IDENTIFYING RAGA SIMILARITY THROUGH EMBEDDINGS LEARNED FROM COMPOSITIONS' NOTATION

Joe Cheri Ross[1] **Abhijit Mishra**[3] **Kaustuv Kanti Ganguli**[2]
Pushpak Bhattacharyya[1] **Preeti Rao**[2]
[1] Dept. of Computer Science & Engineering, [2] Dept. of Electrical Engineering
Indian Institute of Technology Bombay, India
[3] IBM Research India
`joe@cse.iitb.ac.in`

ABSTRACT

Identifying similarities between ragas in Hindustani music impacts tasks like music recommendation, music information retrieval and automatic analysis of large-scale musical content. Quantifying raga similarity becomes extremely challenging as it demands assimilation of both *intrinsic* (*viz.*, notes, tempo) and *extrinsic* (*viz.* raga singing-time, emotions conveyed) properties of ragas. This paper introduces novel frameworks for quantifying similarities between ragas based on their melodic attributes alone, available in the form of *bandish* (composition) notation. Based on the hypothesis that *notes in a particular raga are characterized by the company they keep*, we design and train several deep recursive neural network variants with Long Short-term Memory (LSTM) units to learn *distributed representations* of notes in ragas from *bandish* notations. We refer to these distributed representations as *note-embeddings*. Note-embeddings, as we observe, capture a raga's identity, and thus the similarity between note-embeddings signifies the similarity between the ragas. Evaluations with perplexity measure and clustering based method show the performance improvement in identifying similarities using note-embeddings over n-gram and uni-directional LSTM baselines. While our metric may not capture similarity between ragas in their entirety, it could be quite useful in various computational music settings that heavily rely on melodic information.

1. INTRODUCTION

Hindustani music is one of the Indian classical music traditions developed in northern part of India getting influences from the music of Persia and Arabia [17]. The south Indian music tradition is referred to as Carnatic music [30]. The compositions and their performances in both these classical traditions are strictly based on the grammar prescribed by the raga framework. A raga is a melodic mode or tonal matrix providing the grammar for the notes and melodic phrases, but not limiting the improvisatory possibilities in a performance [25].

Raga being one of the most prominent categorization aspect of Hindustani music, identifying similarities between them is of prime importance to many Hindustani music specific tasks like music information retrieval, music recommendation, automatic analysis of large-scale musical content *etc*. Generally similarity between ragas is inferred through attributes associated with the ragas. For instance, in Hindustani music, classification of ragas based on the tonal material involved is termed as *thaat*. There are 10 *thaats* in Hindustani music [8]. *prahar*, *jati*, *vadi*, *samvadi etc.* are the other important attributes. Most of the accepted similarities between ragas encompass the similarities in many of these attributes. But these similarities cannot always be derived exclusively from these attributes. Melodic similarity is a strong substitute and close to perceived similarity. The melodic similarity between Hindustani ragas is not largely available in documented form. This necessitates systems for raga similarity measurement to be devised, even though the number of ragas in the Hindustani classical framework is fixed.

A composed musical piece termed as *bandish* is written to perform in a particular raga, giving ample freedom to the performer to improvise upon. As the literal meaning suggests, *bandish* is tied to its raga, *tala* (rhythm) and lyrics. *bandish* is taken as the basic framework for a performance which gets enriched with improvisation while the performer renders it. Realization of a *bandish* in a performance brings out all the colors and characteristics of a raga. Given this fact, audio performances of the *bandishes* can be deemed to be excellent sources for analyzing raga similarities from a computational perspective. However, methods for automatic transcription of notations from audio performances have been elusive; this restricts the possibilities of exploiting audio-resources. Our work on raga similarity identification, thus, relies on notations having abstract representation of a performance covering most dimensions of the composition's raga. We use *bandish* notations dataset available from `swarganga.org` [16].

