Assessing Efficacy of High and Low-Level Features in Niche Music Genre Classification

Noman Bashir
HEC Lausanne
MSc in Business Analytics
noman.bashir@unil.ch

Vincent Rossi
HEC Lausanne
MSc in Finance
vincent.rossi@unil.ch

***Abstract*— Music genre classification is one of the most studied aspects in audio recognition. The learnings based on this research are applied in many businesses across the world which have music or audio recommendation and listening as their main revenue stream e.g., YouTube Music and Spotify. However, most of the research in this area has been conducted on more commonplace and prevalent genres such as Rock, Pop or Classical music. Oft overlooked are genres unique to some languages – Urdu, for example. Urdu is spoken by around 100 million people across the world in both Pakistan and India and their diaspora in other countries. Urdu is one of the languages with its own unique genres such as Qawwali and Ghazal which are dissimilar to more studied genres such as Classical music or Pop. The music recommendation algorithm of Spotify uses some features which are accessible through the Spotify API. The hypothesis is that more studied genres are easily classifiable through these Spotify features (which are high-level features) but niche genres in less-studied languages cannot be classified as easily. To test this hypothesis, data and features for 3000 songs across 10 genres were extracted from Spotify using its API. An SVM model was fitted on Spotify's features. Custom features were then extracted for the rest of the 3000 songs and another model was fitted on these custom features. The accuracy of both models with respect to well-known genres and lesser-known genres was compared. Finally, a basic CNN model was built to assess if more complex models perform better on niche genres. The results show that it is difficult to distinguish niche genres based on high-level features while using both SVM and CNN. No particular genre stands out as being exceptionally difficult to classify using either high or low level features.**

Keywords—music genre classification, support vector machine, Spotify, genres, ghazal, qawwali, Urdu, neural networks

I. INTRODUCTION

Spotify is one of the world's most used music listening application and website. It has daily users of approximately 422 million people out of which 182 million are premium subscribers [1]. In the last few years, Spotify has expanded to South-East Asia and into India and Pakistan to make use of the increasing internet penetration rate and their large population to grow their subscriber base and revenue. As such, their database has expanded to include music in the languages spoken in these countries. However, including new languages comes with its own drawbacks as sometimes, these new languages have their own specific genres of music which are unlike existing genres and cannot be neatly classified into those. Urdu is one such language which is spoken by around 100 million people across the world and possesses music genres that cannot be classified into traditional ones. Qawwali and Ghazal are two such genres.

Qawwali differs from traditional genres as it is usually much longer in length – going up to 15 minutes at times. There are also lulls within one "song" where no music instrument is played and no vocals are performed. It uses similar melodic frameworks to some other subcontinental genres such as *ragas* and *talas*. Like these other genres, *qawwali* music feature evenly paced metrics, refrains and flexible solo vocal improvisations. There are fast-paced sections within one single song as well. There are also many vocables i.e., syllables without linguistic meaning within the song and syllables are assigned to specific pitches and sounds. A typical *qawwali* ensemble consists of a lead-singer and chorus of other vocalists who use their hands as a percussion instrument. The lead singer also employs the use of a harmonium [2]. *Ghazals* are somewhat similar but more poetic in nature.

With the advent of Spotify into the subcontinent, many Urdu listeners have flocked to make use of the application and its large databases. However, compared to other more prevalent genres, the music recommendation system for these newer, niche genres is not as adept. While the recommendation system does recommend more songs of the same singer, within genre recommendation is lacking. Spotify creates some features from each song file and these features are understood to be used in some manner in recommendation algorithms. They are, of course, not representative of exactly how the application recommends genres but can be used as a proxy in this case to determine the efficacy of the recommendation algorithm using lesser-known genres. These features are: (1) acousticness (2) danceability (3) energy (4) instrumentalness (5) loudness (6) speechiness (7) tempo (8) valence (9) liveness. There are other features such as mode and key but were not considered in their analysis as they are shared widely across tracks. These 9 pre-extracted features vary on a 0 to 1 scale with 1 depicting how much a track scores in the particular metric. For example, a track rated 0.9 on speechiness implies that most of the track consists of spoken words i.e., is closer to a podcast and an audio book recitation. Tracks in the *rap* genre, for example, would score higher on speechiness than tracks in *pop* while tracks classified as *classical* will have a higher instrumentalness compared to other genres.

In response to the dearth of classification research on lesser-spoken languages and the prevalent usage of Spotify in the Indian subcontinent, this paper seeks to address the following: (1) challenges with classification of music (2) the models that are usually used to classify music genres (3) data collection, cleaning and custom feature extraction used to build the SVM models (4) Comparison of SVM models built on Spotify features and custom extracted features. (5) Using extracted mfccs to train a Convolutional Neural net. Next, the paper will go over the results using the different models and evaluate whether there was any difference in classification of genres and whether some genres were harder to classify than others. Lastly, the paper discusses the limitations within the current methodology, avenues of future

research and an evaluation of the efficacy of unique feature extraction for music genre classification.

