MUSICYOLO: A SIGHT-SINGING ONSET/OFFSET DETECTION FRAMEWORK BASED ON OBJECT DETECTION INSTEAD OF SPECTRUM FRAMES

Xianke Wang, Wei Xu, Weiming Yang, Wenqing Cheng

School of Electronic Information and Communications
Hubei Provincial Key Laboratory of Smart Internet Technology
Huazhong University of Science and Technology, Wuhan 430074, China
{M202072113, xuwei, M202072117, chengwq}@hust.edu.cn

ABSTRACT

In this paper, we propose MusicYOLO based on object detection to detect the onset and offset in singing for the first time. The onset of the vocal is not as stable and clear as that of musical instruments, which makes the frame-based onset/offset detection methods often not work well. Compared with the previous onset/offset detection methods, MusicYOLO detects the whole note object in the spectrogram image instead of transient frame features around onset/offset, improving the onset/offset detection performance significantly. The experiment results show that the MusicYOLO framework has obtained a 94.16% F1 score of onset detection and a 91.35% F1 score of offset detection on the ISMIR2014 dataset, which proves that MusicYOLO is the state-of-the-art onset/offset detection framework for singing situation.

Index Terms— onset detection, offset detection, singing voice, object detection

1. INTRODUCTION

The onset/offset detection of singing is to infer the beginning and end position of each note from the singing audio. Currently, most onset/offset detection models [1] [2] for singing adopt a two-step method to obtain (onset, offset). Firstly, the methods based on frame [3] [4] are used to obtain each frame's probability of onset and offset. Then, post-processing methods such as peak selection obtain onset/offset from frame probabilities. The frame-based approach [5] has achieved a 95% onset F1 value in the automatic piano transcription task. However, the onset F1 value only reaches about 80% in singing [2] and other related tasks, which is not as excellent as piano transcription.

There are two main problems with the frame-based approaches. The first one is that the onset of the vocal is less stable and clear than that of musical instruments. The spectrogram is strongly influenced by the way the note is sung. For example, vibrato, portamento and other singing techniques results in obvious fluctuation of spectrogram along the frequency axis and brings wrong detection to the frame-based methods. Secondly, the frame-based methods are disturbed by the noise. Since the frame-based onset/offset detection principle only analyses edge features, it is error-prone to obtain the onset/offset when the noise occurs.

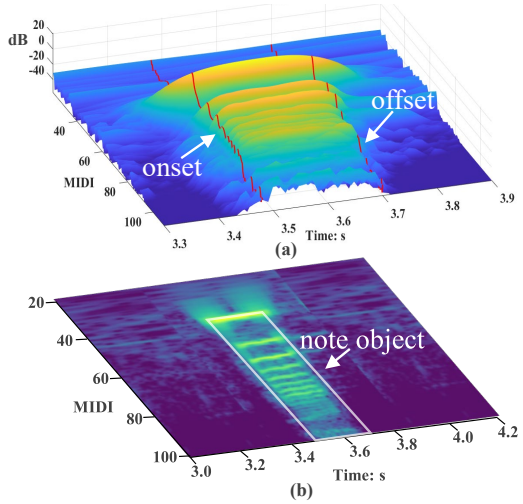

Fig. 1. (a) Previous frame-based onset/offset detection. (b) Object detection based onset/offset detection.

This paper proposes an object detection based method to realize onset/offset detection. Object detection has been applied to sound event detection. One types of methods use the two-stage detection method [6] [7] [8]. These methods selects the candidate region from the spectrogram, and then carries out the classification task. The other types use the single-stage detection method to locate and classify sound events at the same time [9] [10] [11]. This is the first time that object detection is used in the music research. As shown in Figure 1, compared with frame-based methods (Figure 1a), the object detection based method considers the whole note object's features in the spectrogram image instead of transient

This work is supported by The National Natural Science Foundation of China (No. 61877060). Wei Xu is the corresponding author.