Our proposed approach, based on deep recursive neural network with bi-directional LSTM as recurrent

Attribution: Joe Cheri Ross, Abhijit Mishra, Kaustuv Kanti Ganguli, Pushpak Bhattacharyya, Preeti Rao. "Identifying Raga Similarity Through embeddings learned from Compositions' notation", 18th International Society for Music Information Retrieval Conference, Suzhou, China, 2017.

units, learns note-embeddings for each raga from the *bandish* notations available for that raga. We partition our data by raga and train the model independently for each raga. It produces as many note-embeddings, as many different ragas we have represented in the dataset. The *cosine similarity* between the note-embeddings serves for analyzing the similarity between the ragas. Our evaluations with perplexity measure and clustering based methods show the performance improvement in identifying similarities using note-embeddings using our approach over (a) a baseline that uses n-gram overlaps of notes in *bandish* for raga similarity computation (b) a baseline that uses pitch class distribution (PCD) and (c) our approach with uni-directional LSTM. We believe, our approach can be seamlessly adopted to the Carnatic music style as it follows most of the principles as Hindustani music.

2. RELATED WORK

To the best of our knowledge no such attempts to identify raga similarity have been made so far. The work closest to ours is by Bhattacharjee and Srinivasan [5] who discuss raga identification of Hindustani classical audio performances through a transition probability based approach. Here they also discuss about validating the raga identification method through identifying known raga relationship between 10 ragas considered for this work. A good number of research works have been carried out pertaining to raga identification in Hindustani music using note intonation [3], chromagram patterns [11], note histogram [12]. Pandey *et al.* [22] proposed an HMM based approach on automatically transcribed notation data from audio. There has been quite a few raga recognition attempts in Carnatic music also [28, 4, 27, 24].

3. RAGA SIMILARITY BASED ON NOTATION: MOTIVATION AND CENTRAL IDEA

While the general notion of raga similarity is based on various dimensions of ragas like *thaat*, *prahar*, *jati*, *vadi*, *samvadi etc.*, the similarities perceived by humans (musicians and expert listeners) is predominantly substantiated upon the melodic structure. A raga-similarity method solely based on notational (melodic) information can be quite relevant to computational music tasks involving Indian classical music.

Theoretically, the identity of a raga lies in how certain notes and note sequences (called phrases) are used in its compositions. We hypothesize that capturing the semantic association between different notes appearing in the composition can possibly reveal the identity of a raga. Moreover, it can also provide insights into how similar or dissimilar two ragas can be, based on how similar / dissimilar the semantic associations of notes in the compositions are. We believe , notes for a specific raga can be represented in *distributed forms* (such as vectors), reflecting their semantic association with other notes in the same raga (analogous to words having distributed representations in the domain of computational linguistics [18]). These representations could account for how notes are preceded and succeeded by other notes in compositions.

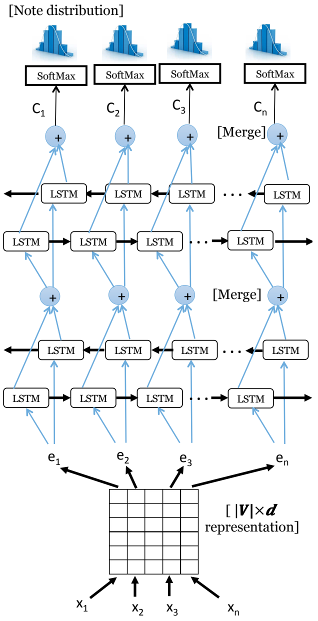


Figure 1. Bi-directional LSTM architecture for learning note-embeddings

Formally, in a composition, a note $x \in V$ (where V represents a vocabulary all notes in three octaves) can be represented as a d dimensional vector that captures semantic-information specific to the raga that the compositions belong to. Such distributed note-representations, referred to as *note-embeddings* ($|V| \times d$ matrix) can be expected to capture more information than other forms of sparse representations (like presenting notes with unique integers). We propose a bi-directional LSTM [14] based architecture that is motivated by the the work of Huang and Wu [15] to learn note-embeddings characterizing a particular style of music. We learn note-embeddings for each raga separately from the compositions available for the raga.

How can note-embeddings help capture similarities between ragas? We hypothesize that embeddings learned for a given note for similar ragas will have more similarity. For example, the representation for note *Ma-elevated* (equivalent note *F#* in C-scale) in raga *Yaman* can be expected to be very similar to that of *Yaman Kalyan* as both of these ragas share very similar melodic characteristics.

