CFMT: A Music Transcription Model using Conformer Architecture

Yulong Wang[1], Hailong Yu[1], Fengchi Sun[1], and Jianyu Zhou[1*]

[1] College of Software, Nankai University, Tianjin, China
yulongwang@mail.nankai.edu.cn, jyzhou@nankai.edu.cn

Abstract. Automatic music transcription is a fundamental process for converting audio recordings of musical compositions into symbolic representations. This study extends the current state of the art in automatic music transcription with a specific focus on leveraging Conformer models, known for their exceptional performance in speech recognition. Furthermore, this research introduces an innovative approach designed to address the longstanding challenge of missing note-end events, which has previously hindered the accurate evaluation of Seq2Seq models using frame-wise metrics. Empirical findings reveal that a slightly modified Conformer model surpasses existing models across a spectrum of evaluation metrics, even outperforming models trained on distinct iterations of the MAESTRO dataset. Notably, this research contributes to the enhancement of frame-wise evaluation metrics for Seq2Seq models by providing estimations of possible note lengths for ongoing musical notes, resulting in a substantial improvement in evaluation accuracy.

Keywords: Automatic Music Transcription, Conformer, Note Events

1 Introduction

Automatic music transcription aims to generate music notes from given audio recordings, facilitating music information retrieval and the categorization of unclassified instruments. In practical applications, automatic music transcription addresses the needs of music learners by transcribing music notes and aiding in error identification. Additionally, it assists musicians in pre-generating MIDI files from audio signals, enhancing the creative process.

In the early stages of music transcription research, traditional machine learning techniques such as support vector machines and hidden Markov models were employed [1], resulting in binary piano roll transcriptions. However, with the success of deep learning in computer vision and natural language processing, researchers began incorporating deep learning into music transcription tasks. Nam *et al.* were among the first to apply deep learning, specifically recurrent neural networks [2], to automatic music transcription.

The development of deep learning techniques further advanced music transcription. Bock *et al.* achieved remarkable results by utilizing bidirectional LSTM net-works with two-resolution spectrograms as input and generating piano roll transcriptions as output [3]. In 2017, Hawthorne *et al.* significantly improved transcription accuracy by

employing stacked convolutional neural networks (CNN) to detect note onset events and generate piano roll transcriptions [4]. Kim *et al.* introduced reinforcement learning into music transcription, offering a generative model-based approach to the task [5]. Kong *et al.* introduced a custom model incorporating pedal information [6], marking the first use of pedal data to enhance music transcription accuracy. Wei *et al.* discussed the harmonic structure and pitch invariance in piano transcription and designed a custom model, HPPNet, based on this structure [7].

Recent models predominantly take spectrograms as input and output numerical representations in MIDI format [8,9,10,11]. This approach treats music transcription as a sequence-to-sequence (Seq2Seq) modeling task, aiming to improve accuracy and generalization. Seq2Seq models, initially applied to translation and speech recognition problems, were first applied to music transcription by Awiszus [12], who utilized both LSTM and Transformer models [13,14].

However, when evaluated using framewise metrics that rely on the existence of timestamps, these Seq2Seq models encountered significant challenges. Kwon *et al.* and Hawthorne *et al.* separately employed CRNN and T5 models for music transcription [15,4,8], using note events as evaluation metrics. Nevertheless, they did not address the issue of poor performance in frame-wise evaluation metrics when using Seq2Seq models. This limitation stemmed from the continuation of some notes into the next note due to the absence of note-end events, resulting in significant false positives.

To address the challenges in automatic music transcription, exploring the application of convolutional and attention mechanisms, which have shown exceptional performance in speech recognition, may help. Inspired by the similarities between music transcription and speech recognition tasks, it is worth determining if these combined mechanisms are equally effective in the domain of music transcription.

