MUSICAL POLYPHONY ESTIMATION

Saarish Kareer
University of Rochester
skareer@ur.rochester.edu

Sattwik Basu
University of Rochester
sbasu5@ur.rochester.edu

ABSTRACT

Knowing the number of sources present in a mixture is useful for many computer audition problems such as polyphonic music transcription, source separation and speech enhancement. Most existing algorithms for these applications require the user to provide this number thereby limiting the possibility of complete automatization. In this paper, we explore a few probabilistic and machine learning approaches for an autonomous source number estimation. We then propose an implementation of a multi-class classification method using convolutional neural networks for musical polyphony estimation. In addition, we use these results to improve the performance of an instrument classifier based on the same dataset. Our final classification results for both the networks, prove that this method is a promising starting point for further advancements in unsupervised source counting and separation algorithms for music and speech.

Index terms: Level of polyphony, instrument recognition, Convolutional Neural Networks (CNNs), multi-class classification, multi-label classification, multi-pitch estimation

1. INTRODUCTION

1.1 Background

Source counting is a relatively under-researched topic in the field of music information retrieval. Most of the current source separation or source estimation algorithms today based on Non-negative Matrix Factorization (NMF), Hidden Markov Models (HMM) or even neural networks need a certain amount of pre-training on the number of sources in the mixture to be able to maintain the accuracy on test data. Unsupervised methods like K-Means or online NMFs also need the number of clusters or sources to be provided by the user. Other acoustics based methods rely on stringent anechoic conditions. If a system can intelligently identify this number and sound source, it would benefit the separation enormously by providing a good initialization and reducing the computational time. Not only separation, but real time transcription of polyphonic music or score following can be made easier once the number and type of sources are known beforehand. For speech processing, potential applications where source counting could be a useful pre-processing step are real-time denoising of speech signals using online NMF or multiple speaker estimation and identification.

On the downside, we found a lot of irregularities in terms of its applicability and implementation. According to Cheveigne in his chapter on multiple F_0 estimation[1], counting the number of sources in a mixture is difficult both computationally and perceptually. The reasons are that some signals are ambiguously perceived as single voices having a lower F_0 or the sum of harmonically related F_0s, thus algorithms set to choose as many or as few voices could lead to splitting partials of one source or combining multiple sources respectively. Real world sound sources tend to be aperiodic and inharmonic making this task more challenging. While many methods like thresholding to find the global weight of an F_0 candidate by Klapuri[1], transcription systems for detection of onsets and offsets of notes by Goto[1] or the cancellation filters by Cheveigne[1] have been proposed, we found the method by Wu[1] to use an HMM to model transitions between different number of voices particularly interesting. Also, exploring the timbral features of music and speech to form clustering algorithms for pitch estimates as proposed by Duan et al[2] would be a great approach for source number estimation, and has been described further as a stepping stone for our method.

With this motivation, we would like to find a novel approach to tackle the problem of polyphony estimation for music. To that end, we organize the paper in the following manner: describe the previous relevant work being done in the domain of multi-pitch estimation/streaming and CNNs for instrument classification — define and explain our problem statement and method online — conclude with our classification results and suggest possible improvements.

1.2 Related work

Previous work in multi pitch estimation involves iterative spectral subtraction [3] and probabilistic modeling of peak and non peak regions [4]. The final goal is to find the maximum likelihood of pitches in a given frame using spectral manipulation of the sources. As the second step for estimating the pitch trajectories once individual pitches have been defined, the method of multi pitch streaming of harmonic sound mixtures [2] has been adopted. This method, uses harmonic structures, MFCCs, & UDCs as timbral feature vectors for the estimated pitch trajectories, and then streams them into sources using a constrained clustering algorithm. Finally, in the predominant instrument recognition for polyphonic music [5], multi-label classification using CNNs is used to estimate the probability of multiple predominant instruments in 1 second long time frames. The IRMAS training dataset with 11 individual instruments and their combination is used with their mel-spectrograms as input to an AlexNet [6] and VGGNet [7] like architecture. The results are shown to outperform previous instrument recognition SVM models.

2. METHOD

2.1 Outline

With this background, we combine the advantages of both these methods to use a CNN to classify the level of polyphony in each audio frame which is the frame wise count of active notes, pitches or instruments. With a suitable dataset, training-labelling process and feature extraction method, we would be able to not only estimate the number of such sources in a frame, but also what instruments/pitches they correspond to. A diagrammatic representation of our multi class classification idea is provided below.