4. NEURAL NETWORK ARCHITECTURE FOR LEARNING *NOTE-EMBEDDINGS*

We design a deep recurrent neural network (RNN), with bi-directional LSTMs as recurrent units, that learns to predict the forth-coming notes that are highly likely to appear in a *bandish* composition, given input sequences of notes. This is analogous to neural language models built for speech and text synthesis [19]. While our network tries to achieve this objective, it learns distributed note representations by regularly updating the note-embedding matrix. The choice of this architecture is due to the facts that

(a) for sequence learning problems like ours, RNNs with LSTM blocks have proven useful [29, 13], and (b) in Hindustani music a note rendered at a moment has dependence on patterns preceding and succeeding it, motivating us to use bi-directional LSTM.

The model architecture is shown in Figure 1. Supposing that a sequence in a composition has n notes (n to be kept constant by padding wherever necessary), denoted as $x_1, x_2, x_3, ..., x_n$, where $\forall i \in n, x_i \in V$. The note x_i can be represented in *one-hot* format, with the j^{th} component of a $|V|$ dimensional zero-vector set to 1, if x_i is the j^{th} element of vocabulary V. Each note is input to a *note-embedding* layer W of dimension $|V| \times d$ where d is the note-embedding dimension. The output of this layer is a sequence of embeddings e_i of dimension d, obtained by performing a matrix multiplication between x_i with W. The embedding sequences $e_1, e_2, e_3, ..., e_n$ are input to two layers of bi-directional LSTMs.

For each time-step ($i \in n$), the context-representations learned by the outer-bidirectional LSTM layer (C_i) is passed through a `softmax` layer that computes the conditional probability distribution of all possible notes given the context representations given by LSTM layers.

For each time-step, the prediction of the forthcoming note in the sequence is done by choosing the note that maximizes the likelihood given the context i.e.

$$\hat{x} = \underset{j \in |V|}{argmax} \quad P(x_{i+1} = v_j | C_i) \tag{1}$$

where C_i is the merged context representations learned by the forward and backward sequences in the bi-directional LSTM layers. Probability of a note at a time-step is computed by the `softmax` function as,

$$P(x_{i+1} = v_j | C_i) = \frac{exp({U_j}^T C_i + b_j)}{\sum_{k=1}^{|V|} exp({U_k}^T C_i + b_k)} \tag{2}$$

where U is the weight matrix in the `softmax` layer and b_j is bias term corresponding to note v_j.

The embedding layer is initialized randomly and during training, errors (in terms of cross-entropy) are back propagated upto the embedding layer, resulting in the updation of the embedding-matrix. Cross-entropy is computed as,

$$\frac{1}{M \times T} \sum_{i=1}^{M} \sum_{t=1}^{T} cross\_entropy(y_t^i, \hat{y}_t^i) \tag{3}$$

$$cross\_entropy(y, \hat{y}) = - \sum_{p=1}^{|V|} y_p \; log \; \hat{y}_p \tag{4}$$

Where M is the number of note sequences in a raga and T is the sequence length. y_t^i denotes the expected distribution of i^{th} note sequence at time-step t and $\hat{y}_t^i$ denotes the predicted distribution. Since our main objective is to learn semantic representation of notes through note-embeddings (and not predict note sequences), we do not heavily regularize our system. Moreover, our network design is inspired by Mikolov *et al.* [18], who also do not heavily regularize their system while learning word-embeddings.

4.1 Raga Similarities from Note-embeddings

For each raga our network learns a $|V| \times d$ matrix representing $|V|$ note-embeddings. We compute (dis)similarity between two ragas by computing *pairwise cosine distance* between embedding vectors of every note in V and then averaging over all notes. This is based on the assumption that distributed representations of notes (as captured by the embeddings) will be similar across ragas that are similar. The choice of cosine similarity (or cosine distance) for computing the similarity between the note-embeddings is driven by its robustness as a measure of vector similarity for vectors and its predominant usage for measuring word embedding similarity [20]. Appropriate distance measures have been adopted for non-LSTM based baselines.

5. BASELINES FOR COMPARISON

To confirm the validity, we compare our approach with a few baseline approaches.

5.1 N-gram based approach

The N-gram based baseline creates an n-gram profile based on the count of each n-gram from the available compositions in a raga. We compute the n-gram for n ranging from 1 to 4. The distance between two ragas is computed using the out-of-place measure described in Cavnar *et al.* [7]. Out-of-place measure depends on the rank order statistics of the two profiles. It computes how far 2 profiles are out-of-place *w.r.t* the n-gram rank order statistics. The distance is taken as the l_2 norm of all the n-gram rank differences, normalized by the number of n-grams. Intuitively, the more similar two ragas are, more would the N-gram profiles overlap, reducing the l_2 norm.