Therefore, this paper introduces a novel Seq2Seq model referred to as CFMT for automatic music transcription. CFMT harnesses the Conformer architecture [16], well-known for its superior performance in speech recognition. Simultaneously, it addresses the frame-wise performance limitations of the Seq2Seq models by introducing a straightforward post-processing technique that estimates possible note lengths for notes without explicit note-end events. Through extensive experimentation on the MAESTRO dataset, CFMT demonstrates significant improvements in accuracy across both frame-wise and note-event evaluation metrics, highlighting its potential applicability and efficacy in practical music transcription tasks.

To sum up,the contributions can be summarized as follows:

1. A Seq2Seq model, integrating both convolution and attention mechanisms, has been implemented, leveraging the strengths of convolution in capturing local details and attention mechanisms in extracting global information, thereby achieving state-of-the-art results in music transcription.

2. An exploration has been conducted on the various contributions of each module within the model towards music transcription.

3. The issue of missing note-ending events in the application of Seq2Seq models has been identified and discussed.

2 Method

2.1 Automatic Music Transcription

Formally, automatic music transcription involves taking a waveform, denoted as W (Fig. 1a), as input and generating a set of music notes, referred to as S (Fig. 1b), which collectively reconstruct this waveform. The output comprises a collection of n notes forming a set $S = \{s_1, s_2, \cdots, s_n\}$. Each individual note s_i adheres to the MIDI specification[17], encompassing four attributes: $onset(s_i)$ representing the note's initiation time, $offset(s_i)$ indicating the note's termination time, $pitch(s_i)$ denoting the pitch of the note, and $velocity(s_i)$ quantifying the intensity applied when the note was played. Consequently, the original waveform can be synthesized directly from the set of notes S through performance and superposition, mathematically expressed as $W = f(S)$.

Conversely, the music transcription presents an inverse challenge, seeking to discern the most probable set of notes from the given waveform, *i.e.*, $S = f^{-1}(W)$. This perspective treats the waveform as a composite of individual waveforms corresponding to each note, thereby illustrating the inherent complexity of music transcription.

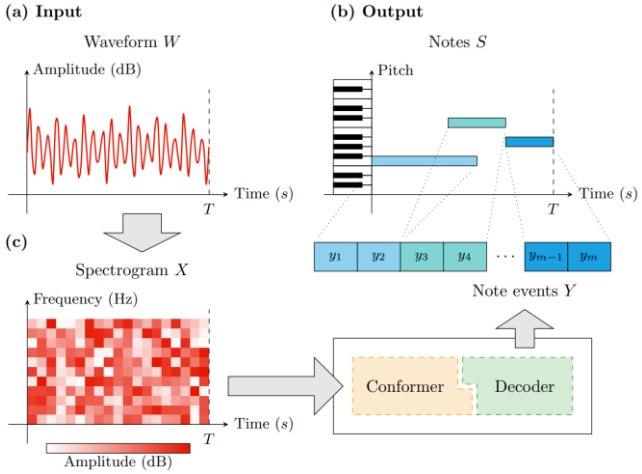

Fig. 1. An overview of CFMT.

In practice, within the realm of audio signal processing, the raw waveform typically undergoes preprocessing via a short-time Fourier transform [18] to yield the log frequency spectrogram (Fig. 1c). This log frequency spectrogram is mathematically represented as the matrix $X \in \mathbb{R}^{\frac{T}{\Delta t}} \times F$ where $N = \frac{T}{\Delta t}$ signifies the number of time frames resulting from the short-time Fourier transform, with T being the total length and Δt representing the time frame size, while F denotes the number of frequency frames generated during this transformation. Within the log-frequency spectrogram, each element x_{ij} denotes the signal amplitude or strength at the ith time frame in the jth frequency frame.

To tackle the challenge of separating individual musical notes from a composite waveform, CFMT leverages the Conformer architecture, which is trained using a known spectrogram X and the corresponding set of notes S to establish an accurate f^{-1} model. CFMT follows a two-step process: Initially, CFMT divides overly extended musical notes into smaller segments by selecting time windows of length $N\Delta t$ and subsequently generating the corresponding spectrogram matrix X using the short-time Fourier transform. Next, the original notes S are tokenized into note events, denoted as Y. With X as input and Y as output, the Conformer model is trained and can be employed to transcribe new waveforms in an autoregressive manner. An overview of CFMT is visually presented in Fig. 1.