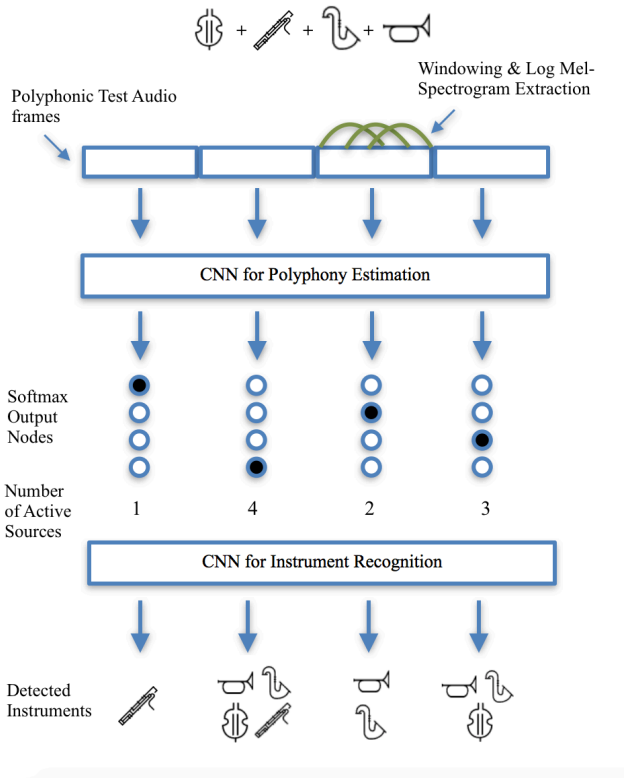


Figure 1. Pictorial representation of our problem statement

Problem statement - Given any polyphonic audio file, the system should be able to estimate the number of active instruments, pitches or notes per frame in real time. Using this information, an instrument classifier would be able to name the constituent instruments active per frame in real time. We now describe the proposed method outline, experiments and results.

2.2 Training Data

The Bach10 Dataset [4] from the Interactive Media Lab of the Northwestern University was used for our training and classification task. It is a polyphonic music dataset which can be used for various research problems like Multi-pitch Estimation and Tracking, Audio-score Alignment, or Source Separation. It consists of the audio recordings of each individual instrument and the ensemble of ten pieces of four-part J.S. Bach chorales, as well as their MIDI scores, the ground-truth alignment between the audio and the score, the ground-truth pitch values of each part and the ground-truth notes of each piece. The individual recordings of the four parts (Soprano, Alto, Tenor and Bass) of each piece are performed by violin, clarinet, saxophone and bassoon respectively.

The University of Iowa Music Instrument Samples (IOWA dataset) [8] contains 24 bit/44.1 KHz recordings of chromatic notes from the entire range of numerous woodwind, brass, string, and percussive instruments. The recordings are made with three Earthworks QTC-40 wide-frequency high resolution microphones arranged in a Decca Tree formation of left, center, and right placements 12 inches apart and 5 feet in front of the performer. The left and right mics were edited into single-note stereo 24/44.1 files archived into downloadable zipped folders which were used for this project.

2.3 Data pre-processing

2.3.1 CNN1 for Polyphony Estimation

Appropriate preprocessing of the dataset is important to obtain better classification accuracy using the Convolutional Neural Networks. From the Bach10 dataset, all possible combinations of instruments in solo, duet, trios and quartets for each of the ten chorales were created using Audacity. For instance, if we denote the four instruments in the Bach10 dataset as B, C, S, V, some of the combinations available are S, SV, BCS, BCSV and so on. Thus, given that there were 10 pieces and 4 instruments, we could create a total of 40 solo, 60 duets, 40 trios and 10 quartets. From the IOWA dataset, we created 33 tetrads, triads, dyad and single notes using non-repeating notes of the violin, bassoon, saxophone and clarinet in all combinations. For instance, a few combinations are violin C5 + saxophone E4 + bassoon G2 + clarinet B4, or violin G5 + clarinet E4 or just Saxophone D3. We made sure to cover the most commonly used note ranges for each instrument.

Finally, an equal amount of data per class and instrument combination was chosen from both the datasets. As a result, we were able to create our own dataset consisting of over 22 minutes of audio data in four different levels of polyphony (class).

