EMITTER International Journal of Engineering Technology
Vol. 11, No. 1, June 2023, pp. 21~34
DOI : 10.24003/emitter.v11i1.764

Deep Learning Approaches for Automatic Drum Transcription

Zakiya Azizah Cahyaningtyas, Diana Purwitasari*, Chastine Fatichah

Department of Informatics, Institut Teknologi Sepuluh Nopember
Surabaya, East Java
*Corresponding author: diana@if.its.ac.id

Received February 8, 2023; Revised March 11, 2023; Accepted April 13, 2023

Abstract

Drum transcription is the task of transcribing audio or music into drum notation. Drum notation is helpful to help drummers as instruction in playing drums and could also be useful for students to learn about drum music theories. Unfortunately, transcribing music is not an easy task. A good transcription can usually be obtained only by an experienced musician. On the other side, musical notation is beneficial not only for professionals but also for amateus. This study develops an Automatic Drum Transcription (ADT) application using the segment and classify method with Deep Learning as the classification method. The segment and classify method is divided into two steps. First, the segmentation step achieved a score of 76.14% in macro F1 after doing a grid search to tune the parameters. Second, the spectrogram feature is extracted on the detected onsets as the input for the classification models. The models are evaluated using the multi-objective optimization (MOO) of macro F1 score and time consumption for prediction. The result shows that the LSTM model outperformed the other models with MOO scores of 77.42%, 86.97%, and 82.87% on MDB Drums, IDMT-SMT Drums, and combined datasets, respectively. The model is then used in the ADT application. The application is built using the FastAPI framework, which delivers the transcription result as a drum tab.

Keywords: Audio Classification, Automatic Drum Transcription, Deep Learning, Multi-Objective Optimization.

1. INTRODUCTION

Music transcription is an activity of writing the musical notation of a song. Music notation is a symbolic representation as an instruction to play a musical instrument in a song [1][2]. Music notation contains notes from the musical instruments being played. For example, on a piano, the musical notation will include the information of the keys played in a song. While on percussion instruments such as drums, musical notation contains information on the drum component (notes) and the time it is played (onset). The drum itself has many components and can be changed according to the player's convenience, a drum set. A drum set usually consists of a snare drum, kick drum, and hi-hat.

Drum music notation as instruction is essential in this area. In addition to performances, drum notation can be used as a means of education in the field of music. Thus, drum transcription to obtain it is also important. However, doing a good transcription cannot be done by just anyone. It takes years of experience to produce a good transcription in a short time [3]. Therefore, transcription automation will be beneficial in the field of music, especially in its education.

Automatic Drum Transcription (ADT) is an activity that focuses on the automatic transcription of drum instruments [4]. The purpose of ADT is to transcribe audio into drum notation. The methods used in ADT are generally divided into two: separate and detect and segment and classify. In the separate and detect method, the transcription model produces an activation function output for each drum component [5]. After that, the peaks are picked from the function to get the onset times. In contrast to the *separate and detect* method, the *segment and classify* method will look for the onset first and then classify the audio at that onset into one or more drum components. Although the approaches are different, both ways produce the same outputs, namely the onset time and the element of the drum being played.

In the *segment and classify* method, one of the essential steps is classifying the audio according to the onset. Audio classification can be done by recognizing the audio representation pattern for a particular drum component. In previous ADT studies that used the *segment and classify* method, audio classification was carried out using machine learning algorithms such as K-Nearest Neighbor (KNN) [6] and Support Vector Machine SVM [7]. On the other hand, audio classification can be done using deep learning (DL) and get good results [8][9]. Therefore, the use of DL seems possible to be used in the segment and classify ADT method.

Therefore, this study implements an automatic drum transcription using the *segment and classify* method with an addition of deep learning methods as the classifier. In selecting the model for classification, this study performs multi-objective optimization to obtain a classification model with good accuracy and optimize the use of time. In addition, this study also explores the audio spectrogram representation of several drum components to study the patterns represented in specific drum components. Finally, the result of the transcription is presented as a website application.

2. RELATED WORKS

ADT is a sub-section of Automatic Music Transcription (AMT) which is part of a broader topic about music, namely Music Information Retrieval (MIR) [10] [11]. The transcription of drum musical instruments is a part of Automatic Drum Transcription (ADT). Input in the case of ADT can be in the form of drum audio only (drums only) or mixed with other instruments. Meanwhile, the output of ADT is generally presented in the form of drum tabs.

