

Automatic Raag Classification of Pitch-tracked Performances Using Pitch-class and Pitch-class Dyad Distributions

Parag Chordia
Department of Music, Georgia Tech
ppc@gatech.edu

Abstract

A system was constructed to automatically identify raags using pitch-class (PCDs) and pitch-class dyad distributions (PCDDs) derived from pitch-tracked performances. Classification performance was 94% in a 10-fold cross-validation test with 17 target raags. Using PCDs alone, performance was 75%, and 82% using only PCDDs. Best performance was attained using a maximum a posteriori (MAP) rule with a multivariate normal (MVN) likelihood model. Each raag was divided into non-overlapping 30 second segments and pitch tracked using the Harmonic Product Spectrum (HPS) algorithm. Pitch tracks were then transformed into pitch-class sequences by segmenting into notes using a complex-domain detection function. For each note, pitch-class was determined by taking the mode of the detected pitches from the onset of the note to the next onset. For the given tuning, the nearest pitch was found based on a just-intoned chromatic scale. The comparison was made in the log-frequency domain. PCDs and PCDDs were estimated from each segment leading to 12 PCD features and 144 PCDD features. Thus, each segment was represented by a 156-dimensional feature vector, representing the relative frequency of pitch-classes and pitch dyads. It was found that performance improved significantly (+15%) when principal component analysis was used to reduce the feature vector dimension to 50. The study suggests that PCDs and PCDDs may be effective features for raag classification. However, the database size must be expanded in size and diversity to confirm this more generally.

1 Background

1.1 Raag in Indian Classical Music

Raag is a melodic concept around which almost all Indian classical music is organized. It specifies a type of melodic space within which musicians improvise. It is between a scale, which specifies a set of categorical pitches, and through-composed melodies, in which the sequence of pitches and

their relative durations are predetermined. A *raag* is most easily explained as a collection of melodic gestures and a technique for developing them. The gestures are sequences of notes that are often articulated with various micro-pitch alterations, such as slides, shakes, and vibrato. Although the gestures themselves do not have a fixed rhythm, tones are often stressed agogically and dynamically. Longer phrases are built by joining these melodic atoms together.

Music constructed from these melodic gestures will lead to notes that are not equally represented. Typically certain notes will be much more prominent, creating a tonal hierarchy. This is because certain notes appear more often in the basic phrases, or are held longer. Notes that end phrases also appear as particularly salient, although this is also usually correlated with duration. Indian music theory has a rich vocabulary for describing describing the function of notes in this hierarchy. The most stressed note is called the *vadi* and the second most stressed note, traditionally a fifth or fourth away, is called the *samvadi*. There are also terms for tones on which phrases begin, and phrase ending tones, although these are less commonly used currently. A typical summary of a *raag* includes its scale type (*that*), *vadi* and *samvadi*. A pitch-class distribution (PCD), which gives the relative frequency of each scale degree, neatly summarizes this information.

Indian classical music (ICM) uses approximately one hundred *raags*, with many less frequently used ones. Although the concept of a note is somewhat different, often including subtle pitch motions that are essential rather than ornamental, it is accurate to say that the notes in any given *raag* conform to one of the twelve chromatic pitches of a standard just-intoned scale. It is extremely rare to hear a tone sustained which is not one of the standard chromatic pitches. A given *raag* will use between five and twelve tones. From this, one can theoretically derive thousands of scale types. In practice, *raags* conform to a much smaller set of scales and many of the most common *raags* share the same set of notes.

The performance context of *raag* music is essentially monophonic, although vocalists will usually be shadowed by an

accompanying melody instrument. The rhythmic accompaniment of the *tabla* is also present in metered sections. To emphasize and make explicit the relationship of the tones to the tonic, there is usually an accompanying drone that sounds the tonic and fifth using a harmonically rich timbre.

1.2 Tonality

?) as well as ?) have shown that stable pitch distributions give rise to a mental schemas that structure expectations and facilitate the processing of musical information. Using the now famous probe-tone method, ?) showed that listeners ratings of the appropriateness of a test tone in relation to a tonal context is directly related to the relative prevalence of that pitch-class in a given key. ?) showed that decisions which required listeners to compare a test tone with a previous tone were significantly faster and more accurate when the both tones belonged to the same tonal context. ?) has shown that emotional adjectives used to describe a tone are highly correlated with that tone’s frequency in a relevant corpus of music. Further, certain qualities seemed to be due to higher-order statistics, such as note-to-note transition probabilities. These experiments show that listeners are sensitive to PCDs and internalize them in ways that affect their experience of music.