2.2 Waveform Slicing and Note Tokenization

To account for the extended duration of the entire musical piece and to enhance the volume of training data, a slicing operation is performed within the temporal domain. This slicing procedure involves the following steps: Initially, a fixed time window of length $N\Delta t$ is established, and subsequently, the starting time t_0 is chosen randomly. Consequently, a time window with a range of $(t_0, t_0 + N\Delta t)$ is defined. Based on this defined time window, both the corresponding waveform and musical notes are segmented into a paired subset, which serves as the training data. Slicing the waveform is a straightforward process. The waveform contained within the time window is then subjected to a transformation via the short-time Fourier transform, resulting in the spectrogram denoted as $X^{(t0,t0+N\Delta t)} \in \mathbb{R}^{N\times F}$ (as it shown in Fig. 2a).

However, during the process of extracting musical notes, it is inevitable that notes may become fragmented. To solve this issue, our approach involves mapping a single note with four attributes into two distinct note events: a start event and an end event. Formally, a note event is denoted as y, and each note event encompasses three attributes: $time(y)$, $pitch(y)$, and $velocity(y)$. Both the start event and the end event share the same pitch as the corresponding note. The start event's time coincides with the note's onset, and its velocity reflects the initial velocity at which the note is played. Conversely, the end event's time corresponds to the note's offset, and its velocity is set to 0. By adopting this approach, a collection of n notes within S is transformed into a set containing $2n$ note events, effectively mapping the note collection S to a set of note events represented as Y. Building upon the definition of note events, the events located within the defined time window are subsequently extracted as labels for the corresponding spectrogram $X^{(t_0,t_0+N\Delta t)}$. This set of note events $Y^{(t_0,t_0+N\Delta t)}$ comprises the events y_i that satisfy the condition $t_0 < time(y_i) < t_0 + N\Delta t$.

The note events must be tokenized for training by discretizing y_i and flattening it into a vector of $(time(y_i), pitch(y_i), velocity(y_i))$. Subsequently, all the individual events are concatenated. At last, a predefined Start of Tokens (SoT) is added at the beginning, and a learnable End of Tokens (EoT) is appended at the end. The resulting vector has a length of m, with each element having a value within a predefined range as follows:

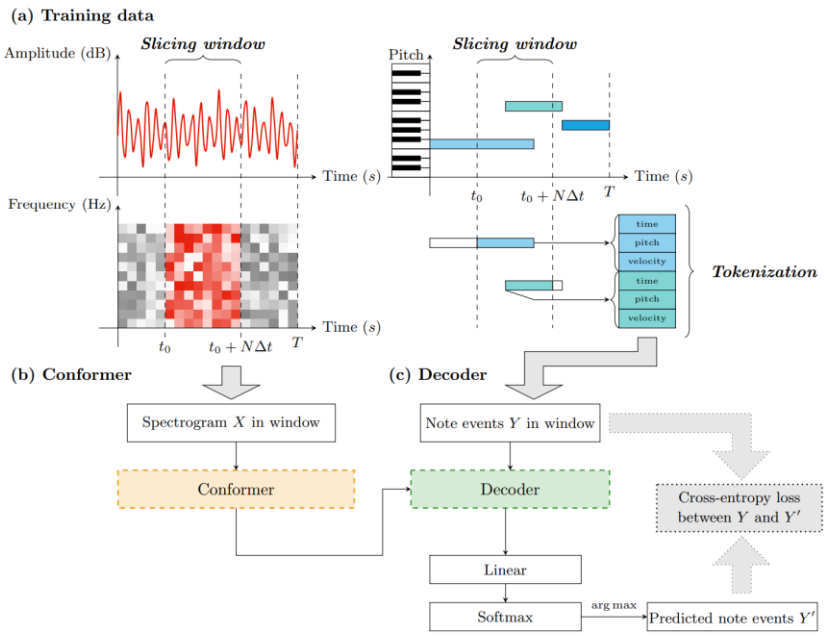

Fig. 2. The training process of CFMT.