In general, the information needed in ADT is the onset and the drum instrument played at that onset. Onset in music signifies the start of a note or playing of a musical instrument. As a percussion instrument, the onset in the drum indicates the time the instrument is struck. One of the drum transcriptions approaches works by generating each drum instrument's activation function. After that, the detection of onset using the peak-picking method was carried out from these results [12]-[16]. This approach is also known as *separate and detect*. This term is taken from how the method works, which separates the detection for each instrument as a function of activation, then picks the onset from the function result.

Another method is otherwise. The other method first segments the audio by detecting the onsets and then classifies the sound at that onset into a particular instrument (drum component) 7. First, onsets are obtained based on the audio signal strength of the drum using the peak-picking method [17]. Then, the classification step works by recognizing the pattern of the onset spectrogram pieces. The second method, the *segment and classify* method, is used in this study. In both methods, feature extraction is required to obtain audio representation. However, previous studies used machine learning algorithms to learn the drum pattern. While in this study, the classification step is done using deep learning methods, knowing its capability to perform an audio classification.

3. ORIGINALITY

This study implements the *segment and classify* method using deep learning as the classifier. In addition, this study will apply an evaluation method using multi-objective optimization for the classification model. Analytical research from Yao et al. shows a correlation between model performance and complexity—the more complex a model, the better the model's performance will be [18] — which ultimately affects execution time. Knowing this, this study evaluates the deep learning model not only on the quality of the prediction (macro F1) but also on the time spent on a prediction. So, it can provide the best experience in using the website.

The evaluation of the deep learning model is adjusted for multi-objective optimization with the objectives being (1) higher F1 macro values and (2) less prediction time (faster execution time). Multi-objective optimization is formally defined as Equation 1. Thus, $\theta*$ is the optimal model where w is the manually determined weight value, k is the number of objectives, and f is the evaluation score. Classification can be approached as a sequence or image input since spectrogram can be represented as mentioned. Hence, the architectures experimented on in this study are recursive-based models for the sequence approach and convolutional-based models for the image approach.

$$f(\theta^{*}) = optimum\left(\sum_{i=1}^{k} w_i f_i(\theta)\right) \tag{1}$$

4. SYSTEM DESIGN

This study used MBD Drums and IDMT-SMT Drums datasets. Both datasets are recorded as 16-bit mono audio with a 44.1 kHz sample rate. The annotations in both datasets contain the onset times information and the corresponding onset instrument. One onset can contain multiple instruments at the same time. Moreover, for the sake of simplicity, we are focusing only to three instruments which are snare drum, kick (or bass) drum, and hi-hat.

The transcription flow is as described in Figure 1. First, the audio is segmented by the onsets detected. Next, the segmented audio is classified using the spectrogram feature as the input. This study experimented with and evaluated several classifiers. Then, the classifiers are evaluated using the multi-objective optimization method based on their macro F1 and prediction time. Finally, the onsets and instrument labels information are yielded to be generated as a drum tab. Each step is described in the following sub-sections.