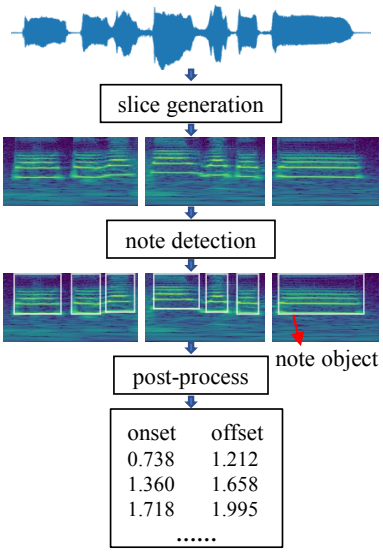

Fig. 2. The MusicYOLO framework.

frame features around onset/offset (Figure 1b). As a result, onset/offset can be observed from a more macro perspective, thus reducing the impact of pitch fluctuation and noise signals.

Our framework of onset/offset detection based on object detection is called MusicYOLO. We adopt YOLOX [12], the most advanced object detection model, to complete onset/offset detection. The audio signal is first converted into a CQT spectrogram matrix. Then a linear intensity mapping is used to convert the single-channel spectrogram matrix into a three-channel image. Afterwards we feed the spectrogram image into the YOLOX model to obtain the note bounding boxes. Finally, the left and right boundaries of the bounding boxes are converted to onset/offset.

We test different models on the ISMIR2014 dataset and our SSVD dataset. The results show that the MusicYOLO framework achieves 94.16% and 97.56% F1 values on the ISMIR2014 dataset and SSVD dataset, respectively, with onset_tolerance=100ms. Since only the ISMIR2014 dataset is annotated with offset, we only evaluate offset on this dataset. With offset_tolerance=100ms, the MusicYOLO framework achieves an F1 value of 91.35%. Overall, the MusicYOLO framework has achieved great improvement in onset detection and offset detection for singing compared with previous methods.

2. THE PROPOSED MODEL

As shown in Figure 2, the MusicYOLO framework is divided into slice generation, note detection, and post-process. Slice generation obtains image slices from the input spectrogram image. Notes detection uses YOLOX to detect notes from spectrogram slice and get the bounding boxes. Finally, onset and offset are obtained through post-processing.

2.1. Slice generation

The singing audio is converted into a two-dimensional matrix after the constant Q transform [13]. Then the spectrogram matrix is converted into a spectrogram image using a linear intensity mapping. Since the aspect ratio of the spectrogram image is too large to conveniently process, we use an automatic slice generation algorithm to obtain approximate square slices along the spectrogram time axis. We use $ratio$ to describe the slice shape, and its calculation formula is as follows:

$$ratio = w/h - 1 \tag{1}$$

where w and h represent the width and height of slice respectively, and their calculation formulas are as follows:

$$\begin{cases} w = frame\_len \cdot scale \\ h = n\_bins \cdot scale \end{cases} \tag{2}$$

where $frame\_len$ is the frame length of CQT matrix, n_bins is the frequency bins of CQT, and $scale$ is the scaling factor from matrix size to image size. $frame\_len$ is calculated as follows:

$$frame\_len = t \cdot sr/hop\_len \tag{3}$$

where sr and hop_len are the CQT parameters, and t is the time length corresponding to the slice. From the above, we get the $ratio$ calculation function written as R:

$$R(t) = t \cdot \frac{sr}{hop\_len \cdot n\_bins} \tag{4}$$

Our automatic slice generation algorithm is as follows:

1) Use $librosa.effects.split$ function [14] to split audio into unmute segments S, where S_i=($start_i$, end_i) represents the start time and end time of the ith unmute segment, i=1, 2, 3, ..., M;

2) Obtain segmentation points $splits$ of the spectrogram image. We divide the types of S_i into three categories according to $ratio$, which represent the appropriate segment time, longer or shorter time, extremely long or extremely short time, respectively:

$$\begin{cases} |ratio| \leq best\_ratio \\ best\_ratio < |ratio| \leq max\_ratio \\ |ratio| > max\_ratio \end{cases} \tag{5}$$

For long silent segments, we specially designed the silent segment cutting function C, which cuts the silent segment into squares. The details for obtaining segmentation points for different segment types is as Algorithm 1.

3) The spectrogram image is segmented into slices using the previously obtained segmentation points $splits$.