- **Time**: The range is determined by the window size and is measured in units of 10ms. To ensure uniqueness, note events must be sorted in chronological order.
- **Pitch**: 128 values, representing different pitch of note events.
- **Velocity**: 128 values for the strength, indicating the state of the pitch for the current musical track until the next note event is encountered. Among these 128 values, 0 is used to denote the end of a note.
- SoT **(Start of Tokens)**: A single value representing the start of the token sequence.
- EoT **(End of Tokens)**: A single value indicating the end of the token sequence.

As a result, through randomly choosing the start time t_0, several training spectrograms $X^{(t_0, t_0+N\,\Delta t)}$ pairing with $Y^{(t_0, t_0+N\,\Delta t)}$ is obtained (as it shown in Fig. 2a)

2.3 CFMT Model and the Training Process

The training process takes the spectrogram matrix X from the sliced time window as input and utilizes the tokenized vector Y representing note events within the window as the target output. The spectrogram matrix X functions as the input to the Conformer model, serving as its encoder component (Fig. 3a). The encoder component comprises a series of L Conformer blocks, which incorporate both convolutional neural networks and self-attention mechanisms. Within each Conformer block, there exists an initial feed-forward neural network layer responsible for nonlinear transformations and feature extraction. The result from this layer is then scaled by a factor of $1/2$ and added to

the input through a residual connection. The output subsequently undergoes a self-attention layer, enabling the model to capture global information and dependencies within the input sequence. Following this, the result is further processed through convolutional layers to capture local features within the sequence. Depth-wise separable convolution layers are employed to handle various levels of features, thereby accelerating training while reducing computational complexity. Finally, another feed-forward neural network layer with a residual connection, in conjunction with layer normalization for training stability, constitutes the Conformer block. This entire process is layered, with each subsequent layer taking the output from the previous one and progressively extracting and transforming features from the input sequence.

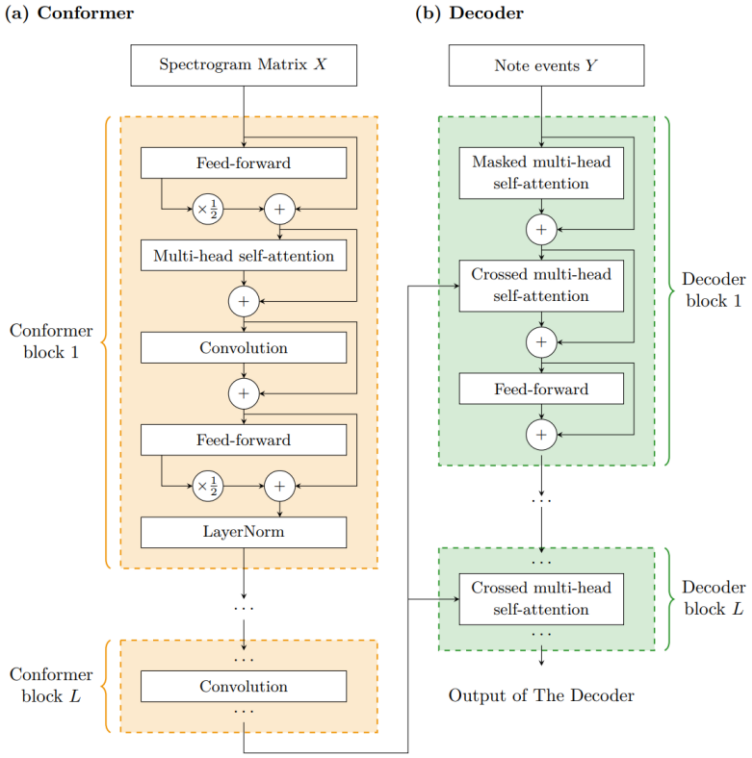

Fig. 3. The architecture of CFMT model.