5.2 Pitch Class Distribution (PCD)

This method computes the distribution of notes from the count of notes in a raga's *bandish* dataset. 36 notes(across 3 octaves) are considered separately for computing PCD. As the method describes, sequence information is not captured here. The similarity distance between two ragas is computed by taking the *euclidean distance* between the corresponding pitch class distributions; the assumption is that each pitch class two similar ragas will share similar probability value, thereby reducing the euclidean distance. For the raga recognition task by Chordia *et al.* [9], euclidean distance is used for computing the distance between pitch class distributions in one of their approaches. This baseline is to verify the relevance of sequence information in capturing raga similarity.

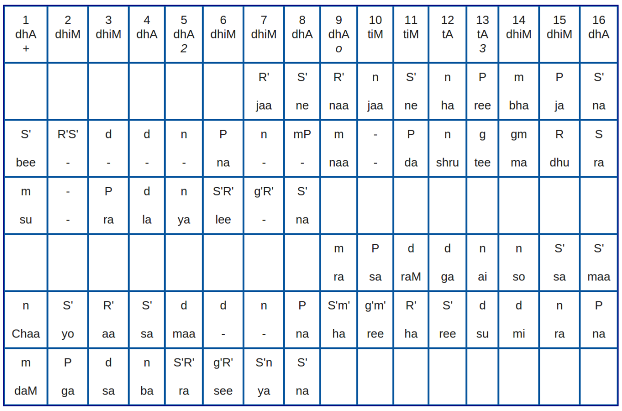
5.3 Uni-directional LSTM

The effectiveness of a bi-directional LSTM for modeling Hindustani music is verified with this baseline. The architecture is same as described in Figure 1, except for the replacement of bi-directional LSTMs with uni-directional LSTMs. Since there is only forward pass in uni-directional

LSTM, the merge operation in bi-directional LSTM design is not required here.

6. DATASET

Our experiments are carried out with the Hindustani *bandish* dataset available from `swarganga.org`, created by Swarganga music foundation. This website is intended to support beginners in Hindustani music. This has a large collection of Hindustani *bandishes*, with lyrics, notation, audio and information on raga, *tala* and *laya*. Figure 2 shows a *bandish* instance from swarganga. The name of this *bandish* is *'jaane naa jaane haree'* in raga *Adana* and in *teen taal* (16 beats cycle). The first row contains the *bol* information which details the tabla strokes corresponding to the *tala* of the *bandish*. Other rows have lyrics (bottom) along with the notes (top) corresponding to the lyrical sections. Each row corresponds to a *tala* cycle. In Hindustani notation system *S r R g G m M P d D n N* corresponds to C C$^{\#}$ D D$^{\#}$ E F F$^{\#}$ G G$^{\#}$ A A$^{\#}$ B notes in western music notation system, when the tonic is at C. A note followed by a single quotation at the right shows it is in the higher octave and a single quotation at the left implies lower octave. Notes mentioned within parenthesis are *kan* notes (grace notes). Each column represents a beat duration.



1 dhA +	2 dhiM	3 dhiM	4 dhA	5 dhA *2*	6 dhiM	7 dhiM	8 dhA	9 dhA *o*	10 tiM	11 tiM	12 tA	13 tA *3*	14 dhiM	15 dhiM	16 dhA
						R' jaa	S' ne	R' naa	n jaa	S' ne	n ha	P ree	m bha	P ja	S' na
S' bee	R'S' -	d -	d -	n -	P na	n -	mP -	m naa	- -	P da	n shru	g tee	gm ma	R dhu	S ra
m su	- -	P ra	d la	n ya	S'R' lee	g'R' -	S' na								
								m ra	P sa	d raM	d ga	n ai	n so	S' sa	S' maa
n Chaa	S' yo	R' aa	S' sa	d maa	d -	n -	P na	S'm' ha	g'm' ree	R' ha	S' ree	d su	d mi	n ra	P na
m daM	P ga	d sa	n ba	S'R' ra	g'R' see	S'n ya	S' na								

Figure 2. A *bandish* instance from swarganga website.

From this dataset we have considered 144 ragas for our study which are represented well with sufficient number of *bandishes*. Table 1 presents dataset statistics.