Algorithm 1 Obtaining segmentation points

initialize $\boldsymbol{splits}$;
set the iteration number $r = 1$ and the maximum iteration number M;
repeat
    $last\_time = \boldsymbol{splits}(-1)$;
    $ratio = R(end_i - last\_time)$;
    if $ratio \geq max\_ratio$ **then**
        $\boldsymbol{splits} = [\boldsymbol{splits},\ end_i]$;
    else if $best\_ratio < ratio \leq max\_ratio$ **then**
        $ratio\_silence = R(start_{i+1} - last\_time)$;
        $ratio\_next = R(end_{i+1} - start_{i+1})$;
        if $ratio\_silence > max\_ratio$ **then**
            $\boldsymbol{splits} = C(\boldsymbol{splits},\ S_{end\_i},\ start_{i+1})$;
        else if $ratio\_next > max\_ratio$ **then**
            $\boldsymbol{splits} = [\boldsymbol{splits},\ start_{i+1}]$;
        else
            $\boldsymbol{splits} = [\boldsymbol{splits},\ end_i]$;
    else
        A process similar to the above;
    $r \leftarrow r + 1$;
until $r > M$.

The algorithm parameters are set as follows: CQT parameters sr=44100Hz, $fmin$=27.5Hz, hop_len=512, $bins\_per\_octave$=24, n_bins=178. $librosa.effects.split$ function parameters top_db=20, $frame\_len$=1024, hop_len=512. And $best\_ratio$=0.2, max_ratio=0.65.

2.2. Note detection

Labelme software [15] is used to annotate the slices and construct the training set. The annotation method is shown in Figure 1b. We annotate the location of note objects in spectrogram images referring to the onset annotation of the audio. The upper boundary of each rectangular box represents the fundamental frequency, and the lower boundary represents the bottom edge of the image. The left boundary represents the onset. Due to the low time resolution of the CQT low-frequency bins, the energy boundary of the fundamental frequency cannot accurately represent the onset. Therefore, we use the original onset annotation combined with the second harmonic energy to determine the left boundary. The right boundary represents offset. At present, most datasets do not have offset annotation. So we adopt two principles to annotate the right boundary. If the onset of the current note is far away from the next onset, we observe the energy of the second and higher harmonics of the current note. When the energy attenuates to negligible, we defined that location as the right boundary; If it is difficult to distinguish the boundary between the current note and the next note, we use the left boundary of the next note as the right boundary of the current note. Then we send these annotated slices into the YOLOX network for training.

YOLOX [12] is a high performance and precision model for object detection. Compared with previous YOLO series models [16], YOLOX mainly has the following differences. The first is the decoupled detection head. Two detection heads are used for classification and bounding box regression, respectively, making the model converge faster. The second is to use an end-to-end way to complete the whole object detection without using NMS. Thirdly, the mode of anchor free is used, avoiding the steps of obtaining prior bounding boxes from datasets by clustering, reducing the potential of over-fitting. Finally, SimOTA enables the model to automatically analyze how many positive samples each ground truth should have and automatically determine which feature map each ground truth should be detected from.

We use the YOLOX-s pre-trained model [12] as our initial model parameters. The number of object categories is set to one. The input image size is 640x640. The data argumentation, learning rate and optimizer are the same as the original YOLOX model. The max epoch is 300. For inference, our confidence threshold is set to 0.6, and non-maximum suppression (NMS) is set to 0.45, maintaining the same configuration as the original YOLOX model.

2.3. Post-process

Singing onset and offset can be obtained through post-processing the bounding boxes of YOLOX. By stitching all slices together and adding corresponding offset to the coordinates of each slice, we can obtain the coordinates of each slice relative to the complete spectrogram image. We convert the left boundary of each box into onset by the linear proportional relationship. Similarly, we convert the right boundary into offset.

However, due to the differences between object detection and singing onset/offset detection, we post-process the output of YOLOX according to singing characteristics. First, we remove the box within another box. In object detection, the box-in-box phenomenon means that small objects are included in large objects. There is only one note object in each note range in the singing spectrogram because the human voice is monophonic. Therefore, if box A, whose upper left coordinate is (x_{a1}, y_{a1}) and lower right coordinate is (x_{a2}, y_{a2}), is contained by box B, whose upper left coordinate is (x_{b1}, y_{b1}) and lower right coordinate is (x_{b2}, y_{b2}), then we remove box A, box B and put a larger box C, whose upper left coordinate is $min(x_{a1}, x_{b1})$, $min(y_{a1}, y_{b2})$ and lower right coordinate is $max(x_{a2}, x_{b2})$, $max(y_{a2}, y_{b2})$, as the detection result. Secondly, since the characteristic of the human voice is that only the current voice ends and then the next voice can begin to sound, two adjacent bounding boxes should not intersect. If two adjacent boxes A, B intersect, where box A is in front of box B, then we use the right boundary of box A as the left boundary of box B.