The demonstration that PCDs are relatively stable in large corpora of tonal Western music led to the development of key and mode finding algorithms based on correlating PCDs of a given excerpt with empirical PCDs calculated on a large sample of related music (?; ?; ?).

These cognitive and computational experiments suggest that PCDs and related higher-order statistics might be effective in characterizing *raag*s, which seem to exhibit tonal hierarchies.

2 Related Work

Raag classification has been a central topic in Indian music theory for centuries, and has lead to a rich debate on the essential characteristics of *raag*s, and the features that make two *raag*s similar or dissimilar (?)

Although automatic *raag* classification has been discussed previously (?), almost no actual attempts have been reported. Most of the discussed methods involved representing musical excerpts as strings of pitch-classes and then scoring them based on matches to pitch sequences postulated theoretically for different *raag*s.

?) classified thirteen *raag*s using spectrally derived PCDs. Pitch was not explicitly estimated, but instead was derived from the DFT of the segments. To determine the value of the pitch-classes for a given segment, energy was summed

in a small window near its fundamental, as well as near its harmonics and subharmonics. Visualizations of the PCDs showed that the technique was successful in capturing the basic shape of the PCD. One hundred and thirty 60 second segments were split into training (60%) and test sets (40%). Perfect results were obtained using a K-NN classifier, although the lack of cross-validation makes the perfect accuracy reported optimistic.

?) developed a system to automatically recognize two *raag*s (*Yaman* and *Bhupali*) using a Markov model (MM). Segments, in which the tonic was given, were pitch-tracked. Pitch-tracks were segmented into notes by thresholding the differentiated pitch-track data. This reduced each segment to a sequence of pitch-classes from which transition probabilities were estimated. A segment was classified by evaluating the MMs learned for each *raag* on the observed pitch-class sequence in the training data, and assigning a category based on a maximum-likelihood rule. The system was trained and tested on a small database of thirty-one test samples. On the two-target test, a success rate of 77% was reported, although the amount of training data used, the testing protocol, and the duration of the segments was not specified. To improve performance, an additional stage was added that searched for specific "catch-phrases" in the test sequences, corresponding to known patterns within each *raag*. This improved performance by 10%.

?) explored generating pitch-strings appropriate to a given *raag* by modeling *raag*s as finite automata based on rules and principles found in standard music theory texts. Although an early example of computer modeling of *raag*s, it did not attempt classification.

3 Motivation

There are several motivations for this work. The foremost is to examine whether conceptions of tonality appropriate to Western tonal music are applicable cross-culturally, and specifically to Indian classical music. If we were able to identify *raag*s using PCDs this would suggest that PCDs are important to establishing musical context in very different musical traditions. Just as the first task of music analysis is often key-finding, ICM listeners almost immediately seek to identify *raag*. Because key and *raag* are such important concepts in their respective musical cultures, showing a common underlying mechanism would be an important discovery. On the other hand, identification is not the same as characterization, and it is possible that even if PCDs and PCDDs are effective for classification that they somehow miss essential aspects of tonality.

A more immediate goal is to use *raag* in various MIR tasks. *Raag* is one of the most concise ways of conveying

information about a piece of ICM, giving substantial information about a listener’s likely psychological experience. It could be used to find performances in a specific *raag*, to find similar music, even where the user does not know the *raag* name, or to find music of a certain mood. It would also be useful in interactive work featuring ICM, such as automatic accompaniment systems for music or visualization.

4 Method

4.1 Raag Database

For this study, a database of short unaccompanied *raag* performances was recorded. Each *raag* was played on *sarod*, a fretless plucked string instrument, by a professional player living in India¹. Each *raag* was played for between four and six minutes for each of seventeen *raags* giving a total of 72 minutes of data. Popular *raags* were chosen, especially emphasizing groups of *raags* that shared similar scalar material. Samples from the collection can be heard at <http://ccrma.stanford.edu/~pchordia/raagClass/>. Table 1 shows the *raags* chosen and the set of notes used in each.

The decision to use a controlled database rather than pre-existing recordings was made to minimize the difficulties posed by accompanying instruments such as the *tanpura* and *tabla*. Preliminary experiments showed that pitch tracks in such cases were unreliable, so it was decided for this work to test the simpler case of an unaccompanied performance.