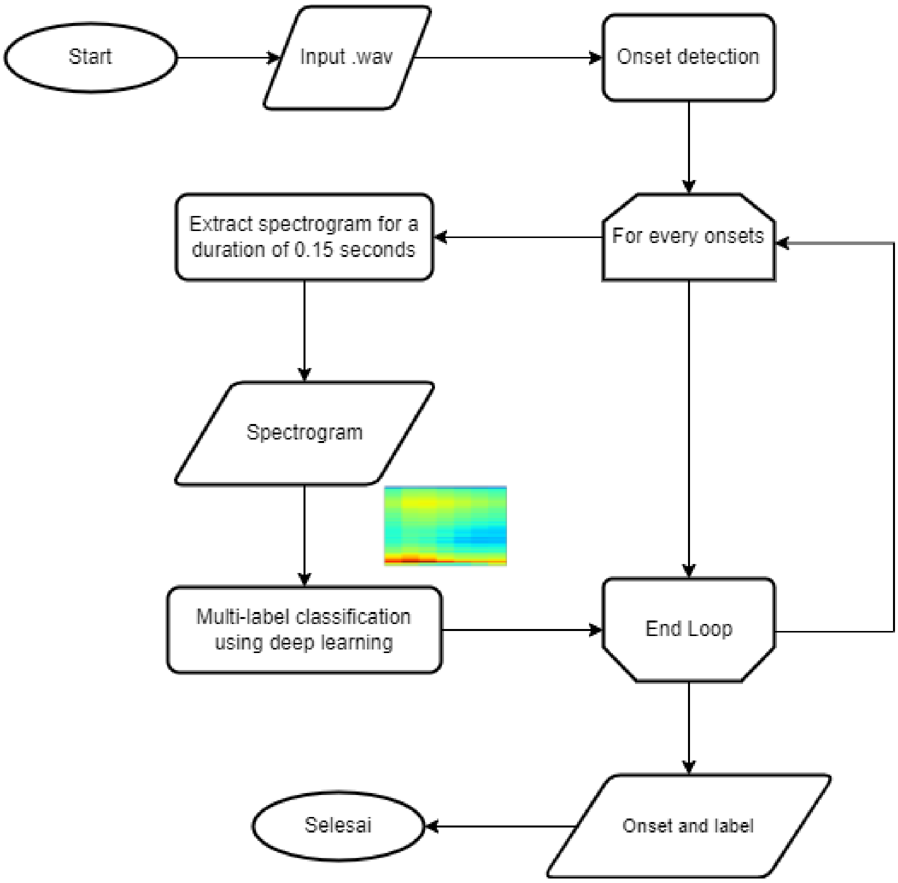

Start
Input .wav
Onset detection
Extract spectrogram for a duration of 0.15 seconds
For every onsets
Spectrogram
Multi-label classification using deep learning
End Loop
Selesai
Onset and label

Figure 1 . Work Flow of The Automatic Drum Transcription Process

4.1 Segmentation

Onset detection is performed using the *onset_detect* function from the Librosa library. This function implements the method introduced by Böck and Widmer [19] by calculating the value of the onset flux spectral. The results of the onset are used to select the peaks by peak-picking. The output of this function can be in the form of a sample index (frame) or onset time in seconds. An *n* sample of signal *x* is said to peak if it satisfies the following conditions:

1. The signal intensity value (x_t) is equal to the maximum value in a particular range defined by the *pre(max)* and *post(max)* of *t*.
2. The signal intensity value (x_t) is greater than or equal to the average value in the range defined by a specific *pre(avg)* and *post(avg)* of t added with a limit value or *delta* (α).
3. The distance between *t* and the previous peak is greater than the waiting time (*wait*).

This condition makes *pre(max)*, *post(max)*, *pre(avg)*, *post(avg)*, *α*, and *wait* parameters configurable. These parameters will be selected by performing a grid search on the MDB Drums dataset. In evaluating the onset detection, this study uses the F1 macro score using the *mir_eval* library. A detected onset is considered a true positive if it is near an onset in the ground truth. In contrast, a false positive happens if the detected onset occurs earlier than the nearest onset in the ground truth; a false negative otherwise. The experimental results will be discussed further in the later chapter.

4.2 Spectrogram Feature Extraction

Mel-spectrogram represents the audio in the time-frequency domain and is built on two concepts, the *mel*-scale, and the spectrogram. A *mel*-spectrogram is a spectrogram that is scaled using the *mel*-scale. The spectrogram is a quadratic transformation of the short-time Fourier transform (STFT). STFT is a sequence of Fourier transform of a windowed signal [20]. On the other hand, the *mel*-scale is a logarithmic scale to the frequency in a spectrogram. There are several definitions for this scale, but the popular one is the one introduced by O'Shaughnessy [21].

Spectrograms are taken from the onset time minus 0.025 seconds to avoid late onset detection with a duration of 0.15 seconds. The duration is chosen to get enough context but also avoid including audio of the subsequent onset. Spectrograms are taken at a sample rate of 44.1 kHz, a sample window of 2048, and a *mel* filter frequency from 20 to 20,000 Hz. This extraction produces a spectrogram with 128 frequency columns with a length of 13 frames for each detected onset. The feature extraction process makes use of the Librosa library.

4.3 Audio Classification using Deep Learning

The deep learning (DL) models perform multi-label classification on the extracted spectrogram. At the input layer, batch normalization takes place to accelerate and stabilize model learning [22].

The three possible labels of interest are snare drums (SD), kick drums (KD), and hi-hat (HH). In the case of multi labels, the model combines several binary classification models from each label. Thus, each model has an output layer in the form of a Dense layer with three neurons and sigmoid activation [23]. Each label (instrument) allows a value of zero to one, i.e., the probability of an instrument being present at the segmented audio. The model is trained with cross-entropy binary loss calculation and Adam's gradient optimization.

One of the things to consider in making a DL model is the number of hidden layers and neurons used. There is no definite rule in the selection of these two things. Jeff Heaton summarizes the results of the number of layers in the hidden layer as in Table 1 [24]. Several other studies also mentioned the ability of multiple hidden layers to learn more profound and abstract patterns [25] [26]. However, adding multiple hidden layers can also lead to overfitting [27]. Overfitting occurs when a model can learn training data very well but does not perform well when faced with data it has never seen. In other words, the model cannot study patterns in general. Previous studies on audio classification generally used two hidden layers in the recursive model [28] [29] and two to 4 hidden layers in the convolution model [30]-[32]. This study uses a model architecture with three to four hidden layers in recursive and convolution models to study the pattern a little deeper.