3. EXPERIMENT AND RESULT

3.1. Dataset and Metrics

The MusicYOLO framework is trained and validated on our sight-singing dataset. Since we use YOLOX's pre-trained model for fine-tuning, only 35 annotated pieces of audio is used for training and validation. We use the ISMIR2014 dataset [17] and our SSVD dataset [1] as test set to compare MusicYOLO with other approaches. The ISMIR2014 dataset contains 38 singing songs, including 14 traditional childrens' songs, 13 untrained males' singing pop melodies, and 11 untrained females' singing pop melodies. The SSVD dataset is dedicated to the sight-singing scene and contains 127 recordings of sight-singing exercises for both children and adults.

We compare SiPTH [18], Tony [19], PDM [20], and our MusicYOLO using the library mir_eval [21] to calculate the precision, recall, and F1 values. Besides, we also evaluate MusicYOLO on a large singing transcription dataset, MIR-ST500 dataset [22] and a music instrument transcription dataset, Bach10 dataset [23]. Detailed experiment results and source code can be found on our GitHub page `https://github.com/itec-hust/MusicYOLO`.

3.2. Result and Analysis

In order to analyze the performance of each model under different time tolerance, we compare various models' onset/offset detection performance with time tolerance of 50ms and 100ms, respectively.

MusicYOLO performs stably in the above two datasets. Especially when onset_tolerance is set to 100ms, MusicYOLO achieves F1 values of 94.16% and 97.56% on the ISMIR2014 dataset and SSVD dataset, respectively. It achieves the best onset detection on the SSVD dataset because MusicYOLO uses sight-singing audio for training which has the same data distribution as the SSVD dataset.

Offset detection: Because only the ISMIR2014 dataset contains offset annotation, we only compare the offset detection performance on this dataset. As shown in Table 3, SiPTH, Tony and PDM methods don't performe very well on the ISMIR2014 dataset no matter whether offset_tolerance is set to 50ms or 100ms. With offset_tolerance =100ms, the best F1 value of all three is 80.31%. In contrast, whether offset_tolerance is set to 50ms or 100ms, the MusicYOLO framework reaches the F1 values of 85.21% and 91.35%, respectively, which greatly improves the offset detection performance.

Onset detection: It can be seen from Table 1 and Table 2 that the performance of SPiTH and PDM are poor on the ISMIR2014 dataset and SSVD dataset. Tony improves a lot in terms of onset detection performance compared with the previous two models. With onset_tolerance=100ms, F1 values of 80.09% and 88.67% are obtained on the ISMIR2014 dataset and SSVD dataset, respectively. However, the precision rate of Tony is much higher than the recall rate, which indicates that there are a lot of missing errors, and the model needs to be improved.

Generally, the offset detection methods perform not well due to no great energy change around offset like onset. This makes it tough to annotate offset and detect offset correctly. The offset detection of the MusicYOLO framework works well because the framework observes complete notes from a macro perspective instead of sticking to the characteristics around offset. Meanwhile, the MusicYOLO framework couples the detection of onset and offset. These led to a significant improvement in offset detection.

Table 1. Onset detection on the ISMIR2014 dataset

	50ms			100ms		
	P	R	F	P	R	F
SiPTH	61.05	65.02	62.52	74.08	78.36	75.56
TONY	72.51	62.91	67.10	86.88	74.87	80.09
PDM	71.27	61.34	65.36	79.84	86.88	75.03
Ours	91.05	89.27	**90.04**	95.16	93.41	**94.16**

Table 2. Onset detection on the SSVD dataset

	50ms			100ms		
	P	R	F	P	R	F
SiPTH	71.81	80.57	75.58	82.77	92.93	87.15
TONY	75.96	73.85	74.80	90.07	87.52	88.67
PDM	67.77	72.26	69.73	83.27	88.83	85.70
Ours	93.25	94.20	**93.68**	97.11	98.10	**97.56**