4.2 Pitch Detection

Pitch detection was done using the Harmonic Product Spectrum (HPS) algorithm (?; ?). The HPS algorithm takes the signal and creates downsampled copies at factors of 2, 3, 4, etc. up to some maximum. The resulting spectra are then multiplied together; the pitch estimate is given by the location of the maximum. Essentially, each downsampled signal gives a “shrunk”, or scaled, version of the original in the frequency domain. If a particular frequency component has a series of associated harmonics, such as we would expect if it is the fundamental, then the downsampled signals will also contain this energy at this component and it will show up as a peak after multiplication.

To be concrete, consider a simple complex with an f_0 at 100 Hz containing four partials 100,200,300,400. Downsampling by a factor of 2 gives 50,100,150,200, by 3 gives 33.33, 66.67, 100, 200, and by 4 gives 25, 50, 75, 100. In this ideal case, we see that the fundamental component is present in all

¹My sincere thanks to the brilliant *sarodiya* Prattyush Banerjee for agreeing to record these pieces

Raag Name	Notes used
<i>Desh</i>	C D E F G A B \flat B
<i>Tilak Kamod</i>	C D E F G A B
<i>Khamaj</i>	C D E F G A B \flat B
<i>Gaud Malhar</i>	C D E F G A B \flat B
<i>Jaijaiwante</i>	C D E E F G A B \flat B
<i>Kedar</i>	C D E F F \sharp G A B
<i>Hameer</i>	C D E F F \sharp G A B
<i>Maru Bihag</i>	C D E F F \sharp G A B
<i>Darbari</i>	C D E \flat F G A \flat B \flat
<i>Jaunpuri</i>	C D E \flat F G A \flat B \flat
<i>Kaushi Kanhra</i>	C D E \flat F G A \flat B \flat
<i>Malkauns</i>	C E \flat F A \flat B \flat
<i>Bhairavi</i>	C D \flat D E \flat (E) F (F \sharp) G A \flat B \flat B
<i>Kaushi Bhairavi</i>	C D \flat D E \flat (E) F (F \sharp) G A \flat B \flat B
<i>Bilaskhani Todi</i>	C d E \flat F \sharp G A \flat B
<i>Komal Asaveri</i>	C D \flat E \flat F G A \flat B \flat
<i>Shree</i>	C D \flat E F \sharp G A \flat B

Table 1: Summary of *raags* in the database. Notes are listed with C as the tonic. Rarely used notes are place in parentheses. Note that some *raags* use the same set of notes.

four signals and thus will be reinforced when they are multiplied. But it is also can be seen that if the fundamental is relatively weak compared with the second partial, then this, which appears in three of the signals, may be taken as the fundamental. Thus, a common problem with the the HPS algorithm are octave errors.

In the current implementation, each segment was divided into 40 ms frames, using a gaussian window. The frames were overlapped by 75%, so that the pitch was estimated every 10 ms. Figure 2 shows a fifteen second excerpt from a performance of *Raag Darbari*. The notes E \flat , F, G, and A \flat are clearly visible, as well as the strokes on the drone strings tuned to C’. The pitch track shows two famous *Darbari* elements, a slow heavy vibrato on on E \flat and A \flat .

4.3 Onset Detection

Note onsets in each segment were found by thresholding a complex detection function (DF) that showed sudden changes in phase and amplitude (?). First the segment was divided into 128 sample regions overlapped 50% using a rectangular window. The DFT of each region was computed and used to construct the complex DF. The DF was constructed so that transient regions would appear as local maxima. This was done by looking for regions that violated the steady-state assumption that phase changes at a constant rate, and that amplitude remain constant for a given frequency bin. If we let t

Figure 1: A pitch-tracked segment of *raag* Darbari. The horizontal grid shows the position of the pitches in the scale. The vertical lines show detected onsets.

denote the time step, then

$$A_{t+1} = A_t \quad (1)$$

and

$$\phi(k)_t = \phi(k)_{t-1} + [\phi(k)_{t-1} - \phi(k)_{t-2}], \quad (2)$$

where A is the amplitude, ϕ the instantaneous frequency, and k the bin number (?). The expected change in the instantaneous frequency was simply calculated by taking the instantaneous frequency difference between the previous two frames.