In addition to the number of layers, the number of neurons also needs to be determined. As with the number of layers, there is no definite rule regarding the number of neurons that should be used for a particular case. However, the rule of thumb is that the number of neurons should not exceed a specific number, as shown in Equation 2, with α being a degree of freedom between 5 and 10. N_s is the number of data in the training data, N_i is the number of neurons in the input layer, and No is the number of neurons in the output layer [33]. Hence, this study uses number of neurons between 33 to 67.

$$N_h = \frac{N_s}{\left(\alpha \times (N_i + N_o)\right)} \tag{2}$$

4.4 Multi-Objective Optimization

At the classification stage, evaluation is carried out on the DL model for classification. Each model will be trained for 50 epochs and use a batch size of 8. The data used is the IDMT-SMT dataset, MDB Drums, and a combination of the two (ALL). In each data scenario, the training, validation, and test data are divided by the proportions of 80%, 10%, and 10%, respectively. Table 2 shows the details of the amount of data. After being trained, each model will be evaluated using test data. Information on the accuracy, precision, recall, and macro F1 scores will be recorded, but only F1 macro is used as the optimization objectives.

In addition to the model's accuracy, the prediction time on the test data is also recorded as a second objective. Prediction time is evaluated based on the prediction time for one data point (onset) in milliseconds. The evaluation score in this study is defined in Equation 5 which is an adaptation of the multi-objective optimization defined in Equation 1. Through these calculations, this study still prioritizes the performance (macro F1) of the model but also give additional score to models with shorter prediction time. According to Equation 5, a model (θ) will get a score of 1 or 100% if it has a macro F1 Equation 3 of 100% and a maximum prediction time of 0.1 milliseconds Equation 4.

$$f_0(\theta) = F1_{macro}(\theta) \tag{3}$$

$$f_1(\theta) = f(x) = \begin{cases} log(0.1), & \theta_t < 0.1 \\ log(\theta_t), & \theta_t \geq 0.1 \end{cases} \tag{4}$$

$$f(\theta) = 0.8\, f_0(\theta) + (-0.2) f_1(\theta) \tag{5}$$

5. EXPERIMENT AND ANALYSIS

5.1 Experimental Scenario

This study uses clean drum audios from two public datasets, namely MDB Drums and IDMT-SMT Drum. Hence, this study has three data scenarios, each of which is its own dataset and a combination of both (ALL). Each scenario splits the dataset into train, validation, and testing data with the proportion of 80%, 10%, and 10%, respectively. Table 1 shows the details of the amount of data in each dataset split. In addition, onset detection uses the MDB Drums for parameter tuning, which achieved an F1 score of 76.14%.

Table 1. Distribution of Dataset Splits

Scenario	Training	Validation	Testing
MDB Drums	5384	673	673
IDMT-SMT Drums	6252	626	626
ALL	11636	1455	1455

