WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

MUSIC GENRE RECOGNITION WITH DEEP NEURAL NETWORKS

[1]P CHARAN TEJA REDDY, [2] I CHARAN, [3]A DEEPIKA, [4] B S VARUN BHAT,[5] T DHARMA TEJA, Prof R.A. Manikandan [6]

UG Students, Department of Artificial Intelligence and Machine Learning (AI&ML) Malla Reddy University, Maisammaguda, Hyderabad.
[6] Assistant Professor,
Department of Artificial Intelligence and Machine Learning (AI&ML), Malla Reddy University, Maisammaguda, Hyderabad.
[1] 2111cs020111@mallareddyuniversity.ac.in, [2] 2111cs020112@mallareddyuniversity.ac.in,
[3]2111cs020117@mallareddyuniversity.ac.in, [4] 2111cs020118@mallareddyuniversity.ac.in,
[5]2111cs020121@mallareddyuniversity.ac.in

ABSTRACT :

We discuss the application of convolutional neural networks and convolutional recurrent neural networks for the task of music genre classification. We focus in the case of a low computational and data budget where we cannot afford to train with a large dataset. We start using a well-known architecture in the field and we use transfer learning techniques to adapt it to our task. Different strategies for fine-tuning, initializations and optimizers will be discussed to see how to obtain the model that fits better in the music genre classification. Moreover, we introduce a multiframe approach with an average stage in order to analyze in detail almost the full song. It is used at training time to generate more samples and at test time to achieve an overview of the whole song. Finally, we evaluate its performance both in a handmade dataset and in the GTZAN dataset, used in a lot of works, in order to compare the performance of our approach with the state of the art.

Index Terms— Music genre classification, recurrent neural networks, convolutional neural networks

INTRODUCTION :

Music genres are a set of descriptive keywords that convey high-level information about a music clip (jazz, classical, rock...). Genre classification is a task that aims to predict music genre using the audio signal. Being able to automatize the task of detecting musical tags allow to create interesting content for the user like music discovery and playlist creations, and for the content provider like music labeling and ordering.
Building this system requires extracting acoustic features that are good estimators of the type of genres we are interested, followed
by a single or multi-label classification or in some cases, regression stage. Conventionally, feature extraction relies on a signal processing front-end in order to compute relevant features from time or frequency domain audio representation. The features are then used as input to the machine learning stage. However, it is difficult to know which features are be the most relevant to perform each task. The recent approaches using Deep Neural Networks (DNNs), unify feature extraction and decision taking. Thus allow learning the relevant features for each task at the same time that the system is learning to classify them.
Several DNN-related algorithms have been proposed for automatic music tagging. In [1] and [2], spherical k- means and multi-layer perceptrons are used as feature ex- tractor and classifier respectively. Multi-resolution spectrograms are used in [1] to leverage the information in the audio signal on different time scales. In [2], pretrained weights of multilayer perceptrons are transferred in order to predict tags for other datasets. A two- layer convolutional network is used in [3] with mel-spectrograms as well as raw audio signals as input features.

CNNS AND CRNN FOR MUSIC CLASSIFICATION

As most of the works, we are using the mel- spectograms of the music signals as an input to our system. For this reason, we focus on neural networks that have been designed to cope with images.

Convolutional Neural Networks

Convolutional neural networks (CNNs) have been actively used for various music

classification tasks such as music tagging [3] [4], genre classification [5] [6], and user-item latent feature prediction for recommendation [7]. CNNs assume features that are in different levels of hierarchy and can be extracted by convolutional kernels. The hierarchical features are learned to achieve a given task during supervised training. For example, learned features from a CNN that is trained for genre classification exhibit low- level features (e.g., onset) to high-level features (e.g., percussive instrument patterns) [8].

RecurrentConvolutional Neural Networks

Recently, CNNs have been combined with recurrent neural networks (RNNs) which are often used to model sequential data such as audio signals or word sequences. This hybrid model is called a convolutional recurrent neural network (CRNN). A CRNN can be described as a modified CNN by replacing the last convolutional layers with a RNN. In CRNNs, CNNs and RNNs play the roles of feature extractor and temporal summariser, respectively. Adopting an RNN for aggregating the features enables the networks to take the global structure into account while local features are extracted by the remaining convolutional layers. This structure was first proposed in [9] for document classification and later applied to image classification [10] and music transcription [11]. CRNNs fit the music tagging task well. RNNs are more flexible in selecting how to summarise the local features than CNNs which are rather static by using weighted average (convolution) and subsampling.