#bandishes	#ragas	#notes	#kan swaras (grace notes)
2955	144	2,95,411	50,749

Table 1. Dataset

6.1 Data Pre-processing

We take all *bandishes* in a raga for training the note-embeddings for the raga. *Kan* notes are also treated in the same way as other notes in the composition, since the *kan* notes also follow the raga rules. The notes are encoded into 36 unique numbers. The notes corresponding to a *tala* (rhythm) cycle is taken as a sequence. The input sequence length is determined by taking the average length of the sequences in a raga dataset; zero-padding (to the left) and left-trimming of sequences are applied to sequences shorter and longer than the average length respectively. If the length of a sequence is more than double the defined sequence length, it is split into 2 separate sequences.

7. EXPERIMENTS

7.1 Evaluation Methods

We rely on 2 different evaluation methods to validate our approach. The first one is based on *perplexity* that evaluates how well a note-sequence generator model (neural-network based, n-gram based *etc.*) can predict a new sequence in a raga. Since note-embeddings are an integral part of our architecture, a low-perplexed note-sequence generator model should learn more accurate note embeddings. The second method relies on *clustering* of ragas based on different raga-similarity measures computed using our approach and baselines.

7.1.1 Perplexity

Perplexity for a language model [2], is computed based on the probability values a learned model assigns to a validation set [10]. For a given model, perplexity (PP) of a validation set with notes $N_1, N_2, ..., N_n$ is defined as

$$PP(N_1, N_2, ..., N_n) = \sqrt[n]{\frac{1}{P(N_1, N_2, ..., N_n)}} \qquad (5)$$

where $P(N_1, N_2, ..., N_n)$ is the joint probability of notes in the validation set. A better performing model will have a lower perplexity over the validation set. For each raga dataset, perplexity is measured with a validation set taken from the dataset. For the LSTM based methods, the learned neural model provides the likelihood of a note, whereas the n-gram baseline uses the learned probabilities for different n-grams.

7.1.2 Clustering

For this evaluation, we take 14 ragas for which similarities between all the ragas and subsets of these ragas are known. These similarities are determined with the help of a professional Hindustani musician. The selected ragas are *Shuddha Kalyan, Yaman Kalyan, Yaman, Marwa, Puriya, Sohni, Alhaiya Bilawal, Bihag, Shankara, Kafi, Bageshree, Bhimpalasi, Bhairav and Jaunpuri*. The first clustering (`Clustering 1`) checks if all the 14 ragas are getting clustered according to their *thaat*. *Thaat* wise grouping of these 14 ragas are shown in Table 2. Since there are 6 different *thaats*, k is taken as 6 for this clustering. For the other clusterings, different subsets of ragas are selected according to the similarities to be verified. Other similarities and the ragas chosen (from the 14 ragas) to verify that are as listed below

- `Clustering 2`: *Sohni* is more similar to *Yaman* and *Yaman Kalyan* compared to ragas in other *thaats* because they share the same characteristic

Thaat	**Ragas**
Kalyan	Shuddha Kalyan, Yaman Kalyan, Yaman
Marwa	Marwa, Puriya, Sohni
Bilawal	Alhaiya Bilawal, Bihag, Shankara
Kafi	Kafi, Bageshree, Bhimpalasi
Bhairav	Bhairav
Asavari	Jaunpuri

Table 2. Thaat based grouping of the selected ragas

phrase (*MDNS*). To verify this, *Sohni, Yaman, Yaman Kalyan, Kafi, Bhairav* are considered taking k=3 and we expect the first 3 ragas to get clustered together and, *Kafi and Bhairav* in 2 different clusters.

- `Clustering 3`: Within *Kafi thaat*, *Bhimpalasi* and *Bageshree* are more similar compared to their similarity with *Kafi* because of the similarity in these ragas' characteristic phrases (*mDnS, mPnS*). To verify this, these 3 ragas are considered for clustering taking k=2 and we expect *Bhimpalasi* and *Bageshree* to get clustered together and *Kafi* in another cluster.
- `Clustering 4`: Raga *Jaunpuri* is more similar to *Kafi thaat* ragas because they differ only by a note. To verify this, *Jaunpuri, Kafi, Bageshree, Bhimpalasi, Bhairav, Shuddha Kalyan, Puriya, Bihag* are considered taking k=5. We expect *Jaunpuri* to be clustered together with *Kafi, Bageshree and Bhimpalasi* and the other ragas in 4 different clusters.