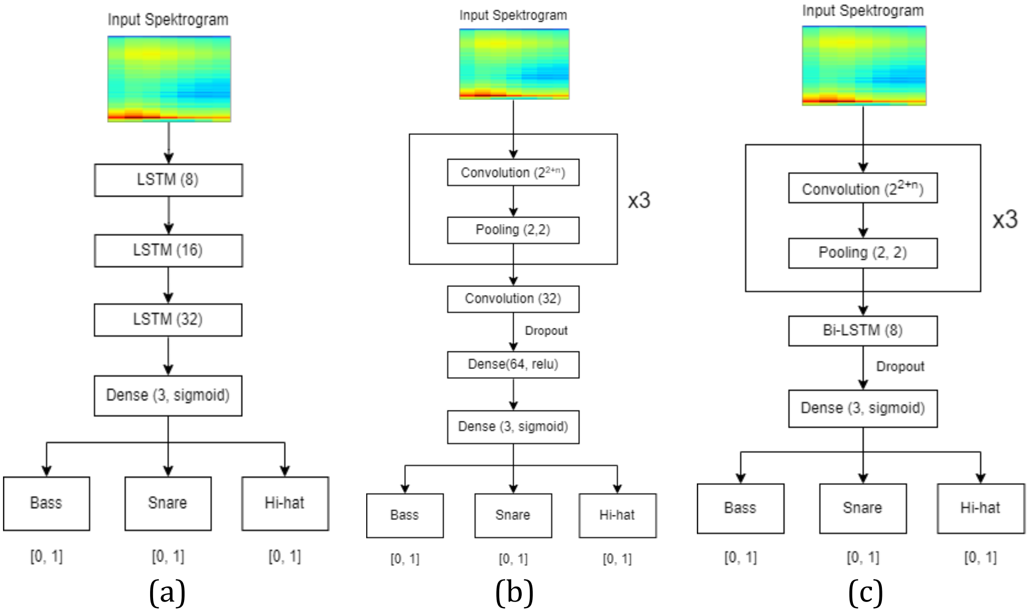
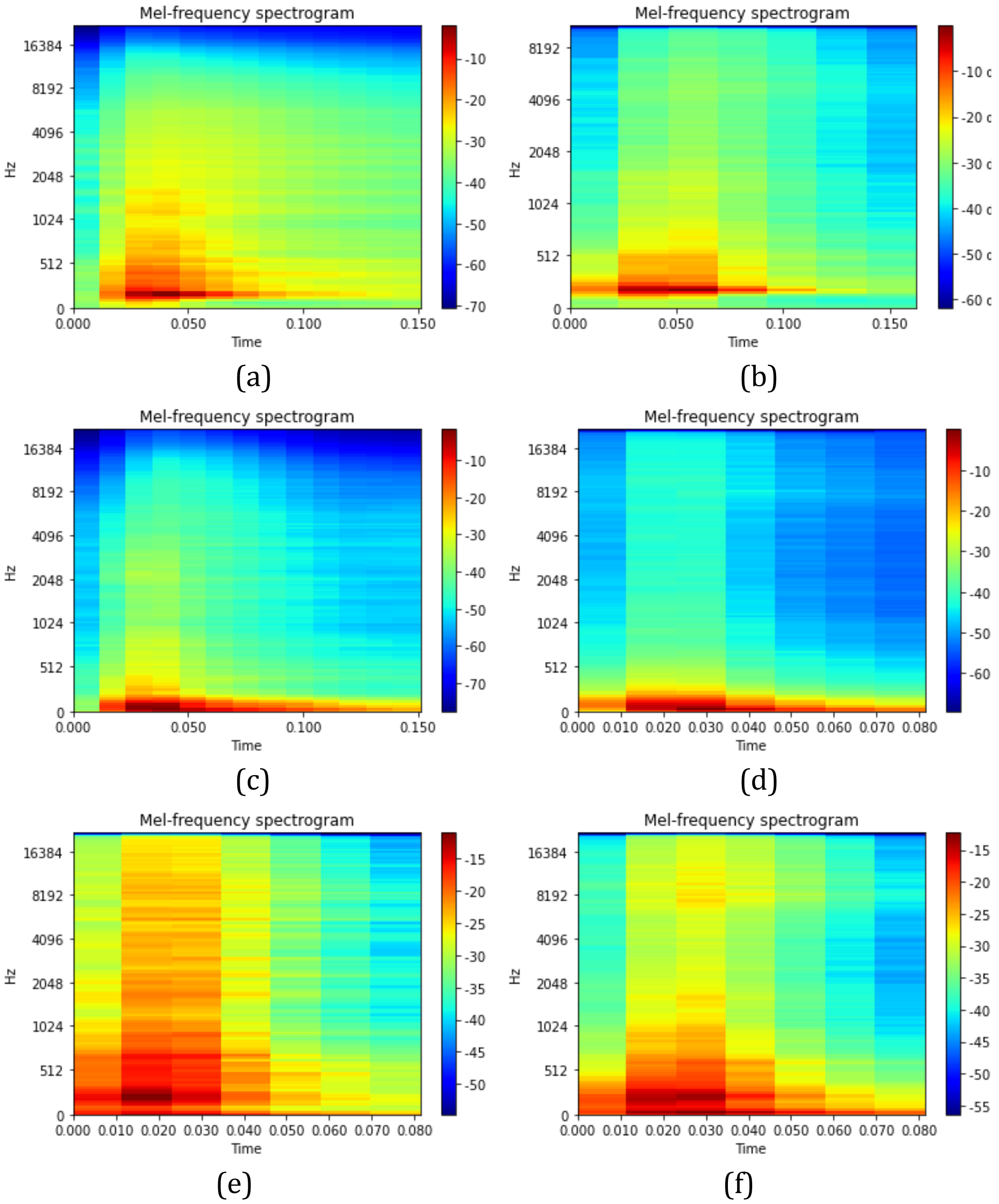


Figure 2. Architecture Illustration of (a) LSTM Model, (b) Convolutional Model, (c) CNN-BiLSTM Model

This study used LSTM, CNN, and a combination of both architectures to build the model. For LSTM ones, this study built a model with LSTM layers and another using Bi-LSTM layers. While on the CNN ones, the models built used

2-Dimensional and 1-Dimensional convolution layers. Last, a model combines the 2-Dimensional convolution and Bi-LSTM layers. Figure 2 illustrates some of the models built in this study.