II. LITERATURE REVIEW AND RESEARCH QUESTION

A. Models and Features

There is an abundance of research on music genre classification. Support Vector Machines, in particular, have been used extensively in the early 2000s for both multi-class and binary genre classification as seen in [3] and [4]. In both cases, SVMs were used in order classify music based on several features. [4] uses only four extracted features, however. In general, various audio features are extracted from the audio files. The features extracted usually contain low-level signal properties that are not humanly interpretable such as mel-frequency spectral coefficients. On the other hand, high-level features like tempo, rhythm can be extracted as well. These features have the advantage that they can be interpretable [5].

Audio features can be classified into different groups such as *Temporal Shape*, *Temporal Feature (*zero-crossing rate is an example), *Energy Features* (global energy, harmonic energy), *Spectral Shape* (centroid, spread, spectral roll-off, mfcc) among others [6]. These are the features most commonly utilized when discussing audio feature classification. However, these are a mixture of low-level and high-level features. State of the art machine learning algorithms tend to rely on low-level features as they are more adept at classification problems. However, there has been a lot of research and recent development in music classification using high-level features that are more tangible to users. In particular, Skowronek and McKinney argue in [7] that it is possible to discriminate between several genres using only high-level features, percussiveness for example. Nonetheless, research continues on low-level features with lots of research devoted to developing improvements on features such as zero-crossing rate. The success of feature-based methods depends mainly on the discriminative power of features; thus, it comes as no surprise that more research is being devoted to this area. In terms of discriminative power, Breebart and McKinney show in [5] that features that rely on auditory perception rather than other standard high-level features are more adept at classification. It is also worth noting that MFCC, Zero-Crossing Rate and Spectral Centroids were the most successful in not only discriminating music from other types of sound, but also music within genres.

Recent advances in deep learning methods have allowed the use of CNNs for image classification. However, music researches have proposed various architecture designs mainly based on CNNs that allow for automatic music tagging as well. In [8], Costa shows that CNNs can be used and compared with SVM classification methods. As the *mfccs* can be visualized using various methodologies, this allows them to be used with CNNs. The experiments show that CNN compare favourably to other classifiers in general and hence are an interesting alternative for music genre classification.

B. Genres and Languages

Most of the research available on music genre classification covers western genres – classical appears to be a popular choice when judging the efficacy of genre classification models and features. Research on genres outside the popular domain is rare. Nonetheless, there have been attempts to research on how timbre and melody can be used to classify devotional music such as *qawwali* and other subcontinental genres in [9]. The research found that features related to tempo and timbre were effective discriminants in classifying North Indian devotional music.

Since *ghazal* and *qawwali* are more vocal genres, Huh and Miduthuri attempt to distinguish and classify vocal genres with and without accompaniment in [10]. They raise a concern that although music genre classification has come a long way, the focus has been on feature-extraction and judging the discriminative power of those low-level features. However, features which humans rely on such as rhythm, harmony and vocal content have not made an impact in the machine learning sphere for genre classification. Their results show that isolating components of the music file and then extracting features from the isolated components (especially vocals) would prove most efficient in classifying such vocal genres.

C. Conclusion and Research Question

In conclusion, advances in deep learning methodology have shown that not only are Convolutional Neural Networks generally more accurate at classification of music genres, low-level features are often better at classification even if they lack human interpretability. Features such as mel-frequency spectral coefficients, zero-crossing rate, spectroid, and chrome are extracted from audio features and used for classification tasks. There has been recent research on high-level features in machine learning and advances have been made as these features are easier to interpret. Spotify makes certain high-level features available to its users for each track and they can be utilized to see the accuracy of machine learning algorithms using high-level features.

At the same time, research into genres of other languages is limited. There has been some headway in separating vocal features from other audio features and while this has not resulted in greater accuracy of classification, there is room for further improvement.

Thus, this report will try to show that genres in languages that have not been extensively studied tend to have poorer performance than average when using high-level features as those features tend to centre around ideas like ‘danceability’ and ‘instrumentalness’ which are not central to vocalist genres such as *ghazal* and *qawwali*. The hypothesis is that low-level features will showcase a much more significant increase in accuracy for these niche genres than for more-studied genres.

III. METHODOLOGY

In order to test the aforementioned hypothesis, we will be utilizing Google and Spotify to gather data and create models in Python. Since the idea is to test whether niche genres from different languages can be classified with a much-improved accuracy when using low-level features compared to high-level features, a three-step modelling process will be utilized. The first model will only take into

account high-level features and will be a basic machine learning model rather than a convolutional neural net. The second model will be built on extracted custom low-level features such as mfcc, zero-crossing rate and chroma that have shown great discriminative power in previous research. Once again, a simple model will be used. Lastly, the extracted features will be visualized and used to train a convolutional neural net to check if increased model complexity allows us to forecast niche genres with greater accuracy.