The decoder component is an attention-based decoder (Fig. 3b). The target output Y is initially embedded, and the Transformer's positional encoding information, as described in Vaswani *et al.*'s work [19], is used as input to the decoder. Much like the Conformer, the decoder is comprised of L stacked decoder blocks. Each block commences with a masked multi-head self-attention mechanism, ensuring that the model can only predict the ith element based on the preceding $i-1$ elements while not having access to subsequent elements. The output of this step is then directed through a cross-attention layer with multiple heads. This layer integrates information from both the Conformer's output and the information pertaining to the previous $i-1$ elements.

Finally, the output passes through a feed-forward layer, facilitating non-linear transformation and feature extraction. Each layer builds upon the output of the previous layer, and ultimately, the decoder produces its final output.

Following the Conformer and decoder, the output dimension is reduced through a linear layer. Subsequently, a SoftMax operation is applied to compute the probabilities of the values associated with individual elements. By selecting the value with the highest probability, a predicted vector of note events within the time window is generated, denoted as $Y' = \{y'_1, y'_2, \cdots, y'_m\}$. The training process aims to optimize the weights of both the Conformer and decoder components to minimize the disparity between the original events Y and the predicted events Y'. This disparity is quantified using cross-entropy, making the loss function for training CFMT as $\mathcal{L}(Y, Y') = -\sum_{i=1}^{m} y_i \log(y'_i)$. The objective is to minimize this cross-entropy loss, thereby training CFMT to generate accurate predictions of note events from the input spectrogram.

2.4 Note Event Prediction and Detokenization

With the trained Conformer model, predictions can be made based on input waveform data. Since CFMT is trained with a time window size of $N\Delta t$, the prediction is also conducted in a segmented manner. As illustrated in (Fig. 4), the prediction begins with the initial time window $(0, N\Delta t)$, where the spectrogram $X^{(0,N\Delta t)}$ is fed into the Conformer. For the decoder component, a greedy autoregressive encoding strategy is employed. Starting with a predefined start-of-tokens $y^{(0)} = SoT$, the output is sequentially generated by continuously sliding the time window, producing a sequence like $(Y^{(0)}, Y^{(0:N\Delta t)}, Y^{(0:2N\Delta t)}, \dots)$, eventually resulting in the predicted note events Y.

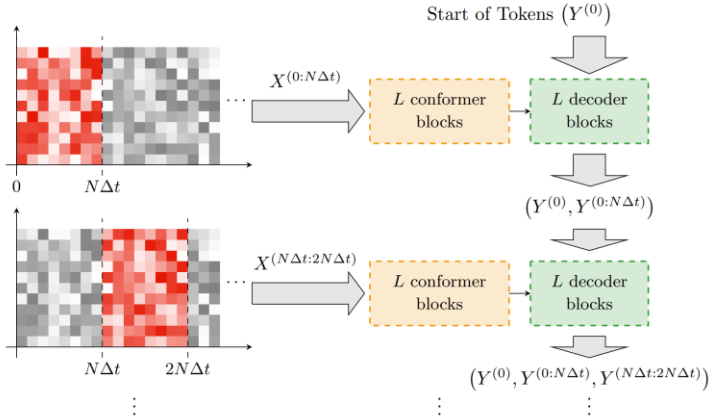

Fig. 4. The autoregressive way for predicting in CFMT.

In the final step, the predicted note events need to be detokenized and converted into musical notes. The detokenization process is straightforward. The conversion from the three-dimensional note events to four-dimensional notes that adhere to the MIDI specification follows this procedure: The note events in Y are iterated through in chrono-

logical order. If the current note event y_i has a non-zero velocity, the state of the corresponding pitch class is set to 'on'. and the $time(y_i)$ value (representing $onset(s)$ and velocity $velocity(y_i)$ are recorded. If the current note event y_i has a velocity of 0, the state of the corresponding pitch class is set of 'off', and the $time(y_i)$ value (indicating $offset(s)$, pitch $pitch(s) = pitch(y_i)$, and velocity $veloicty(s) = velocity(y_i)$.