These were further divided into segments of length 0.3 seconds (13230 samples) after silence-removal and then arranged as columns in a matrix. The length 0.3 seconds was chosen based on the rationale that this duration is sufficient for humans to perceive and identify different levels of polyphony and it also reduces computational time. The segmentation resulted in an audio data matrix of dimensions 13230 x 4344. In addition, a vector of dimension 4344 x 1 containing the true labels indicating the level of polyphony of each column was created. We decided to leave the sample rate to the original value of 44100 Hz as the audio files were completely noise free.

2.3.2 CNN2 for Instrument Recognition

To demonstrate the application of source counting in instrument recognition, we trained another CNN using a subset of the dataset created already. Here, we used all the 10 chorales played by single instruments from Bach10 and the 33 single notes we created using the IOWA dataset. Choosing an equal number of data for the 4 instruments — bassoon, clarinet, saxophone and violin, we were able to retrieve 18.06 minutes per instrument. We divided them again into 0.3 seconds (13230 samples) segments and labeled them as 0 for bassoon, 1 for clarinet, 2 for saxophone and 3 for violin, thus making this a multi-label classification task.

An input data matrix 13230 x 3612 and ground truth label vector of 1 x 3612 was created using these segments in a similar manner described earlier.

The python library librosa was used to convert each of the 0.3 second long audio segments into linear frequency spectrogram. To compute the Short-time Fourier Transform, the frame size and hop length were set to 1024 and 256 samples respectively. Next, the mel-scale and natural logarithm compression were applied to the linear frequency spectrogram to obtain the log-mel-spectrogram. We used 128 mel-frequency bins following the methods used by Han et al in [5]. This allowed us to strike a balance between preserving the harmonic character of the music while reducing the dimensionality of the data. Each audio segment resulted in a log-mel spectrogram of dimensions 128 x 52 which were then stacked together into a tensor of dimensions 4344 x 128 x 52 for CNN1 and 3612 x 128 x 52 for CNN2. These tensors were then used as the input to the Convolutional Neural Networks whose details are given below.

2.4 Architecture of the CNNs

We used the deep learning library Keras with TensorFlow backend to build our neural networks. The final architecture of the CNNs used in both the tasks were obtained after a fair amount of trial and error. We initially began testing out the popular AlexNet which consists of a very deep convolutional architecture with several layers of convolution followed by max-pooling. Specifically, this architecture consisted of three sets of double-convolution layers of size 3 x 3 with 32, 64 and 128 filters respectively followed by a 2 x 2 max-pooling layer.

A batch regularization layer was added with a regularization parameter of 0.01. However, due to the heavy architecture, the computation time came out to be 24 minutes for 20 epochs on a MacBook Pro with a 3.3GHz i5 core and 16GB RAM. The validation and testing accuracy obtained using this architecture were 59% and 64% respectively. These values were not good enough and we continued to experiment with other possible architectures.

Finally, through some more hyper-parameter tuning we arrived at an architecture that resulted in validation and testing accuracy of 76.81% and 81. 37% respectively, and took about 7 minutes to complete 20 epochs on the same machine. The specifics of this architecture are shown in the table below. We decided to use three single-convolution layers in our architecture as that reduced computation time and increased accuracy. The non-saturating activation function ReLU was used after each of the convolution layers as they produce much faster learning rates compared to saturating activation functions like the sigmoid or tanh. The final classification layer in the first CNN used a softmax function since we are looking for specific labels that denote the level of polyphony in a given piece of audio. For the instrument recognition task we used a sigmoid function classification to obtain a multi-label classification output. We also observed that max-pooling gave better results than average-pooling for this application.

Dimensions	CNN 1 and CNN 2 Architecture
4344 x 128 x 52 3612 x 128 x 52	CNN 1 input CNN 2 input
128 x 52 x 1	Log-mel spectrogram (frame size 1024, hop size 256)
126 x 50 x 32	3 x 3 Conv, 32 filters (ReLU)
63 x 25 x 32	2 x 2 max pool
61x 23 x 64	3 x 3 Conv, 64 filters (ReLU)
30 x 11 x 64	2 x 2 max pool
28 x 9 x 128	3 x 3 Conv, 128 filters (ReLU)
14 x 4 x 128	2 x 2 max pool
7168	Flatten
128	Dense (ReLU)
4	RMSprop optimizer Dense (softmax), categorical-crossentropy Dense (sigmoid), binary-crossentropy

Table 1. Architecture of the CNN model

We trained the first and second CNNs by optimizing the categorical-crossentropy and binary-crossentropy loss functions respectively, using the adaptive learning rate RMSprop optimizer. Dropout layers were not used as they were reducing the accuracy in both the training and testing stages. We also carried out tests using both linear amplitude mel-spectrograms and log-amplitude mel spectrograms as our input to the CNN. We observed that log-amplitude mel-spectrograms were the clear winners in this case as they took a short time to compute and produced higher accuracies.