A. Data Gathering and Preprocessing

The hypothesis will be tested by turning this into a 10-genre classification problem. The genres under consideration are 8 popular genres (classical, pop, rock, rap, blues, edm, metal, hip-hop) and two niche genres from the Urdu language (qawwali and ghazal). Approximately 300 songs from 10 genres will be downloaded, their features extracted and then evaluated based on two machine learning models. In order to get the relevant songs from each genre, Google will be utilized to get a subset of artists within that genre. Then, these artists will be looked up on Spotify and data regarding their 10 most popular songs will be extracted. Since Spotify already creates high-level features and allows access of them through their API, we will utilize this functionality to build a data frame with these high-level features for the first step in the modelling process. Spotify also allows extraction of a *preview URL* which has a 30-second snippet of the music in question. This sample will be downloaded for each song in order to build custom features. These features will be used to train the next model. The accuracy of the two models will then be compared to determine if the hypothesis is true or not.

However, all data requires preprocessing and audio signals especially so. Due to Spotify geographic and account-based restrictions, many artists and songs will not have a *preview URL* through which sample songs can be downloaded. Without a sample, it will be impossible to extract relevant features and conduct steps 2 and 3 of the modelling process. Therefore, these songs will be removed from the data frame and only those with downloadable previews be kept. Moreover, in order to run the second model on custom low-level features, the music files will be preprocessed and the following relevant features will be extracted: (1) chrome stft (2) root mean square energy (3) spectral centroid (4) spectral bandwidth (5) rolloff (6) zero-crossing rate (7) mfccs.

Lastly, for the final CNN model, the mfccs need to be visualized so that the neural network can be trained. All the downloaded music files will be preprocessed to create a .json file that contains 13 mfccs for each sample within an audio file. The model will then be trained from the resulting image.

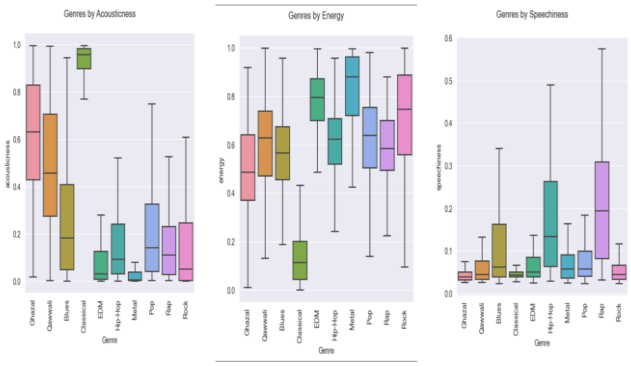
B. Description of Spotify & Custom Features

As aforementioned, Spotify allows the extraction of several high-level features directly from its API. The extracted features and a short description are presented below in **Table I.**

TABLE I. DESCRIPTION OF SPOTIFY FEATURES

Feature	Description
Acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic.
Danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity
energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy
instrumentalness	Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context.
liveness	Detects the presence of an audience in the recording.
loudness	The overall loudness of a track in decibels (dB)
speechiness	Speechiness detects the presence of spoken words in a track
tempo	The overall estimated tempo of a track in beats per minute (BPM).
valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track.

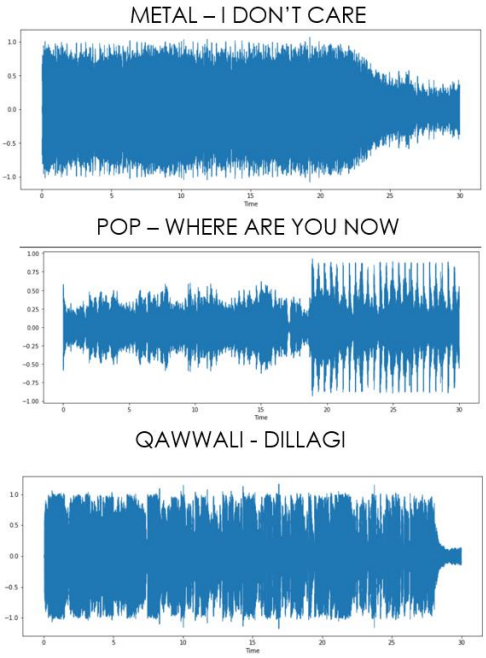
These high-level features are fairly descriptive. We can have an intuition about where some genres would fall. For example, *Classical* music would have a high instrumentalness while *Rap* would have a high speechiness. The figure below investigates the relationship between Genres and some of these features.



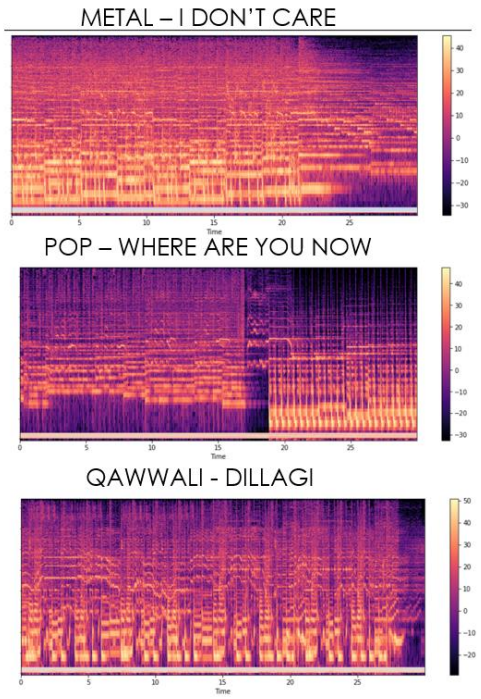
As can be seen from the figure, *ghazal* and *qawwali* rank highly on the acousticness alongside classical. They are also generally lower on the energy scale when compared with Rock, Hip-Hop and Metal. *Rap* and *Hip-Hop* dominate in terms of speechiness. Notably low on the speechiness scale is *qawwali* which is surprising since it's primarily a vocal genre. However, "ohs" and "ahs" are predominant in qawwali but are considered in instrumentalness by Spotify. There were small other distinguishing points in the other features as well.