2.5 Modeling Other Instruments

In the MIDI specification [20], the electronic musical score storage format remains consistent across all instrument types. This uniformity allows for the direct replacement of datasets and the application of identical processing methods, regardless of the instrument's nature. MIDI, as a standardized music file format, records musical information uniformly, encompassing notes, pitch, and volume. The interoperability of this standardized format streamlines the handling and exchange of musical data across diverse instruments, enabling efficient and flexible processing without the need for instrument-specific adaptations or conversions.

3 Results

3.1 Datasets and Parameter Setting

To evaluate the model's performance, the MAESTRO dataset, as described in the work by Hawthorne *et al.* (2018), was utilized. This dataset consists of piano performances by numerous pianists, meticulously annotated with timing accuracy within 3ms.

The dataset is divided into test, training, and validation sets. It's important to note that different models employ different versions of the MAESTRO dataset. Like all other models, we do not apply any noise processing to the dataset. In our research, the model was trained using the MAESTRO V3 dataset, and comparisons were conducted using datasets specific to different models.

During training, our model was trained on the MAESTRO training set and evaluated on the validation set. The final model selection was based on the epoch that yielded the lowest cross-entropy loss on the validation set. Like prior studies, sustain pedal effects were accounted for by extending notes when the sustain pedal was depressed until the pedal event concluded.

The model was trained using an RTX 3090 GPU with a batch size of 6, which was the maximum batch size that could be used for stable training. The training process spanned 15 days and included a total of 800 epochs, resulting in over 3 million training steps. This trained model was used as the baseline for the experiments. The model selected for this experiment is a small Conformer model with 4 heads and 16 layers. The only modification made was to the embedding dimension, which was set to 512, while the other parameters remained consistent with the small Conformer model. Additionally, a corresponding decoder was used. This choice was influenced by the fact that Conformer models have previously been employed with similar parameters when dealing with audio signals from the LibriSpeech dataset, which also contained over 100 hours of audio data.

The optimizer used was AdamW, with parameters set as follows: β_1=0.9, $\beta_2 = 0.98$, and $\varepsilon = 10^{-9}$. Learning rate scheduling was performed using the Transformer's learning rate adjustment mechanism, with a warm-up phase that spanned 1500 warm-up steps and a peak learning rate of 0.0005. A dropout rate of 0.1 was used, although no regularization was applied since the model did not exhibit overfitting. While smaller batch sizes may yield better results in some cases, it was observed that increasing the batch size beyond 6 did not significantly improve performance. Therefore, a batch size of 6, which was the largest batch size for stable training, was used.

For the spectrogram, the original sampling rate of 16000kHz was employed, with a frame length of 2048 and a hop width of 128. The input size was set to 512 Mel bins. The input length was restricted to 512 tokens, with padding added as needed when random cropping resulted in a length less than 512. Theoretically, there was no limit to the length of the output sequence, extending until the end of tokens was reached. In practice, the maximum token length was around 1000. As a result, each slice had a length of 4088ms, meaning that the maximum token value representing a time offset was 409(rounded up for results). The output sequence length was not constrained, as tokens could be of unlimited length. Although limiting the output length would have been more memory-efficient, it could have affected accuracy, so no such limit was imposed.

The model and optimizer parameters are detailed in Table 1 The model was implemented following the Conformer architecture as described in [16]. Pre-processing and evaluation were carried out using *seqio* and *note_seq*. The code is available at: *https://github.com/ylwang0425/CFMT*. These parameters were determined to yield the best results among all the different parameter settings used in the experiments.

The F1 score was used to evaluate performance, calculated as the harmonic mean of precision and recall. Precision and recall were calculated based on the pitch, start time, and end time of the notes. A bipartite graph maximum matching algorithm, as described in [23], was employed to match notes within an error tolerance. Then, the evaluation metrics were computed based on onset, or onset and offset, to calculate the F1 score.