2.5 Performance Evaluation

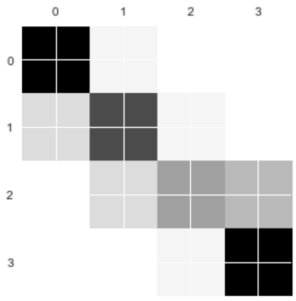
2.5.1 Performance of CNN1 for Polyphony Estimation

To test how well our algorithm performs on unknown data, we composed three new pieces of polyphonic music from our datasets. These pieces comprised of music of varying levels of polyphony like in the training data with each instrument playing only one note at one time.

We now describe the results obtained for one of the pieces, a mixed sequence of fourteen units. Each unit was either a solo note, a dyad, a triad or a tetrad. The prediction accuracy and the classification results are displayed below.

To gain a more intuitive understanding of how our CNN is performing we generated the confusion matrix. The matrix values and a heat map are shown below. The strong diagonal clearly shows that the system is performing as desired in this classification task.

Here, the labels 0, 1, 2 and 3 correspond to increasing levels of polyphony.



```
array([[13,  1,  0,  0],
       [ 3, 10,  1,  0],
       [ 0,  3,  6,  5],
       [ 0,  0,  1, 13]])
```

Figure 2: Classification results for CNN1

To demonstrate these results in real-time, we further segmented each unit into six sub-frames and used our model to predict the labels. The six predicted labels (red dots) and the ground truth (black solid line) per unit are shown below for four such units.

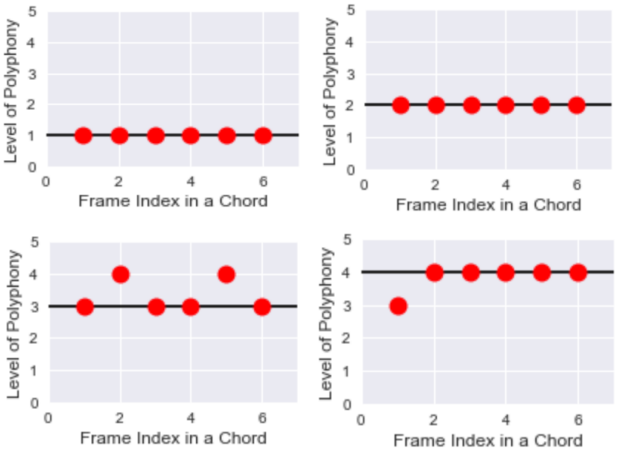
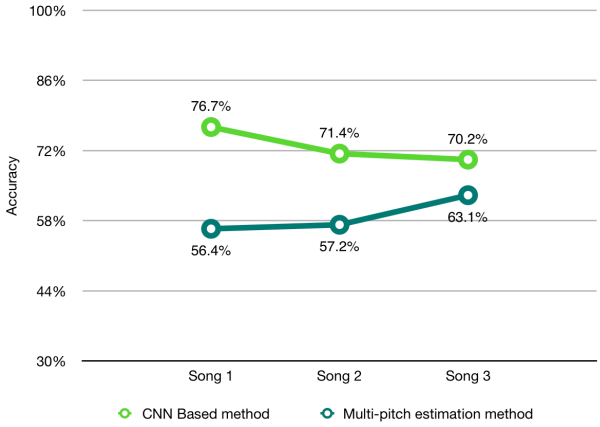


Figure 3: Real-time estimation of level of polyphony

Next, we made a comparative analysis of our polyphony estimation algorithm and the multi-pitch estimation method proposed by Duan et al [4]. The multi-pitch estimation algorithm produces the MIDI numbers of the pitches present per frame in a piece of polyphonic music. This gives us enough information to know the level of polyphony in per frame as well. By comparing the polyphony estimates against the ground truth values in the three pieces, we observed that our algorithm produced slight improvements over the multi-pitch estimation algorithm. The figure below illustrates the accuracies of both the methods in a comparative manner.