In terms of custom features, a brief overview is presented here. The zero-crossing rate is the rate of sign-changes along with a signal, i.e., the rate at which the signal changes from positive to negative or back. This usually can be shown by zooming in on the waveform; however, differences in the waveform are also elucidating themselves. It usually has higher values for highly percussive sounds like those in metal and rock. Meanwhile, spectral centroid indicates where the "center of mass" for a sound is located and is calculated as the weighted mean of the frequencies present in the sound. Spectral rolloff represents the frequency below which a specified percentage of the total

spectral energy lies. The waveform differences between the audio files (for zero-crossing rate et al) are shown below.



Mel-Frequency Cepstral Coefficients (mfccs) are a set of features that describe the shape of the spectral envelope. Lastly, chroma frequencies project the entire frequency of an audio into 12 bins representing the 12 semitones (chroma) of the musical octave. Three songs were chosen – one each from *qawwali*, *metal* and *pop* to indicate how the spectrogram and mfccs vary for the three songs. These spectrograms are below with the log of frequency on the x-axis and time on the y-axis



We can see that there is greater variety within these features than there was in the Spotify features. Theoretically, these custom features should allow for better classification.

C. Models Considered and Hyperparamter Tuning

The models under consideration for this report are Support Vector Machines and Convolutional Neural Networks. SVM was chosen as the base comparison model because of its prevalence in the music genre classification space. SVMs attempt to find a hyperplane in dimensional space that separates classes better. SVMs offer good accuracy and perform faster predictions compared to Naïve Bayes or lazy-learners like KNN. As we have seen from the evaluation of Spotify features, there is not a lot of difference between the means/medians of most genres when considering the 9 features that have been extracted. SVM's usage of kernels to transfer this data to a higher dimensional space so they become separable without sacrificing performance means that it is an excellent choice when datapoints are not easily separable. Moreover, SVM only requires the support vectors to classify a point in each genre. Thus, with a large dataset such as this one, it would be optimal to use SVM. Nonetheless, SVM does have its drawbacks. It is usually used in binary classification instead of multi-class classifications and does not work well with extremely large datasets. However, we believe that the number of instances in our dataset are low enough that SVM should outperform most other models computationally.

SVMs can utilize several different kernels in order to transform the data and find a hyperplane that allows the items to be linearly separable. The kernels under consideration that will be tuned as hyperparameters are: polynomial, linear, sigmoid and radial. Other hyperparameters to be tuned are: gamma (how much observations are approached), C (regularization parameter) and degree (if polynomial kernel is selected).

Advancements in GPUs have resulted in Neural Networks being used widely in many contexts. The application of CNNs in music genre classification is a curious development since they are mainly used for image classification. However, because of their layering of convolutional and pooling, they manage to reduce dimensionality whilst keeping the most relevant features of an instance (unlike multi-layer perceptrons where each neuron is connected to each neuron in the preceding and succeeding layers) and thus have high accuracy. The fact that audio signals can be converted into images using their mfccs means that they can be used with CNNs in order to forecast.

CNNs have many hyperparameters that can be tuned. However, due to computational complexity and the large data size, the only hyperparameters under consideration are: number of filters in each convolutional layer, the learning rate and the number of epochs.

D. Evaluation Criteria

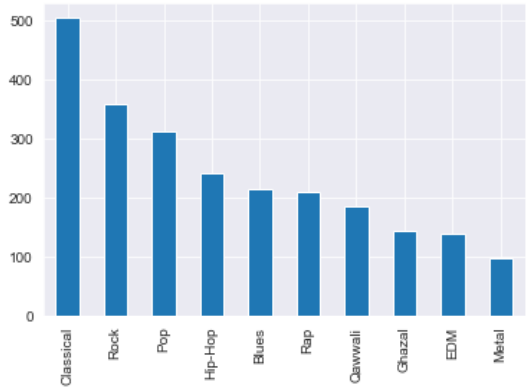
The models will be evaluated on accuracy. However, accuracy for multi-classification models is not the best measure. When you have a lot of different classes, the classification model might not be able to predict the right class exactly. This especially becomes a problem in cases where you have a vast number of features, and the data is not adequately clustered across classes (as is the case in this dataset). Therefore, just looking at accuracy might be misleading at times. Therefore, F-1 score will also be calculated which is commonly used as a scoring metric for multi-class classification problems. However, the real

measure of interest is how well each model classifies lesser-known genres such as *qawwali* and *ghazal* and which other genres confound these ones.