The deviations in each bin of the complex amplitude from the expected complex amplitude at time t were summed to compute the DF. A difficulty of almost all DFs is determining which local maxima are true peaks. Because changes are relative to some context, the median value of the DF was calculated in a sliding window and multiplied by an adjustable scale factor. Only peaks that exceeded this value were retained as onsets.

4.4 Pitch-class and Pitch-class Dyad Features

The detected onsets were used to segment the pitch track into notes. A note was assumed to begin at one onset and continue to the next onset. Each note was then assigned a pitch-class label. In this experiment, because the recordings were made on the same day, by the same instrumentalist, all the segments had a common tuning with the tonic equal to 261.7 Hz, or approximately middle C. The frequencies for each note in the scale were then calculated using the ratios of the just-intoned scale: 1/1 16/15 9/8 6/5 5/4 4/3 45/32 3/2 8/5 5/3 9/5 15/8. This was extended an octave above and below as well, giving a total of 36 pitches. For each note, the mode of the pitch values calculated in the 10 ms frames was used to estimate the overall pitch of the note. Taking the mode dealt quite effectively with pitch variations due to micro-pitch structure, attacks, and errors by the detection algorithm. The label of the nearest pitch in the just-intoned scale was assigned to the note, where the comparison was made in the log-frequency domain to emulate perceptual pitch distance. This resulted in transforming the pitch track into a sequence of pitches. Octave information was discarded giving a series of pitch-classes.

The PCD was computed by counting the number of instances of a given pitch-class and dividing by the total number of notes. Preliminary work showed that weighting by duration did not affect results so note durations were not used to weight the PCDs.

To determine the the PCDD the pitch-classes were arranged in groups of two (bi-grams), or in musical terms, dyads.

Since there were 12 pitch-classes there were 144 possible dyads. The number of each dyad type was counted and divided by the total number of dyads in the segment, which was just the number of notes minus one. Tables 2 and 3 show two examples, taken from *raags* that share the same set of notes.

The pitch-class transition matrix was also estimated from this data. This expressed the the probability of making a transition from one pitch-class to another. Although similar, this normalized values relative to a given pitch class. In other words, even if a pitch-class was very rare, the transition probabilities from that pitch-class would have as much weight as a much more commonly occurring one, since the probabilities were normalized to add to one for every pitch-class. It was found that this gave undue influence to rarely occurring pitch-classes significantly worsening classification performance, a good example of how subtle changes in the feature representation can lead to major differences in classification performance.

5 Classification

A total of 142 thirty second segments were used for training and testing. A 10-fold cross-validation scheme was used that split the data into training and test sets, reserving 90% for training and using the remaining 10% for testing. This was repeated ten times, each time randomly choosing segments for training and testing. Training data was used to train the classifiers as described below. The reported accuracy was the average success rate on test segments from the ten trials.

5.1 Multivariate Normal (MVN)

The feature vector was modeled as being drawn from a MVN distribution. The labeled samples from the training data were used to estimate the mean and covariance of the likelihood distribution for each class, i.e. $Pr(data|class_i)$. The covariance matrix was computed on the pooled data, rather than being estimated separately for each class. The prior probability of each class was determined by calculating the relative proportion of strokes in the training database. Using Bayes rule, the likelihood and prior probabilities were multiplied, and divided by the evidence, to calculate the posterior probability for each class: $Pr(class_i|data)$. The label was selected according to Bayes rule; the class that had the largest posterior probability was chosen.

5.2 Feed-forward neural network (FFNN)

Neural networks, rather than explicitly estimating the likelihood and prior distributions, use training examples to compute a non-linear mapping from the feature space to categories. Their advantage primarily is their ability to non-linearly combine features to represent complex decision boundaries; the decision boundaries need not be convex or contiguous. Neural networks are comprised of nodes and associated weights. Inputs to each node are multiplied by a weight and fed to a non-linear function that emits a value close to one if the input exceeds some threshold, and close to zero otherwise. In a FFNN, the input layers consists of as many nodes as there are features. For every node, each feature is multiplied by a weight and summed before being fed to the non-linear function. Each node in the input layer is connected to each node in a hidden layer, where the process is repeated. The output layer has as many nodes as there are classes. In the ideal case, when presented with a sample from a given class, the network outputs a value of one in the corresponding output node, and zero elsewhere.