(a) (b) (c) (d) (e) (f)

Figure 3. Visualization of Spectrogram's Median Value in Each Labels (SD, KD, HH from Top to Bottom) of MDB Drums (left) and IDMT-SMT Drums (Right)

5.2 Spectrogram Exploratory Data Analysis

Exploratory Data Analysis (EDA) or Exploratory Data Analysis is carried out on the spectrogram to analyze the patterns contained in each label. Figure 3 is a median visualization of the spectrogram of each label on each dataset. The median was chosen to avoid extreme values. However, the mean and median are visually not too different. This similarity can also indicate that the pattern distribution is close to the normal distribution because it has the same mean and median [34].

In spectrogram visualization, the horizontal axis denotes the timeline, and the vertical axis shows the magnitude of the frequency, with the higher axis indicating the higher frequency. The color of the heatmap on the spectrogram shows the intensity of a particular time and frequency, with red representing high intensity and blue representing low intensity.

From Figure 3, we can see that the median of each label in the two datasets has a pattern that is not much different. This similarity shows that each label has a similar pattern even though it comes from a different distribution. This finding reassures that each label has a learnable pattern. Hence, the datasets can also be combined to train the classification model.

Table 2. F1 Macro Results of Each Models

Scenario	Model	F1 Macro
MDB Drums	LSTM	**0.7981**
	Bi-LSTM	0.7862
	Conv2D	0.7751
	Conv1D	0.7684
	Conv2D + Bi-LSTM	0.7943
IDMT-SMT Drums	LSTM	0.9075
	Bi-LSTM	0.9079
	Conv2D	0.9078
	Conv1D	0.9112
	Conv2D + Bi-LSTM	**0.9209**
ALL	LSTM	0.8521
	Bi-LSTM	0.8497
	Conv2D	**0.8707**
	Conv1D	0.8328
	Conv2D + Bi-LSTM	0.8605

5.3 Multi-Objective Optimizations Results

The experiment was carried out with several dataset scenarios, namely using the MDB Drums, IDMT-SMT, and a mixture of both (ALL) datasets. In addition to the model's accuracy, this study also records the prediction time on testing data. Table 2 shows the results of the F1 macro from the test. No model outperformed all test data scenarios. However, the Conv2D + Bi-LSTM model seemed to outperform other models in the validation data. Interestingly, the Conv2D + Bi-LSTM model has a higher score on the validation data than the test data for the IDMT-SMT and ALL dataset scenarios.

This higher validation means that there is still room to improve the model's performance by increasing its complexity.

Table 3. Prediction Time Results of Each Models

Scenario	Model	Prediction Time (ms)
MDB Drums	LSTM	141
	Bi-LSTM	187
	Conv2D	573
	Conv1D	370
	Conv2D + Bi-LSTM	597
IDMT-SMT Drums	LSTM	240
	Bi-LSTM	331
	Conv2D	1060
	Conv1D	373
	Conv2D + Bi-LSTM	1140
ALL	LSTM	262
	Bi-LSTM	598
	Conv2D	1290
	Conv1D	480
	Conv2D + Bi-LSTM	1420

Table 4. Multi-Objective Optimizations Score of Each Models

Scenario	Model	*f(θ)*
MDB Drums	LSTM	**0.7742**
	Bi-LSTM	0.7402
	Conv2D	0.6340
	Conv1D	0.6667
	Conv2D + Bi-LSTM	0.6459
IDMT-SMT Drums	LSTM	**0.8697**
	Bi-LSTM	0.8415
	Conv2D	0.7406
	Conv1D	0.8341
	Conv2D + Bi-LSTM	0.7448
ALL	LSTM	**0.8287**
	Bi-LSTM	0.7589
	Conv2D	0.7070
	Conv1D	0.7625
	Conv2D + Bi-LSTM	0.6905

Based on the dataset scenarios, IDMT-SMT tends to give high scores on each model compared to other scenarios with F1 macros above 90%. The results are not significantly different, with a score of 78.44% ± 1.25% in the MDB Drums scenario, 91.11% ± 0.57% in the IDMT-SMT scenario, and 85.32% ± 1.40% in the ALL scenario. Thus, in selecting the model for the website, another objective is added, namely the prediction time of the model. This objective aims to provide a good user experience. The formula to calculate the multi-objective score is described in Chapter 4.4. Prediction time in each onset of test data defines the time objective. In other words, θ_t is the total prediction time of the test data divided by the number of data points in the test data. Table 3 shows the prediction time in milliseconds, and Table 4 shows the results of calculating the multi-objective optimization score of each model.