IV. DATASET

The data for this project has been accessed through a multi-step process. The first step in the process was to manually google top artists across the genres in consideration and add them to a csv file along with their genres. This step was undertaken since the main source to be used in this project is Spotify and Spotify does not classify tracks with genres. This data file contains about 100 singers divided across the 10 genres under consideration and is stored in *singers.csv*.

The rest of the data has been accessed using the Spotify API available at https://developer.spotify.com/documentation/web-api/libraries/. The API was accessed using the *spotipy* library for python. https://github.com/plamere/spotipy. This is done so that the same songs can be analyzed for both high-level (Spotify) features and low-level (extracted) features. The data is retrieved and parsed into a data frame from which various useful features can be extracted. First, we extracted the Artist URI (URI is a Spotify specific term referring to an identifier in their system) which allows searching for all albums and all tracks for those artists available in the Spotify database. This also allows us to retrieve a collection of 10 songs with a preview URL linking to a 30-second sample of the songs in question. All this information was stored in a data frame and joined together with our singers list from *singers.csv*. After the tracks had been joined together into a new data frame, the Spotify features (danceability, acousticness et al) are extracted by pinging the API for these features. This database is then compiled, missing values are dropped and stored in a csv file called *subsampled.csv*. Below is a graph that shows the number of songs extracted for each genre.

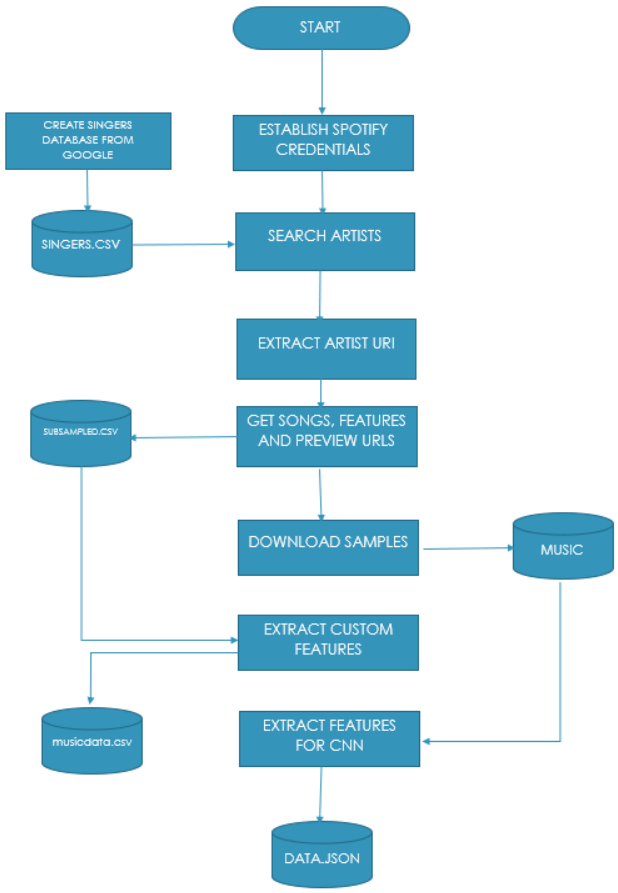


Regrettably, a lot of artists in the *Qawwali*, *Ghazal*, *EDM*, and *Metal* categories did not possess preview URLs due to either regional restrictions or not having allowable previews despite manually adding albums to be downloaded. However, this imbalance will be addressed using stratification later on.

After this data has been extracted, the preview URLs are used to download the 30-second samples of all the songs in the database. The entire song dataset was stored in a folder called *music* with each audio file being stored as an *.mp3* and uploaded on Google Drive with subfolders for each genre. The data can be found at: https://drive.google.com/drive/folders/1htpfeyNDVqkUQdTJUkDzBde5KdJmjrN-?usp=sharing.

Following this extraction of Spotify features and music files, the next step was to extract the custom features (rmse, rolloff et al). To do this, the *librosa* library was used in conjunction with the *os* library that allows one to iterate through the directors and store each song in the database. Since we are only interested in a select few features previously mentioned in Section III Part A, only those features will be extracted and stored. The extraction process was lengthy and the data was stored in *musicdata.csv*.

Finally, since only images can be used with CNNs, the mfccs for each audio file will be extracted and stored into a .json file called *data.json*. Since there are multiple mfccs for each second and a total of 20 mfccs for 30 seconds, the data is 3-Dimensional and cannot be stored in a csv file. Once again, the *librosa* library was used to extract the relevant features for extraction. This entire process can be summarized in the flowchart below:



V. IMPLEMENTATION

As aforementioned, all the data was gathered using the *spotipy* library and the Spotify API. A basic exploratory data analysis was conducted using seaborn and matplotlib to identify correlations between the pre-extracted high-level features from Spotify. The *librosa* library was used and the relevant features were extracted using its predefined commands. However, for the CNN model, some description of the way the features were extracted has to be made as that has an impact on the overall result of the process. The librosa library allows you to set a sampling rate i.e., the number of

samples that can be taken from a continuous file. By default, this is at 22k and we used the default value. Next, in order to aggrandize the dataset, you can pick the number of samples per audio file as a whole. In this case, since the audio file is 30 seconds long, we decided to divide it into 10 parts to increase our dataset but to not increase computational complexity too much. Thus, the total sample was set to 10. A larger sample would result in more data points and perhaps a more efficient model.