We apply these four clustering methods on our test dataset and evaluation scores pertaining to each clustering method is averaged to get a single evaluation score.

7.2 Setup

For the experiments, we consider notes from 3 octaves, amounting to a vocabulary size of 37 (including the null note). The common hyper-parameters for the LSTM based methods (our approach and one of the baselines) are kept the same. The number of LSTM blocks used in the LSTM layer is set to the sequence length. Each LSTM block has 24 hidden units, mapping the output to 24 dimensions. For all our experiments, embedding dimension is empirically set to 36. We use tensorflow (version: 0.10.0) [1] for the LSTM implementations. Note sequences are picked from each raga dataset ensuring the presence of 100 notes in total for the validation set. This size is made variable in order to accommodate variable length sequences. While training the network, the perplexity of the validation set is computed during each epoch and used for setting the early-stopping criterion. Training stops on achieving minimum perplexity and the note-embeddings at that instance are taken for our experiments.

For the clustering baseline, we employ one of the hierarchical clustering methods, agglomerative clustering (*linkage:complete*). In our setting, a hierarchical method is preferred over K-means because, K-means work well only with isotropic clusters [21] and it is empirically observed that our clusters are not always isotropic. Also when experimented, the clustering scores with K-means are less compared to agglomerative clustering for all the approaches. For implementing the clustering methods (both agglomerative and k-means) we use scikit-learn toolkit [23].

8. RESULTS

Before reporting our qualitative and quantitative results, to get a feel of how well note-embeddings capture raga similarities, we first visualize the 37 × 36 note-embedding matrices by plotting their heatmaps, higher intensity indicating higher magnitude of the vector component. Figure 3 shows heatmaps of embedding matrices for three ragas *viz.* *Yaman Kalyan*, *Yaman* and *Pilu*. *Yaman Kalyan* and *Yaman* are more similar to each other than *Pilu*. This is quite evident from the embedding heatmaps.

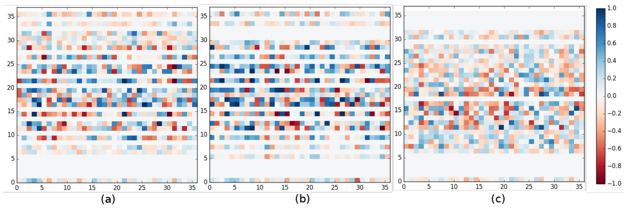


Figure 3. Note-embeddings visualization of (a) Yaman Kalyan (b) Yaman (c) Pilu

The results of quantitative evaluation is now reported with the evaluation methods described in Section 7.1. Further, a manual evaluation is done with the help of trained Hindustani musician considering all the 144 ragas mentioned in the dataset, to better understand the distinctions between bi-LSTM and uni-LSTM. Table 3 shows perplexity values (averaged across all the ragas in the dataset) with the validation set for our approach (bi-LSTM) and the baseline approaches with n-gram and uni-directional LSTM (uni-LSTM). We can not report perplexity for the PCD approach as the likelihood of the notes (and hence, the perplexity of the model) can not be determined with PCD. We observe that the perplexity values of n-gram and uni-LSTM are quite similar. The lower perplexity value with bi-LSTM shows its capability in generating a new notes sequence adhering to the raga rules. This shows the performance advantage of bi-LSTM over the baselines on note-sequence generation task, thereby providing indications on the goodness of the note-embeddings learned. Moreover, the bi-LSTM model, having the lowest perplexity, is able to capture the semantic association between notes more accurately, yielding more accurate note-embeddings.

Experiment	**Perplexity**
N-gram	6.39
uni-LSTM	6.40
bi-LSTM	**2.31**

Table 3. Results: Comparison with perplexity on validation set (Best performance in bold)

Experiment	Homogeneity	Completeness	V-measure
N-gram	0.3973	0.4036	0.4004
PCD	0.6430	0.6488	0.6451
uni-LSTM	0.7828	0.7858	0.7843
bi-LSTM	**0.9008**	**0.9069**	**0.9038**

Table 4. Results: Comparison of clustering results with different clustering metrics (Best performance in bold)

Table 4 shows the results of clustering using a standard set of metrics for clustering, *viz.* homogeneity, completeness and V-measure [26]. The clustering scores with n-gram and PCD baselines show their inability towards identifying the known similarities between the ragas. The bi-LSTM approach performs better compared to the baselines; the performance of uni-LSTM baseline is comparable with bi-LSTM approach. On analyzing each individual clustering, we observed,

- N-gram approach does not do well for all the individual clusterings, resulting in poor clustering scores compared to other approaches. A relatively better performance is observed only with `Clustering 4`.