Figure 4: Comparison of polyphony estimation accuracy



2.5.2 Performance of CNN2 for Instrument Recognition

Similar validation and prediction metrics were used for the evaluation of the CNN2 for instrument recognition. However, as this was a multi-label classification, we used the sigmoid activation function with binary cross-entropy loss function with adaptive RMSprop optimization. First, the train-test split on the training data gave a strikingly consistent validation accuracy of 98.38% and testing accuracy of 95.65% due to the test data being monophonic played by a single instrument.

We then evaluated the model on mixture signals like those created in our testing set for CNN1 - 3 songs of varying polyphony. The accuracy dropped to 63.99%, proving that it couldn't predict the correct instruments even after tweaking different thresholds for the multi-label sigmoid layer [0.3 to 0.7].

We decided to use the results of CNN1, which essentially provided us with the number of active sources (instruments) in every frame of the testing songs to guide and improve the predictions of CNN2. For example, if in a testing frame containing just the bassoon, the CNN2 inaccurately predicted high probabilities for the violin and saxophone in addition to the bassoon, and the CNN1 predicted a level one polyphony for that frame, we used this information to choose the one highest probability amongst the bassoon, violin and saxophone.

In general, the n^{th} level of polyphony for each frame predicted by CNN1 was used to select the n best instrument probabilities predicted by CNN2. This helped the multi-label predictions get closer to the ground truth and the accuracy rose to 83.04%. The average accuracy, recall, precision and f1 score improvement using this technique is evident from the figure below.

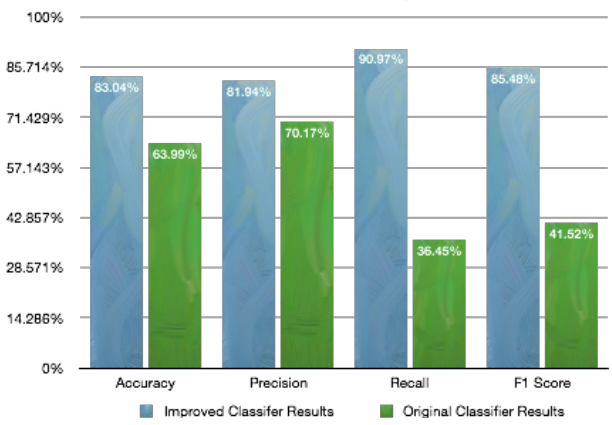


Figure 5: Improvements in the classification results of the CNN2

3. CONCLUSIONS

Here, we tackled the problem of polyphony estimation for instruments playing one note in a time frame like the bassoon or clarinet. The results are first compared to a multi-pitch estimation algorithm and then used to improve the performance of an instrument classifier.

Our next step would be to find augmentations to the model to cater to inherently polyphonic/polytimbral instruments like the piano, guitar or percussions. A combination of spectral features to first detect the polyphony and then timbral features to predict the instruments or pitches could be a future line of work. Once this is achieved, our CNN based polyphony estimation model can work in concert with probabilistic models like the multi-pitch streaming [2] algorithms for the entire audio file.

4. REFERENCES

1. Alain de Cheveigne; "CASA: Multiple F0 Estimation", 2006.

2. Zhiyao Duan, Member, IEEE, Jinyu Han, and Bryan Pardo, "Multi-pitch Streaming of Harmonic Sound Mixtures", *IEEE Trans. Audio Speech Language Processing.*

3. A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," *in Proc. Int. Conf. Music Inf. Retrieval (ISMIR)*, 2006, pp. 216–221.

4. Zhiyao Duan, Bryan Pardo and Changshui Zhang, "Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions," *IEEE Trans. Audio Speech Language Process*, vol. 18, no. 8, pp. 2121-2133, 2010.

5. Yoonchang Han, Jaehun Kim, and Kyogu Lee, "Deep convolutional neural networks for predominant instrument recognition in polyphonic music" *Journal of Latex Class Files, VOL 14, NO. 8, MAY 2016.*

6. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

7. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

8. Lawrence Fritts. University of iowa musical instrument samples. http://theremin.music. uiowa.edu/MIS.html, 1997. Online; accessed: 18-October-2017