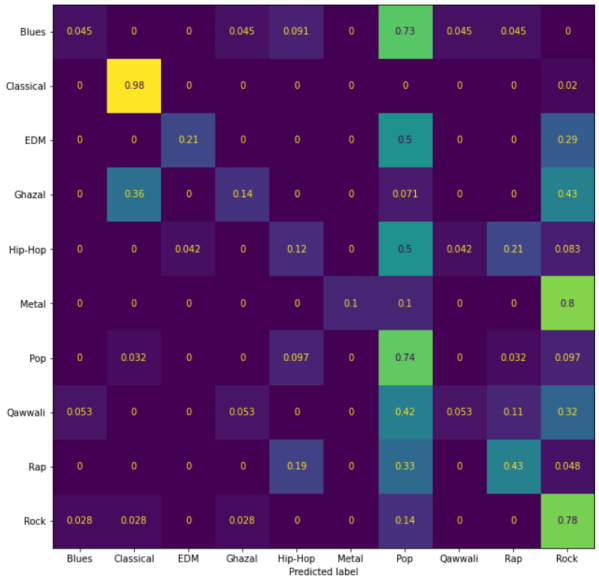
As mentioned previously, the dataset we have is imbalanced. However, we decided not to resample for two reasons: (1) it would increase computation time extraordinarily on our limited resources (2) we expect the number of *qawwali* songs in the world to be lower than the number of *pop* songs in any case thus stratification seemed like a good choice with a 10-fold CV at the end to make sure there were no blind spots in the results.

For the first test on Spotify features - data was split into training and test sets using *sklearn's* train_test_split with a 90-10 ratio (we felt we had enough samples to go with a smaller sized test-set). An SVM was fitted on the training data without any hyperparameter tuning. The accuracy was 48%. This seemed like a low accuracy compared to what we expected even from our base model. Next, we used GridSearchCV to tune our hyperparameters (gamma, degree, kernel and C). The best parameters were extracted and the model was rerun on the tuned hyperparameters. The improved model had an accuracy of 50% and an F1 Score of 45%. The classification report can be seen in Table II.

TABLE II. BASIC SVM CLASSIFICATION REPORT

Genre	Metrics		
	Precision	***Recall***	***F1-Score***
Blues	0.25	0.09	0.13
Classical	0.86	1.00	0.93
EDM	0.50	0.36	0.42
Ghazal	0.40	0.14	0.21
Hip-Hop	0.18	0.12	0.15
Metal	0.00	0.00	0.00
Pop	0.29	0.65	0.4
Qawwali	0.5	0.11	0.17
Rap	0.4	0.29	0.33
Rock	0.46	0.69	0.56

The confusion matrix can be seen below:

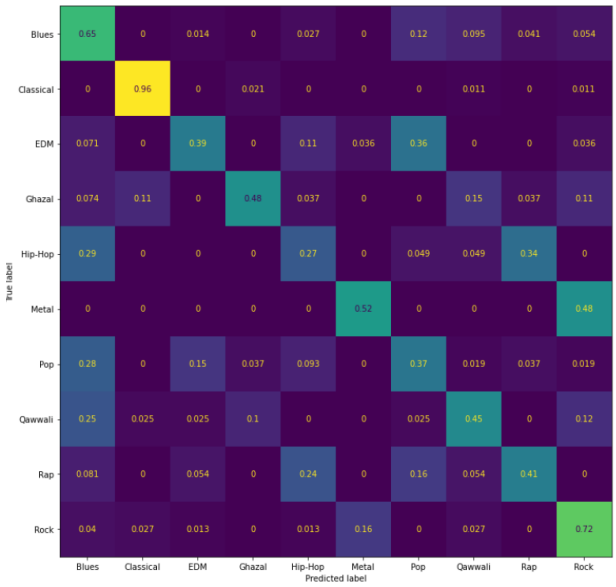


Following this basic model, the second model was also built using SVM from *sklearn*. Data was similarly divided into training and test split with stratification. A basic model without hyperparameter tuning was trained. The model returned an accuracy of 55%. Thereafter, hyperparameter tuning was done using GridSearchCV as in the last model, and the model was retrained and tested using the test data. The tuned model performed better than the tuned basic feature model, resulting in an accuracy of 59% and a f1 score of 59%. The classification report is in Table III

TABLE III. EXTRACTED FEATURE SVM

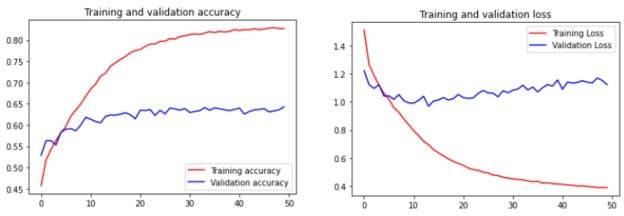
Genre	Metrics		
	Precision	***Recall***	***F1-Score***
Blues	0.51	0.65	0.57
Classical	0.94	0.96	0.95
EDM	0.46	0.39	0.42
Ghazal	0.62	0.48	0.54
Hip-Hop	0.34	0.27	0.30
Metal	0.48	0.52	0.50
Pop	0.42	0.37	0.39
Qawwali	0.49	0.45	0.47
Rap	0.43	0.41	0.42
Rock	0.68	0.72	0.70