The FFNN is essentially a collection of weights and non-linear functions. It turns out that the particular choice of non-linearity is usually not critical, leaving the central question of how to set the weights. This is done by the back-propagation algorithm. Weights are initially assigned randomly. During training, a sample is presented to the network, and each output node emits a value. Since we are aiming for a one at one of the nodes, and zeros at the other nodes, we can calculate the difference between the outputted values and the desired values. The square of this is our error function. The weights are adjusted to minimize the error function. In particular, we perform a gradient descent in the weight space. Details of the backpropagation algorithm can be found in (?).

Architecture plays a crucial role in the network's ability to generalize. The number of hidden layer nodes must be selected so that the FFNN is sufficiently expressive without overfitting the data. In the FFNN, the main architectural decisions concern the number of hidden layers and the number of nodes in each hidden layer. In this study, a single hidden layer with 100 nodes was used. This number of hidden nodes was determined by experimenting with FFNNs containing 30-200 hidden nodes.

5.3 K-Nearest Neighbor (K-NN)

The K-NN algorithm compares the feature vector of the segment to be classified with all the feature vectors in the training set. The distance between the test vector and the training vector is computed, usually using Euclidean distance. The most common category among the k nearest samples is assigned to the test segment.

5.4 Tree-based

A binary tree based classifier was constructed. Tree based classifiers work by considering questions that divide the training data. The tree consists of nodes and branches. At each node a question is posed that divides the data. They are essentially formalizations of a series of questions, giving them a certain intuitive appeal. At the end, we would like to have all the examples at a terminal node belong to the same class. This is called a pure node. At each node the goal is to split the data in a way that maximally purifies the data. A common way of doing this is by using an entropy criterion. A pure node has zero entropy, while a node that has several classes uniformly distributed will have normalized entropy value of one. Splits are chosen that maximally reduce entropy. Although purity can generally be increased by further splitting, overly complex trees will lead to overfitting. To avoid this, training is usually halted when the error rate begins to increase on an independent verification set.

6 Results

The best performance (94%) was obtained with the MVN classifier using using a feature vector that had been reduced to 50 dimension by PCA. Dimension reduction improved performance by 15% on average for all classifiers. The success rate for the FFNN was 75%, while the K-NN performance was 67%, and the tree-based classifier 50%. Classification was also done using either the PCD or the PCDD features alone. The success rate was 75% and 82% for the PCD and PCDD features respectively. Finally, the effect of speeding up pitch tracking by using non-overlapped windows was explored. Although this reduced the number of computations for pitch tracking by a one-fourth, the classification accuracy decreased to 55% using the MVN classifier.

7 Discussion

The 94% success rate in a fairly difficult 17 target test suggests that PCDs and PCDDs are effective means for classifying *raags*. Beyond detecting what subset of notes were used from the chromatic scale, the performance on *raags* that had identical or nearly identical scale types shows that these features represent important melodic characteristics of the *raags*. Figures ??-?? show PCDs of four *raags* with the same set of notes. It can be seen that these *raags* give rise to distinctive PCDs that reveal differing tonal hierarchies. For example, take *Desh* and *Jaijaiwante*, two *raags* that are similar in pitch-classes used and phraseology, In *Desh* the dominance of D and G compared E and F can be seen, whereas they are more equally distributed in *Jaijaiwante*. Further, the heightened

importance of B in *Desh* is obvious. The PCD for *Khamaj* shows clear peaks at E, G, and A, which is consistent with the fact that these are the terminal notes of many typical phrases (e.g., E F G, E F G A, C E F G, etc.). *Gaud Malhar*, unlike the other raags, has its main peaks at F and A. Satisfyingly, any trained Indian classical musician would recognize the clear connection between the typical *raag* phrases and the derived PCD. Further the overlaid PCDs demonstrate that the shape of PCD is fairly stable from one *raag* segments to another.

The PCDDs provide an even greater level of detail into the melodic structure of each *raag*. The PCDDs shown here were aggregated across all the segments of a given *raag*. To simplify comparison, rows and columns corresponding to notes not in the *raag* were omitted. Tables 2 and 3 show PCDDs for *Desh* and *Jaijaiwante*. For a given entry in the table, the row and column gives the first and second pitch-classes of the dyad, and the value gives the percentage of times the dyad was used relative to the total number of dyads used. For example, the sequence D E was used 3.46% of the time in *Jaijaiwante* and 1.32% of the time in *Desh*. Significant differences between the PCDDs of the two *raags* are shown in bold typeface. These differences are in accordance with each *raag*'s phrase structure. For example, E F is used much less frequently in *Desh* than *Jaijaiwante*. This is reflected in their values of .66% and 5.54% respectively. Another signature difference that is apparent here is the transition from D to F. In *Jaijaiwante*, it is usually approached through E, rather than directly, whereas in *Desh* the E is almost always skipped. Thus the D F dyad value is 2.09% in *Desh* and .92% in *Jaijaiwante*.