Table 3. Offset detection on the ISMIR2014 dataset

	50ms			100ms		
	P	R	F	P	R	F
SiPTH	33.29	35.71	34.22	50.68	50.68	51.62
TONY	81.89	70.46	75.42	87.21	75.01	80.31
PDM	78.95	68.07	72.43	87.02	75.02	79.85
Ours	86.05	84.58	**85.21**	92.31	90.62	**91.35**

4. CONCLUSION

This paper proposes the MusicYOLO framework based on object detection to complete singing onset/offset detection. The MusicYOLO framework uses the object detection model, YOLOX, to locate notes in the spectrogram. At the same time, considering the differences between music notes in singing and visual objects, we process the detection results of YOLOX, reducing some redundant bounding boxes and making the detection more accurate. The performance on the ISMIR2014 dataset shows that MusicYOLO improves onset/offset detection compared with previous approaches.

[1] https://github.com/itec-hust/Sight-Singing-Vocal-Data

5. REFERENCES

[1] J. Sebastian and H.A. Murthy, "Onset detection in composition items of carnatic music," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 560–567.

[2] S. Chang and K. Lee, "A pairwise approach to simultaneous onset/offset detection for singing voice using correntropy," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 629–633.

[3] S. Böck and G. Widmer, "Local group delay based vibrato and tremolo suppression for onset detection," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2013, pp. 361–366.

[4] L. Su and Y.H. Yang, "Power-scaled spectral flux and peak-valley group-delay methods for robust musical onset detection," in *ICMC*, 2014.

[5] C. Hawthorne, A. Stasyuk, and A. Roberts, "Enabling factorized piano music modeling and generation with the maestro dataset," in *Proceedings of International Conference on Learning Representations*, 2019.

[6] Y. Cao, Q. Kong, and T. Iqbal, "Polyphonic sound event detection and localization using a two-stage strategy," in *Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019.

[7] P. Pham, J. Li, and J. Szurley, "Eventness: Object detection on spectrograms for temporal localization of audio events," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 2491–2495.

[8] I.Y. Park and H.K. Kim, "Two-stage polyphonic sound event detection based on faster r-cnn-lstm with multi-token connectionist temporal classification," in *INTERSPEECH*, 2020, pp. 856–860.

[9] J. Yan, Y. Song, and W. Guo, "A region based attention method for weakly supervised sound event detection and classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 755–759.

[10] Y. Segal, T.S. Fuchs, and J. Keshet, "Speechyolo: Detection and localization of speech objects," in *arXiv preprint arXiv:1904.07704*, 2019.

[11] Y. He, N. Trigoni, and A. Markham, "Sounddet: Polyphonic moving sound event detection and localization from raw waveform," in *International Conference on Machine Learning (PMLR)*, 2021, pp. 4160–4170.

[12] Z. Ge, S. Liu, and F. Wang, "Yolox: Exceeding yolo series in 2021," in *arXiv preprint arXiv:2107.08430*, 2021.

[13] C. Schörkhuber and A. Klapuri, "Constant-q transform toolbox for music processing," in *Proceedings of Sound and Music Computing Conference*, 2010, pp. 3–64.

[14] B. McFee, C. Raffel, and D. Liang, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[15] W. Kentaro, "labelme: Image polygonal annotation with python," https://github.com/wkentaro/labelme, 2016.

[16] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," in *arXiv preprint arXiv:1804.02767*, 2018.

[17] E. Molina, A.M. Barbancho, and L.J. Tardon, "Evaluation framework for automatic singing transcription," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 567–572.

[18] E. Molina, L.J. Tardón, and A.M. Barbancho, "Sipth: Singing transcription based on hysteresis defined on the pitch-time curve," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 2, pp. 252–263, 2014.

[19] M. Mauch, C. Cannam, and R. Bittner, "Computer-aided melody note transcription using the tony software: Accuracy and efficiency," in *Proceedings of the First International Conference on Technologies for Music Notation and Representation*, 2015.

[20] L. Yang, A. Maezawa, and J.B. Smith, "Probabilistic transcription of sung melody using a pitch dynamic model," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 301–305.

[21] C. Raffel, B. McFee, and E.J. Humphrey, "mir_eval: A transparent implementation of common mir metrics," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 367–372.

[22] J. Y. Wang and J. S. R. Jang, "On the preparation and validation of a large-scale dataset of singing transcription," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 276–280, IEEE.

[23] Z. Duan, B. Pardo, and C. Zhang, "Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 2121–2133, 2010.