The confusion matric can be seen below:

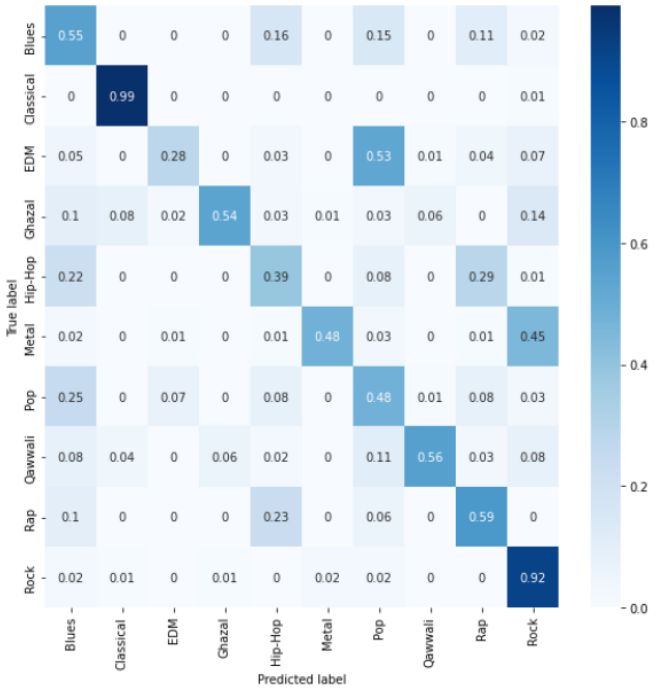


Finally, the CNN model was trained using the data from *data.json*. Due to computational constraint, the model was kept relatively simple. The model has 3 convolutional layers followed by 3 pooling layers. At the end of each layer, we include batch normalization as well. The third layer of convolutional also has a dropout of 0.3 added. The model was built using the *tensorflow* and *keras* libraries. Some of the important things to consider when building a convolutional neural network are the kernel size and the stride. The stride determines how many horizontal and vertical pixels the kernel moves during a convolution. The kernel size is the size of the grid through which convolution takes place. For this model, we kept both static at a stride of (2,2) i.e., two horizontal movements and two vertical ones. The kernel size was kept at (3,3) for the first two convolution layers but (2,2) for the third and last convolution layer as we want to reduce dimensionality more towards the end.

In order to finetune the hyperparamters, keras_tuner was used. The hyperparameters to be tuned were the number of epochs, the number of filters in each of the convolutional layers and the learning rate. The algorithm used to calculate gradient descent was Adam and the loss function was sparse categorical entropy. This was a stepped procedure where the model was first trained over 30 epochs and the best values were extracted. We also implemented an early stopping procedure with a patience of 5 so the model did not overfit. After we got the best hyperparameter values (384 filters and a learning rate of 0.0001), the model was fit again using a validation split of 0.2 to find the ideal number of epochs. This training process is outlined in the figures below:



The hyperparameter tuning decided on a best tune of 50 epochs. These final tuned hyperparameters were then used to train the final model on the training data. The CNN model gave the highest accuracy of 62.5% - a 3.5% increase over the SVM model. The confusion matrix of this model is below:



VI. RESULTS

Based on our analysis, we can make a few clear observations: (1) high-level features such as those extracted and provided by Spotify are not good at making any general predictions. The overall accuracy of the model is only 50% even after tuning of hyperparameters. There are a few interesting observations within the confusion matrix however. It seems that based on these pre-extracted features, *Classical* and *Rock* music can be predicted with great accuracy. Almost all of the *Classical* music pieces in the test set were predicted and the same can be said of *Rock*. However, it is interesting to note that *Metal* was completely confounded with *Rock*. Similarly, the accuracy of *Blues* was remarkably low as the model predicted the genre to be *Pop* most of the time.

As far as our hypothesis of niche genres being poorly predicted, it holds out in this version. While *Metal* was the worst predicted genre, *Qawwali* was predicted correctly only 5% of the time. Surprisingly, it was not confounded with *Classical* with which it shares some features but with *Rock*. *Qawwali* had a high precision compared to the other genres i.e., the songs predicted as *Qawwali* were indeed *Qawwali* half the time. However, the recall i.e., total detected vs actual detected was one of the poorest. Similarly, *Ghazal* was also conflated with *Rock* most of all. It also had a low accuracy. Nonetheless, due to the poor overall accuracy and the high precision scores for *Qawwali*, we cannot safely conclude that high-level features are any worse at predicting niche genres than other genres, despite the fact that *Qawwali* had the second-lowest accuracy. Intuitively, some of these results make sense. *Rap* and *Hip-Hop* are closely related due to their speechiness and *Metal* and *Rock* are also similar sounding to the human ear. The benefit of high-level features is that they are

interpretable by humans. This can be noted in the fact that these features are good at classifying popular songs i.e., the *Pop* genre.