Literature Review:

Music genre recognition (MGR) is a critical task in music information retrieval (MIR), aiming to categorize music tracks into predefined genres. Traditional approaches to MGR relied heavily on hand-crafted features such as mel-frequency cepstral coefficients (MFCCs), chroma features, spectral contrast, and rhythmic patterns. These features were then fed into classical machine learning algorithms like Support Vector Machines (SVMs), k-Nearest Neighbors (k-NN), and Gaussian Mixture Models (GMMs). While these methods provided reasonable accuracy, their performance was limited by the quality of the extracted features and the models' ability to capture complex patterns in music data.

Network architectures

In this work we will take as a starting point the architectures of a CNN and the CRNN proposed by Choi et al. in [4], [12]. The first one is a fully convolutional neural network with 5 convolutional layers of 33 kernels and max-pooling layers ((2 × 4)-(2 × 4)-(2 × 4)- (3 × 5)-(4 × 4)) as illustrated in Figure 1a. The network reduces the size of feature maps to 1
× 1 at the final layer, where each feature covers the whole input. This model allows time and frequency invariances in different scale by gradual 2D sub-samplings. Being fully convolutional reduces considerably the number of parameters.

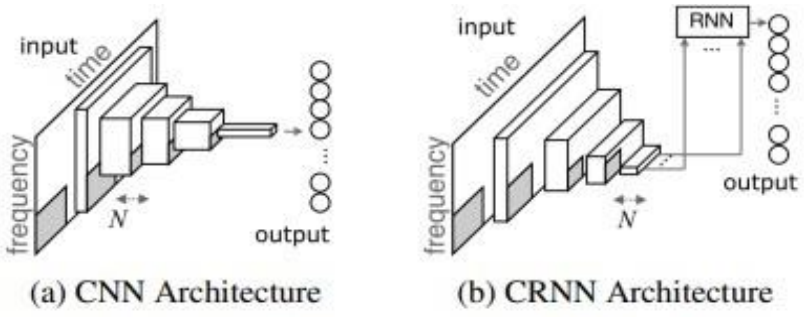

Fig. 1: Network Architectures from [4] and [12]

The second architecture uses a 2-layer RNN with gated recurrent units (GRU) [13] to summarize temporal patterns on the top of two- dimensional 4-layer CNNs as shown in Figure 1b. The assumption underlying this model is that the temporal pattern can be aggregated better with RNNs then CNNs, while relying on CNNs on input side for local feature extraction. In CRNN, RNNs are used to aggregate the temporal patterns instead of, for instance, averaging the results from shorter segments as in [3] or convolution and sub-sampling as in other CNNs. In its CNN sub-structure, the sizes of convolutional layers and max-pooling layers are 33 and (2
× 2)- (3 × 3)-(4 × 4)-(4 × 4). This sub-
sampling results in a feature map size of N × 1 × 15 (number of feature maps x frequency x time). They are then fed into a 2-layer RNN, of which the last hidden state is connected to the output of the network.

Transfer Learning

We aim at learning from a source data distribution (multiclass tags) a well performing model on a different target data distribution (single class genres). Inside the transfer learning paradigm this is known as domain adaptation. The two most common practices and the ones that we will apply are: Using the network as feature extractor. That is removing the last fully-connected layer and

treat the network as a feature extractor. Once we have extracted all the features at the top we can include a classifier like SVM or a Softmax classifier for the new dataset. Fine- tuning the network. This strategy is based on not only replace the classifier layer of the network, but also retrain part or the whole network. Through backpropagation we can modify the weights of the pre-trained model to adapt the model to the new data distribution. Sometimes its preferable to keep the first layers of the network fixed to avoid overfitting, and only fine-tune the deeper part. This is motivated because the lower layers of the networks capture generic features, that are similar to many tasks while the higher layers contain features that are task and dataset oriented as demonstrated in [18].

Multiframe

We propose to use a multiframe strategy that allows us to extract more than one frame per song. For each song we discard the first and last N seconds and then, we divide the rest into frames of equal time-length t. The final parameters are stated in the experiments. This approach has two advantages:
At training time: we are able to generate more data to train the network than in the approach of [4], as they are only extracting the central part of the song. We are very limited by the number of songs, this is a very useful tool to provide data augmentation.
At test time: we can average or perform a KNN with the scores of every frame to infer the genre tag for the complete song with more confidence.

EXPERIMENTS

Model Evaluation Matrices

When evaluating a music genre recognition model built with deep neural networks, several metrics can be employed to assess its performance. Here are some common evaluation metrics:

Accuracy:

Accuracy measures the proportion of correctly classified instances out of the total instances in the dataset. While accuracy provides a general overview of the model's performance, it may not be sufficient if the dataset is imbalanced.

Precision:

Precision measures the ratio of correctly predicted positive observations to the total predicted +tive observations. In the context of music genre recognition, precision indicates how many of the predicted instances of a particular genre were actually correct.

