M. Müller, and J. Serrà, “Audio content-based music retrieval,” in *Dagstuhl Follow-Ups*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2012, vol. 3.
- [4] B. Logan and A. Salomon, “A music similarity function based on signal analysis,” in *ICME*, 2001, pp. 22–25.
- [5] B. McFee, L. Barrington, and G. Lanckriet, “Learning content similarity for music recommendation,” *IEEE transactions on audio, speech, and language processing*.
- [6] M. Slaney, K. Weinberger, and W. White, “Learning a metric for music similarity,” in *ISMIR*, 2008.
- [7] D. Wolff, S. Stober, A. Nürnberger, and T. Weyde, “A systematic comparison of music similarity adaptation approaches,” in *ISMIR*. FEUP Edições, 2012, pp. 103–108.
- [8] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, “Learning fine-grained image similarity with deep ranking,” in *CVPR*, 2014, pp. 1386–1393.
- [9] E. Hoffer and N. Ailon, “Deep metric learning using triplet network,” in *Int. Workshop on Similarity-Based Pattern Rec.* Springer, 2015, pp. 84–92.
- [10] A. Jansen, M. Plakal, R. Pandya, D.P.W. Ellis, S. Hershey, J. Liu, R.C. Moore, and R.A. Saurous, “Unsupervised learning of semantic audio representations,” in *ICASSP*. IEEE, 2018, pp. 126–130.
- [11] R. Lu, K. Wu, Z. Duan, and C. Zhang, “Deep ranking: Triplet matchnet for music metric learning,” in *ICASSP*. IEEE, 2017, pp. 121–125.
- [12] J. Park, J. Lee, J. Park, J.-W. Ha, and J. Nam, “Representation learning of music using artist labels,” in *ISMIR*, 2018, pp. 717–724.
- [13] J. Lee, J. Park, and J. Nam, “Representation learning of music using artist, album, and track information,” in *Machine Learning for Music Discovery Workshop, ICML*, 2019.
- [14] J. Choi, J. Lee, J. Park, and J. Nam, “Zero-shot learning for audio-based music classification and tagging,” in *ISMIR*, 2019.
- [15] A. Veit, S. Belongie, and T. Karaletsos, “Conditional similarity networks,” in *CVPR*, 2017, pp. 830–838.
- [16] T. Bertin-Mahieux, D.P.W. Ellis, B. Whitman, and P. Lamere, “The million song dataset,” in *ISMIR*, 2011.
- [17] K. Choi, G. Fazekas, M. Sandler, and K. Cho, “Convolutional recurrent neural networks for music classification,” in *ICASSP*. IEEE, 2017, pp. 2392–2396.
- [18] S. Böck, F. Krebs, and G. Widmer, “Accurate tempo estimation based on recurrent neural networks and resonating comb filters,” in *ISMIR*, 2015, pp. 625–631.
- [19] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, “Madmom: A new python audio and music signal processing library,” in *24th ACM Int. Conf. on Multimedia*. ACM, 2016, pp. 1174–1178.
- [20] J. Lee and J. Nam, “Multi-level and multi-scale feature aggregation using pretrained convolutional neural networks for music auto-tagging,” *IEEE SPL*, vol. 24, no. 8, pp. 1208–1212, 2017.
- [21] B. McFee, C. Raffel, D. Liang, D.P.W. Ellis, M. McVicar, E. Battenberg, and O. Nieto, “librosa: Audio and music signal analysis in python,” in *14th Python in Science Conf.*, 2015, vol. 8.
- [22] T. Kim, J. Lee, and J. Nam, “Comparison and analysis of samplecnn architectures for audio classification,” *IEEE JSTSP*, vol. 13, no. 2, pp. 285–297, 2019.
- [23] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, 2016, pp. 770–778.
- [24] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *CVPR*, 2018, pp. 7132–7141.
- [25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *CVPR*, 2015, pp. 1–9.
- [26] D. P Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2015.
- [27] D. Liang, J.W. Paisley, and D.P.W. Ellis, “Codebook-based scalable music tagging with poisson matrix factorization,” in *ISMIR*, 2014, pp. 167–172.