In terms of the second model, it is obvious that low-level features were far more effective at predicting the genre than high-level features were. The accuracy went up by 9% and the f1 score by 10%. The low-level features also appear to be more adept at distinguishing between *Blues* and *Pop*, and *Metal* and *Rock*. Both our niche genres also improve in predictability. The remarkable observation here is that *Pop* has a markedly lower accuracy with low-level features. The accuracy halves from 74% to 37%. Once again, this result does make sense. Spotify's features are built for listening for mainstream audiences i.e., most of the tracks on Spotify will be linked to *Pop* invariably. Therefore, the Spotify features are better at predicting *Pop*. Nonetheless, it comes as a surprise that the accuracy has halved. It perhaps points to the fact that *Pop* as a genre is not as distinct as *Classical* or *Rap* for example. While it's higher-level features may be unique (e.g., danceability), it is very likely that it borrows from other genres and thus cannot be neatly classified. Therefore, there will be a lot of confounding at the lower level.

The results of the CNN appear to provide the most lucid confusion matrix. Most of the accuracies have increased and we can see that almost all genres are being predicted with an accuracy of around 50%. *Rock* and *Classical* remain easily distinguishable here as well. This result shows us the relationship between genres better than the two earlier results. *Pop* is most confused with *EDM*. *Metal* is confused with *Rock*. Lastly, *Hip-Hop* and *Rap* also appear to closely related. To the human ear, these results do not seem all that surprising as we are aware of the nature of these genres.

VII. CONCLUSION

This report has attempted to classify well-known and niche genres in music into their respective categories. This was done in order to test the hypothesis whether niche genres can be easily classified using high-level features in comparison to low-level features. However, the results indicate that all genres are difficult to classify using high-level features and while these niche genres have the worst accuracy, it does not necessarily mean they require more discriminating features due to their nature.

However, there are certain limitations with this study that need be addressed. First, the genre and artists were manually selected. It is entirely possible that one artist may sing in more than one genre which would add complexity to the model that was not considered. Moreover, choosing artists manually means only the most popular artists will be considered rather than a wider representative sample. Since popularity depends on how "mainstream" an artist is, it is likely that the music they make have a "averageness" to them i.e., they may not fit into one genre completely but may have features that average together all the features of multiple genres i.e., they might be danceable, instrumental and acoustic at the same time. Next, only ten genres were considered and only two genres from a lesser-spoken language were taken into account. It is possible that with more genres from niche languages, the models and their predictive capabilities improve. Lastly, only a basic CNN model was built due to computational complexity and lack of resources. A more structured and fine-tuned model may perform better.

In terms of further research, there are quite a few avenues to explore. More languages could be considered and more varied genres rather than just vocal genres that were used in this report. Moreover, more features could be calculated. We only calculated the most well-known of features that already have solid research behind them. The possibilities of testing new features unique to languages are abundant. In [10], Huh show that separating sound from vocals may be a path to improving accuracy for subcontinental genres.

Music genre classification is difficult to perfect as it requires extraction of features before the models can be trained. Moreover, there are multitudes of genres with different types of sounds that are difficult to accurately classify with the available features. However, this is a rapidly developing field with newer algorithms and features that help classify music better being developed. We hope to have scratched the surface of whether genres in different languages require newer, better discriminating feature extraction.

REFERENCES

[1] Spotify Investor Hub. *Spotify.com.* [Online]. Available : https://investors.spotify.com/home/default.aspx [Accessed 26 May 2022]

[2] British Broadcasting Company "A Beginner's Guide to Qawwali Music" bbc.com. [Online]. Available : https://www.bbc.co.uk/programmes/articles/5Plm8bBlBd7wXjZN2zdb8Fm/a-beginners-guide-to-qawwali-music [Accessed 26 May 2022]

[3] Xu, Changsheng, et al. "Musical genre classification using support vector machines." 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).. Vol. 5. IEEE, 2003.

[4] Liu, Jing, and Lingyun Xie. "SVM-based automatic classification of musical instruments." 2010 International Conference on Intelligent Computation Technology and Automation. Vol. 3. IEEE, 2010.

[5] McKinney, Martin, and Jeroen Breebaart. "Features for audio and music classification." (2003).

[6] Peeters, Geoffroy. "A large set of audio features for sound description (similarity and classification) in the CUIDADO project." CUIDADO Ist Project Report 54.0 (2004): 1-25.

[7] Skowronek, Janto, and Martin McKinney. "Features for audio classification: Percussiveness of sounds." Intelligent algorithms in ambient and biomedical computing. Springer, Dordrecht, 2006. 103-118.

[8] Costa, Yandre MG, Luiz S. Oliveira, and Carlos N. Silla Jr. "An evaluation of convolutional neural networks for music classification using spectrograms." Applied soft computing 52 (2017): 28-38.

[9] Kini, Sujeet, Sankalp Gulati, and Preeti Rao. "Automatic genre classification of North Indian devotional music." 2011 National Conference on Communications (NCC). IEEE, 2011.

[10] Huh, Bryan, and Arun Miduthuri. "VOCAL-BASED MUSICAL GENRE CLASSIFICATION."