The results in Table 3 show that the multi-objective optimization score of the LSTM model outperformed all scenarios of the dataset. The superiority of LSTM occurs because of its high accuracy but can maintain low processing times. In contrast, the nature of the model itself can cause the CNN model's longer processing time. In the CNN model, the convolution kernel will run to every data in the image. This iteration can take some time when making predictions. In addition, although both types are built in three layers, the architecture on CNN needs to pass through the MaxPooling layer after convolution is applied. Thus, additional time is required to pass through the MaxPooling layer.

Previously, it was mentioned that the IDMT-SMT Drums dataset scenario yields a higher F1 macro score than other scenarios. The prediction for the HH label may influence this. The prediction of the selected model on each dataset scenario is shown to analyze this.

The confusion matrix was calculated using the one-vs-rest method. Thus, a confusion matrix is created for each label. For example, Figure X shows the confusion matrix of the HH labels in the selected model, where the Y axis represents ground truth, and the X axis represents predictions. In Figure X, we can see that the true positive in the test scenario of the IDMT-SMT Drums dataset has a significant number, as seen from the color. Meanwhile, in the scenario of other datasets and labels, more data falls on the false negative prediction results. As we already know, a higher number of true positives yields a higher F1 macro score.

6. CONCLUSION

The results showed that the segment and classify method can be used to create an Automatic Drum Transcription Application. The *segment and classify* method have two main stages: audio segmentation based on the detected onset and audio classification of the segmented audio. This study obtained a macro F1 score of 76.14% on onset detection through hyper-parameter search. In classification, this study showed that LSTM outperforms other models with a multi-objective optimization score of 77.42%, 86.97%, and 82.87% on MDB Drums, IDMT-SMT Drums, and combined datasets, respectively. In addition, the study showed a similar spectrogram pattern of each label in both datasets. This finding shows that the spectrogram is suitable for representing audio drums and that there are learnable patterns for classification.

Prediction-wise, we can see about 1% improvement of prediction in ALL datasets with an average F1-macro of 85.32% (± 1.40%) compared to the similar approach of *segment and classify* that had around 83.9% correct recognition. Furthermore, the model is still struggling to detect multiple instruments played at the same time.

On the other hand, timewise, instrument recognition of an onset takes approximately 240 milliseconds to detect an onset, using the chosen model. Assuming having at least two divisions per beat in a three-minute tracks, a 120bpm song will take approximately two to three minutes to complete transcription, with a possibility to miss multiple onsets.

Although not limited to real-life, multi-phonic soundtrack, the performance of this method ought to be lower than transcribing a clean drum soundtrack. This is possibly due to its struggle in detecting drum onsets (noised by other instruments) and its lack of capacity in detecting multiple drum instruments at the same time, especially in songs with high tempo.

The possible future development is mainly related to the performance of methods such as onset detection and BPM. For example, the resulting drum tab can also be better with a more precise onset and BPM. Alternatively, music theories can be used to learn the drumming pattern. For example, corrects the drum notes based on the predicted pattern. On the other hand, the selected classification model has shown a reasonably good performance. However, the classification model can still be improved in several ways, such as training using more diverse datasets. In addition, classification for other labels on drum instruments is also possible.

REFERENCES

[1] Ian D., B. **musical notation | Description, Systems, & Note Symbols |** Britannica.com. https://www.britannica.com/art/musical-notation (1998).

[2] Strayer, **H. From Neumes to Notes: The Evolution of Music Notation**. *Musical Offerings* **4**, 1–14 (2013).

[3] Hainsworth, S. W. & Macleod, M. D. **The Automated Music Transcription Problem**. 1–23 (2003).

[4] Wu, C. W. *et al.* **A Review of Automatic Drum Transcription**. *IEEE/ACM Transactions on Audio Speech and Language Processing* vol. 26 1457–1483 Preprint at https://doi.org/10.1109/TASLP.2018.2830113 (2018).

[5] Vogl, R. **Deep Learning Methods for Drum Transcription and Drum Pattern Generation**. (2018).

[6] Miron, M., Davies, M. E. P. & Gouyon, F. **An open-source drum transcription system for Pure Data and Max MSP**. in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* 221–225 (2013). doi:10.1109/ICASSP.2013.6637641.

[7] Gillet, O. & Richard, G. **Automatic transcription of drum loops**. in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* vol. 4 (2004).

[8] Blaszke, M. & Kostek, B. **Musical Instrument Identification Using Deep Learning Approach**. *Sensors* **22**, 3033 (2022).

[9] Haidar-Ahmad, L. **Music and Instrument Classification using Deep Learning Technics**. (2018).

[10] Benetos, E., Dixon, S., Duan, Z. & Ewert, S. **Automatic Music Transcription: An Overview**. *IEEE Signal Processing Magazine* vol. 36 20–30 Preprint at https://doi.org/10.1109/MSP.2018.2869928 (2019).