- PCD has better scores compared to n-gram as it out-performs n-gram with a huge margin in `Clustering 1`. PCD's performance in `Clustering 1` is superior to the LSTM approaches as well. However, its performance is quite inferior to that of other approaches in the other three clustering settings. PCD's ability in modeling notes distribution efficiently helps in *thaat* based clustering (`Clustering 1`), because *thaat* based classification quite depends on the distribution of tonal material.

- uni-LSTM performance is better than bi-LSTM in `Clustering 1` where the ragas are supposed to be clustered according to the *thaat*. But it fails to cluster *Sohni*, *Yaman* and *Yaman Kalyan* in the same cluster, leading to poor performance in `Clustering 2`

- Even though bi-LSTM gives slightly lower scores with `Clustering 1`, it does perfect clustering for the other three clustering schemes. This gives an indication on the capability of bi-LSTM approach for identifying melodic similarities beyond *thaat*.

Overall, these observations show the practicality of both the LSTM based methods to learn note-embeddings with the aim of identifying raga similarity.

Figures 4 show Multi-Dimensional Scaling (MDS) [6] visualizations showing the similarity between note-embeddings of the selected 14 ragas (same color specifies same *thaat*) with bi-LSTM approach. These visualizations give an overall idea on how well the similarities are captured. The finer similarities observed in the clustering evaluations are not clearly perceivable from these visualizations.

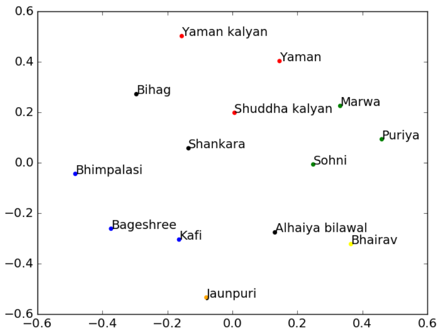


Figure 4. MDS visualization of bi-LSTM note-embeddings similarities

We have also carried out separate experiments by including note duration information along with the notes by pre-processing the data, but the performance is worse compared to the reported results. Chordia [9] has also reported that weighting by duration had no impact on their raga recognition task.

To confirm the validity of our approach, one expert musician checked the MDS visualizations of similarities between all 144 ragas with bi-LSTM and uni-LSTM approaches [1] . The musician identified clusters of similar ragas in both the visualizations matching with his musical notion. A few observations made are: *Asavari thaat* ragas appear to be closer to each other with bi-LSTM compared to uni-LSTM. Also *Miyan ki todi, Multani, Gujari Todi* which are very similar ragas are found closer in bi-LSTM. But the same *thaat* ragas *Marwa, Puriya and Sohni* are found to be more similar to each other with uni-LSTM.

9. CONCLUSION AND FUTURE WORK

This paper investigated on the effectiveness of note-embeddings for unveiling the raga similarities and on methods to learn note-embeddings. The perplexity based evaluation shows the superior performance of bi-directional LSTM method over unidirectional-LSTM and other baselines. The clustering based evaluation also confirms this, but it also shows that the performance of unidirectional approach is comparable to the bi-directional approach for certain cases.

The utility of our approach is not confined only to raga similarity; it can also be extended to verify if a given *bandish* complies with the raga rules. This immensely benefits to Hindustani music pedagogy; for instance, it helps to select the right *bandish* for a learner. In future, for better learning of note-embeddings, we plan to design a network to handle duration information effectively. The current experiments take one line in the *bandish* as a sequence. We plan to experiment with more meaningful segmentation schemes like lyrical phrase delimited by a long pause.

[1] Similarity visualizations and note-embeddings of all 144 ragas are available for download from [anonymous website]

10. ACKNOWLEDGMENTS

We would like to thank Swarganga.org and its founder Adwait Joshi for letting to use the rich bandish dataset for research. We also thank Anoop Kunchukuttan, Arun Iyer and Aditya Joshi for their valuable suggestions.

11. REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA*, 2016.

[2] Lalit R Bahl, Frederick Jelinek, and Robert L Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE transactions on pattern analysis and machine intelligence*, pages 179–190, 1983.

[3] Shreyas Belle, Rushikesh Joshi, and Preeti Rao. Raga identification by using swara intonation. *Journal of ITC Sangeet Research Academy*, 23, 2009.

[4] Ashwin Bellur, Vignesh Ishwar, and Hema A Murthy. Motivic analysis and its relevance to raga identification in carnatic music. In *Proceedings of the 2nd CompMusic Workshop; 2012 Jul 12-13; Istanbul, Turkey. Barcelona: Universitat Pompeu Fabra; 2012. p. 153-157.* Universitat Pompeu Fabra, 2012.

[5] Arindam Bhattacharjee and Narayanan Srinivasan. Hindustani raga representation and identification: a transition probability based approach. *International Journal of Mind, Brain and Cognition*, 2(1-2):66–91, 2011.

[6] I Borg and P Groenen. Modern multidimensional scaling: theory and applications. *Journal of Educational Measurement*, 40(3):277–280, 2003.

[7] William B Cavnar and John M Trenkle. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

[8] Soubhik Chakraborty, Guerino Mazzola, Swarima Tewari, and Moujhuri Patra. *Computational Musicology in Hindustani Music*. Springer, 2014.

[9] Parag Chordia. Automatic raag classification of pitch-tracked performances using pitch-class and pitch-class dyad distributions. In *Proceedings of the International Computer Music Conference*, 2006.

[10] Philip Clarkson and Tony Robinson. Improved language modelling through better language model evaluation measures. *Computer Speech & Language*, 15(1):39–53, 2001.

[11] Pranay Dighe, Parul Agrawal, Harish Karnick, Siddartha Thota, and Bhiksha Raj. Scale independent raga identification using chromagram patterns and swara based features. In *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on*, pages 1–4. IEEE, 2013.

[12] Pranay Dighe, Harish Karnick, and Bhiksha Raj. Swara histogram based structural analysis and identification of indian classical ragas. In *The 14th International Society for Music Information Retrieval Conference*, pages 35–40, 2013.

[13] Douglas Eck and Juergen Schmidhuber. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 103, 2002.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[15] Allen Huang and Raymond Wu. Deep learning for music. *arXiv preprint arXiv:1606.04930*, 2016.

[16] Adwait Joshi. swarganga.org, 2004.

[17] Manfred Junius, Alain Daniélou, Ernst Waldschmidt, Rose Waldschmidt, and Walter Kaufmann. The ragas of northern indian music, 1969.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[19] Tomas Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Extensions of recurrent neural network language model. In *Proceedings of 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5528–5531. IEEE, 2011.

[20] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 12th annual conference of the North American Chapter of the Association for Computational Linguistics*, volume 13, pages 746–751, 2013.

[21] George Nagy. State of the art in pattern recognition. *Proceedings of the IEEE*, 56(5):836–863, 1968.

[22] Gaurav Pandey, Chaitanya Mishra, and Paul Ipe. Tansen: A system for automatic raga identification. In *Proceedings of the 1st Indian International Conference on Artificial Intelligence*, pages 1350–1363, 2003.

[23] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[24] HG Ranjani, S Arthi, and TV Sreenivas. Carnatic music analysis: Shadja, swara identification and raga verification in alapana using stochastic models. In *2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 29–32. IEEE, 2011.

[25] Suvarnalata Rao and Preeti Rao. An overview of hindustani music in the context of computational musicology. *Journal of New Music Research*, 43(1):24–33, 2014.

[26] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing-CoNLL*, volume 7, pages 410–420, 2007.

[27] Surendra Shetty, KK Achary, and Sarika Hegde. Clustering of ragas based on jump sequence for automatic raga identification. In *Wireless Networks and Computational Intelligence*, pages 318–328. Springer, 2012.

[28] Rajeswari Sridhar, Manasa Subramanian, BM Lavanya, B Malinidevi, and TV Geetha. Latent dirichlet allocation model for raga identification of carnatic music. *Journal of Computer Science*, 7(11):1711, 2011.

[29] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[30] T Viswanathan and Matthew Harp Allen. Music in south india, 2004.