Recall (Sensitivity):

Recall measures the ratio of correctly predicted positive observations to the all observations in the actual class. It indicates the model's ability to correctly identify instances of a particular genre from all instances of that genre in the dataset.

F1-score:

The F1-score is the harmonic mean of precision and recall. It provides a balanced measure of the model's performance by considering both precision and recall. F1- score is especially useful when dealing with imbalanced datasets.

Confusion Matrix:

A confusion matrix provides a tabular summary of the model's predictions versus the actual labels. It shows the number of true positives, false positives, true negatives, and false negatives for each class, allowing for a detailed analysis of the model's performance across different genres.

GTZAN dataset

GTZAN is a dataset created by Tzanetakis et al. It is compounded of 1000 music excerpts of 30 seconds duration with 100 examples in each of 10 different music genres: Blues, Classical, Country, Disco, Hip Hop, Jazz, Metal, Popular, Reggae, and Rock. The files were collected in 2000-2001 from a variety of sources including personal CDs, radio, microphone recordings, in order to represent a variety of recording conditions. And they are all 22050Hz Mono 16-bit audio files in
.wav format. However, regarding our approach it has one important limitation. The duration of each music excerpt (30s) makes that only one frame per song can be extracted and without discarding anything neither at the begging nor in the end of the song.
Therefore, although this dataset has been useful to compare the transfer learning performance with other works, it cannot be applied to evaluate the multiframe approach, as in this case more than one frame per song is needed.

Handmade dataset

In order to evaluate the performance of the multiframe approach, we built a dataset with the genres of the GTZAN dataset, but using longer songs. Specifically, our dataset is compounded by 300 music excerpts with 30 examples for each of 10 GTZAN genres. To be coherent, the genre-song association has been taking using Spotify lists [20]. As we use whole songs, the number of frames per song is variable, leading to 4 frames per song in the shortest songs and more than 10 in the opposite case (e.g. classical songs). Furthermore, to process this dataset we have trimmed both the begging and the end of each song. We have divided the dataset into train subset, 20 songs per genre or 1468 frames and test with 10 songs per genre or 747 frames.

Training

As we have stated before, we have fine- tuned two different networks, a CNN and a CRNN. We have made experiments by freezing the lowest layers and fine-tuning different top layers to see the differences. The parameters have been set as in standard fine- tuning, setting the learning rate a bit slower than in the original model. Batch normalization and dropout layers are implemented as the original authors did. To perform the training we use all of our handmade dataset as training data (2215 frames) and the GTZAN dataset as testing data (1000 frames).

Data Preprocessing Techniques

Data preprocessing plays a pivotal role in the success of Music Genre Recognition (MGR) systems based on Deep Neural Networks (DNNs). The quality and effectiveness of feature extraction and model training heavily depend on how well the raw audio data is processed and transformed. Here, we discuss some key
data preprocessing techniques tailored for MGR with DNNs:

Audio Data Loading:

Format Conversion: Convert audio files into a common format (e.g., WAV, MP3) for consistency.
Sampling Rate Standardization: Ensure all audio samples have the same sampling rate to facilitate processing.

Audio Feature Extraction:

Preprocessing of data is required before we finally train the data. We will try and focus on the last column that is 'label' and will encode it with the function LabelEncoder() of sklearn.preprocessing.
We can't have text in our data if we're going to run any kind of model on it. So before we can run a model, we need to make this data ready for the model. To convert this kind of categorical text data into model- understandable numerical data, we use the Label Encoder class.

Data Augmentation:

Time Stretching: Alter the duration of audio samples while preserving pitch to augment the dataset.
Pitch Shifting: Modify the pitch of audio samples to introduce variations and enhance model robustness.
Background Noise Injection: Introduce background noise to simulate real-world scenarios and improve model generalization.

Normalization and Standardization:

Mean Normalization: Subtract the mean of each feature dimension to center the data around zero.
Feature Scaling: Scale feature values to a specified range (e.g., [0, 1]) to ensure uniformity.

Data Splitting:

Training-Validation-Testing Split: Divide the dataset into training, validation, and testing sets for model training, tuning, and evaluation.
Stratified Sampling: Maintain class distribution proportions across splits to prevent bias.

Handling Class Imbalance:

Oversampling: Increase the number of samples in minority classes to balance class distribution.
Undersampling: Reduce the number of samples in majority classes to achieve balance.

Feature Concatenation:

Combine Multiple Features: Concatenate multiple feature representations (e.g., MFCCs, chroma) to capture diverse aspects of audio content.

Data Encoding:

One-Hot Encoding: Convert categorical genre labels into binary vectors to facilitate model training.
Label Encoding: Map genre labels to integer indices for model compatibility.