[11] Klapuri, **A. Introduction to music transcription**. *Signal Processing Methods for Music Transcription* 3–20 Preprint at https://doi.org/10.1007/0-387-32845-9_1 (2006).

[12] Dittmar, C. & Gärtner, **D. Real-time transcription and separation of drum recordings based on NMF decomposition**. in *DAFx 2014 - Proceedings of the 17th International Conference on Digital Audio Effects* 8 (2014).

[13] Southall, C., Stables, R. & Hockman, J. **Player vs transcriber: A game approach to data manipulation for automatic drum transcription**. in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018* 58–65 (2018).

[14] Southall, C., Stables, R. & Hockman, J. **Automatic drum transcription using bi-directional recurrent neural networks**. in *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016* 591–597 (2016).

[15] Vogl, R., Dorfer, M. & Knees, P. **Recurrent Neural Networks for Drum Transcription**. *Proc. of the International Society for Music Information Retrieval Conference (ISMIR)* 730–736 (2016).

[16] Vogl, R., Dorfer, M., Widmer, G. & Knees, P. **Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks**. in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017* 150–157 (2017).

[17] Bello, J. P. *et al.* **A tutorial on onset detection in music signals**. *IEEE Transactions on Speech and Audio Processing* **13**, 1035–1046 (2005).

[18] Yao, Y. *et al.* **Complexity vs. Performance: Empirical analysis of machine learning as a service**. in *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC* vol. Part F1319 384–397 (Association for Computing Machinery, 2017).

[19] Böck, S. & Widmer, G. **Maximum filter vibrato suppression for onset detection**. in *DAFx 2013 - 16th International Conference on Digital Audio Effects* (2013).

[20] Kehtarnavaz, N. **Frequency Domain Processing**. in *Digital Signal Processing System Design* 175–196 (Academic Press, 2008). doi:10.1016/b978-0-12-374490-6.00007-6.

[21] O'Shaughnessy, D. & Deng, L. **Speech Processing: A Dynamic and Optimization-Oriented Approach.** (2003).

[22] Santurkar, S., Tsipras, D., Ilyas, A. & Madry, **A. How does batch normalization help optimization? in *Advances*** *in Neural Information Processing Systems* vols. 2018-Decem 2483–2493 (2018).

[23] Huang, Y., Wang, W., Wang, L. & Tan, T. **Multi-task deep neural network for multi-label learning**. in *2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings* 2897–2900 (IEEE Computer Society, 2013). doi:10.1109/ICIP.2013.6738596.

[24] Heaton, J. **The Number of Hidden Layers**. *Introduction to Neural Networks for Java* 157-158 Preprint at (2008).

[25] Bengio, Y., Courville, A. & Vincent, P. **Representation learning: A review and new perspectives.** *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35**, 1798–1828 (2013).

[26] Sebt, M. v., Ghasemi, S. H. & Mehrkian, S. S. **Predicting the number of customer transactions using stacked LSTM recurrent neural networks.** *Social Network Analysis and Mining* **11**, 1–13 (2021).

[27] Sachdev, H. S. **Choosing number of Hidden Layers and number of hidden neurons in Neural Networks**. (2020).

[28] Scarpiniti, M., Comminiello, D., Uncini, A. & Lee, Y. C. **Deep recurrent neural networks for audio classification in construction sites**. in *European Signal Processing Conference* vols. 2021-Janua 810–814 (2021).

[29] Stowell, D., Giannoulis, D., Benetos, E., Lagrange, M. & Plumbley, M. D. **Detection and Classification of Acoustic Scenes and Events**. *IEEE Transactions on Multimedia* **17**, 1733–1746 (2015).

[30] Lee, J., Kim, T., Park, J. & Nam, J. **Raw Waveform-based Audio Classification Using Sample-level CNN Architectures**. (2017) doi:10.48550/arxiv.1712.00866.

[31] Maccagno, A. *et al.* **A CNN Approach for Audio Classification in Construction Sites.** in *Smart Innovation, Systems and Technologies* vol. 184 371–381 (Springer, 2021).

[32] Palanisamy, K., Singhania, D. & Yao, **A. Rethinking CNN Models for Audio Classification.** (2020) doi:10.48550/arxiv.2007.11154.

[33] FrontlineSolvers. **Training an Artificial Neural Network**. https://www.solver.com/training-artificial-neural-network-intro (2020).

[34] UtahDeptSociology. **The Normal Distribution** - Sociology 3112 - Department of Sociology - The University of utah. https://soc.utah.edu/sociology3112/normal-distribution.php (2022).