Although many dyad values are easily understood, others are more difficult to interpret directly and reveal many subtleties that are not simply encapsulated by other means. Also, it can be seen that many of the most common dyads are repeated notes, although some of these are no doubt due to segmentation errors. In many cases, the absolute differences may appear small (.92% vs. 2.09%), however in percentage terms the differences are large. It remains to be shown the magnitude of difference required for listener's to internalize different schemas. It is likely that smaller distinctions can be heard with increased exposure.

Table 4) shows the confusion matrix for the MVN classifier after a cross-validation test. *Raags* that were confused were nearly always very similar, and in general seemed to follow the types of errors that would be made by a non-expert human listener.

8 Conclusions and Future Work

Raag classification and characterization has played a central role in Indian music theory. The current work suggests

	C	D	E	F	G	A	b	B
C	13.9	3.63	0.33	1.65	2.09	0.22	2	1.87
D	2.42	6.49	1.32	2.09	1.98	0.22	0	0.33
E	0.33	2.97	2.64	0.66	0.11	0	0	0.44
F	1.43	0.88	1.43	3.19	1.54	0	0	0
G	3.08	0.99	0.99	0.77	3.74	0.11	0	1.87
A	0.11	0.11	0.33	0	0.66	0.22	0	0.11
b	1.65	0.33	0	0	0.55	0.55	1	0
B	3.74	0.11	0.11	0	0.55	0.11	0	0.44

Table 2: Pitch-class dyad distribution for *raag Desh*. Bold entries indicate important differences between *Desh* and *Jaijaiwante* which share the same scale. Numbers given are percentages and sum to 100% for the matrix.

	C	D	E	F	G	A	b	B
C	23.9	3.46	3	1	1.85	0.38	1	0.85
D	2.31	4.69	3.46	0.92	1.08	0.62	0	0.85
E	1.54	1.46	4	5.54	0.69	0.08	0	0
F	2.38	3.38	1.92	3.77	0.54	0.15	0	0.15
G	1.85	0.31	1	0.69	3.23	0.31	0	0.38
A	0.69	0.08	0	0.15	1	1.69	0	0.08
b	0.31	0.15	0	0	0.15	0.62	1	0
B	2.15	0	0	0.08	0.08	0	0	0.77

Table 3: Pitch-class dyad distribution for *raag Jaijaiwante*. Numbers given are percentages and sum to 100% for the entire matrix.

that PCD and PCDDs are effective ways of characterizing *raag* structure. To confirm this, a much larger, and more diverse database must be assembled, containing multiple performers and instruments. For the system to be widely applicable, it must be robust to background instruments and recording noise. The current pitch algorithm is insufficient for this task. Future work will examine ways of pre-processing the signal to improve pitch detection, perhaps by attempting source-separation through signal modeling or ICA, as well as more advanced statistically-based pitch-detection algorithms that are able to take into account the specific timbral structure of Indian instruments and vocal production.

	Bh	Bi	Da	De	Ga	Ha	Jai	Jau	KB	KK	Ke	Kh	Ko	Ma	Ma	Sh	Ti
Bhairavi	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bilaskhani	0	7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Darbari	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Desh	0	0	0	3	2	0	0	0	0	0	0	0	0	0	0	0	0
Gaud	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0
Hameer	0	0	0	0	0	12	1	0	0	0	0	0	0	0	0	0	0
Jajaiwante	0	0	0	0	2	0	12	0	0	0	0	0	0	0	0	0	0
Jaunpuri	0	0	0	0	0	0	0	13	0	0	0	0	0	0	0	0	0
KaushiBhairavi	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0
KaushiKanhra	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0
Kedar	0	0	0	0	0	0	0	0	0	0	3	0	0	0	1	0	0
Khamaj	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0
KomalAsavari	0	0	0	0	0	0	0	0	0	0	0	0	8	2	0	0	0
Malkauns	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0
MaruBihag	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	1	0
Shree	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
TilakKamod	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8

Table 4: Confusion matrix for MVN classifier in cross-validation experiment.