Handling Variable-Length Sequences:

Padding: Pad sequences with zeros to ensure uniform length, enabling batch processing.
Sequence Truncation: Truncate sequences to a fixed length for compatibility with model inputs.

Data Visualization:

Exploratory Data Analysis (EDA): Visualize audio features and class distributions to gain insights and identify patterns.

Methods & Algorithms

- **Convolutional Neural Networks (CNNs)**: The core deep learning architecture used for image classification. CNNs leverage convolutional layers with learnable filters to extract features from image data.
- **Transfer Learning**: This technique utilizes a pre-trained model as a starting point, leveraging its learned features for your specific classification task. Fine-tuning the final layers on your data allows the model to adapt to your problem.
- **Data Augmentation:** This method artificially expands your dataset by generating variations of existing images. It helps the model learn from a wider range of image features and improve generalization to unseen data.
- **Optimization Algorithms:** Techniques like Adam or SGD (Stochastic Gradient Descent) are used to update the model's weights during training, minimizing the loss function and improving classification accuracy.
- **Loss Functions:** Functions like categorical cross-entropy measure the difference between the predicted probabilities and the true labels, guiding the model towards better predictions during training.
- **Evaluation Metrics:** Accuracy, precision, recall, and F1-score are used to assess the model's performance on the validation set. These metrics provide insights into how well the model is classifying images.

CONCLUSION

Project Scope

We explore the application of CNN and CRNN for the task of music genre classification focusing in the case of a low computational and data budget. The results have shown that this kind of networks need large quantities of data to be trained from scratch. In the scenario of having a small dataset and a task to perform, transfer learning can be used to fine-tune models that have been trained on large datasets and for other different purposes. We have shown that our multiframe approach with an average stage improves the single-frame song model. In the experiments, a homemade dataset compounded by songs longer than our frame duration has been used. These songs belong to 10 different genres and the experiments have revealed that the average stage achieves better results in 9 of these 10 genres and a higher total accuracy. Therefore, using the average stage we are able to remove the non-representative frames dependence.

Future Scope

The future scope for Music Genre
Recognition with Deep Neural Networks (DNNs) is promising, with several avenues for further advancement and application. Here are some potential directions for future research and development in this domain:

- **Multimodal Fusion:** Integrating audio features with other modalities such as lyrics, album artwork, or user listening behavior could enhance genre recognition accuracy. Multimodal fusion techniques leveraging DNNs could exploit complementary information from different sources to improve classification performance.
- **Cross-domain Transfer Learning:** Applying transfer learning techniques from related domains such as speech recognition or natural language processing could boost the performance of music genre recognition models, especially in scenarios with limited labeled data.
- **Personalized Genre Recognition:** Tailoring genre recognition models to individual preferences and listening habits could lead to personalized music recommendation systems that adapt to users' evolving tastes and preferences over time.
- **Adversarial Robustness:** Ensuring the robustness of genre recognition models against adversarial attacks and audio manipulations is crucial for deploying them in real-world applications where security and reliability are paramount.
- **Multilingual and Cross-cultural Recognition:** Extending genre recognition models to recognize music from diverse cultural backgrounds and languages would make them more inclusive and applicable in global contexts.
- **Collaborative Filtering and Social Context:** Leveraging social context and collaborative filtering techniques could enrich genre recognition models by incorporating social interactions, user- generated content, and community preferences into the classification process.

REFERENCES :

1. Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10(5), 293-302.
2. Humphrey, E. J., Bello, J. P., & LeCun, Y. (2013). Feature learning and deep architectures: new directions for music informatics. Journal of Intelligent Information Systems, 41(3), 461-481.
3. Dieleman, S., & Schrauwen, B. (2014). End- to-end learning for music audio. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 6964-6968). IEEE.
4. Li, X., & Ogihara, M. (2003). A
5. comparative study on content-based music genre classification. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval (pp. 282-289).
6. Schlüter, J., & Böck, S. (2014). Improved music genre classification with convolutional neural networks. In Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR) (pp. 101-106).
7. Choi, K., Fazekas, G., Sandler, M., & Cho,
8. K. (2017). Convolutional recurrent neural
9. networks for music classification. arXiv preprint arXiv:1703.01719.
10. Kons, Z. A., & Sáry, G. (2018). Deep learning-based music genre classification. IEEE Access, 6, 44994-45004.
11. Law, E., & von Ahn, L. (2009). Human computation. Synthesis Lectures on Artificial Intelligence and Machine Learning, 3(3), 1-121.
12. Wang, Y., Khanna, N., & Jayant, C. (2015).
13. On the utility of convolutional neural networks for acoustic scene classification. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 559-563). IEEE.
14. McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2018). librosa: Audio and music signal analysis in python. In Proceedings of the 14th python in science conference (pp. 18- 25).