Table 1. Parameters for training CFMT

Parameter	Value	Parameter	Value
Embedding Dimension	512	Max Learning Rate	0.0005
Feed-forward Neural Network Dimension	1024	Attention Heads	4
Decoder Attention Layers	16	Batch Size	6
Encoder Attention Layers	16	Training Epochs	400
Activation Function	Swish [21]	Input Dimension	512
Drop Out	0.1	ε	0.9
Optimizer	AdamW [22]	β_1	0.9
Max Token Length	1024	β_2	0.98

The *mir_eval* package was used for precise evaluation of the transcription model [24]. As the piano is a percussive instrument, the spectrogram exhibits significant oscillations when keys are struck. Consequently, the default tolerances provided by the

mir_eval package were employed. Specifically, for notes of the same pitch, velocity differences within 10% of the whole velocity range, an onset error within 50ms, and an offset error within either 20% of the note duration or 50ms were allowed as evaluation criteria.

To compare the performance of CFMT with other models, the evaluation scores were computed, including the Onset F1, Onset & Offset F1, and Onset, Offset & Velocity F1 scores. For frame-wise evaluation, each note was assigned the most likely note length. For direct model comparison, CFMT was evaluated on all three datasets within MAESTRO, and the results were compared to models trained and evaluated on each individual dataset. Our model achieved competitive results compared to all sequence-to-sequence models, demonstrating its effectiveness across a variety of models and datasets. The experimental results are presented in Table 2.

Table 2. Results of experiments on different version of MASESTRO.

Dataset	Model	Onset F1	Onset & Offset F1	Onset, Offset & Velocity F1
MAESTRO V1	Hawthorne *et al.* (2019)	95.53	80.50	77.54
	Hawthorne *et al.* (2021)	95.59	83.46	82.18
	Kwon (2020)	94.67	79.36	-
	Kim & Bello (2019)	95.60	81.30	80.20
	CFMT (ours)	**96.69**	**84.46**	**85.53**
MAESTRO V2	Kong *et al.* (2021)	96.72	82.47	80.92
	Xiao *et al.* (2023) [25]	**96.88**	83.18	-
	CFMT (ours)	96.01	**84.23**	**83.21**
MAESTRO V3	Hawthorne *et al.* (2021)	96.01	83.94	82.75
	Wei *et al.* (2022)	**97.18**	82.76	82.24
	CFMT (ours)	96.26	**84.45**	**83.44**

From the experimental results, it is evident that the Conformer model outperforms other models in terms of note-event evaluation metrics across all datasets. The Conformer model consistently achieves the best performance when compared to models trained and evaluated on different dataset versions. Conformer, with its improved performance characteristics from the field of speech recognition, offers significant advantages for music transcription.

3.2 Conformer Model Distillation

To validate the contributions of individual modules to the improvement of the Conformer model over Transformer-based models, ablation experiments were conducted on the three main modules of the Conformer: Attention, CNN, and Macaron FFN+RPE (relative position encoding). We compared the performance of the Conformer with ablation experiments on the Transformer with Macaron FFN+ relative position encoding, the Transformer (implemented by us), the Transformer (implemented by [9]), and the CNN component of the Conformer. The resulting model structure comparison is as follows Table 3.

From the experimental results in Table 4, it can be observed that the inclusion of CNN improves the model's performance across various metrics compared to other baseline models. However, the addition of Macaron FFN+RPE does not lead to improvement in every aspect of the model. The standalone use of CNN, lacking attention mechanisms, results in situations where the model struggles to make accurate predictions and inferences.

Table 3. The model structure of ablation experiment.

Model Structure	Attention	CNN	Macaron FFN+RPE
Conformer	✓	✓	✓
Transformer + Macaron FFN+RPE	✓		✓
Transformer	✓		
Hawthorne *et al.*	✓		
CNN in Conformer		✓	

Table 4. The result of ablation experiment.

Model Structure	Onset F1	Onset & Offset F1	Onset, offset, & Velocity F1
Conformer	96.38	85.34	84.27
Transformer + Macaron FFN+RPE	96.27	84.98	83.70
Transformer	96.31	84.61	82.75
Hawthorne *et al.*	96.01	83.94	82.75
CNN in Conformer	-	-	-

3.3 Parametric Sensitivity Analysis

To achieve the optimal performance, comparative experiments were conducted on model-specific parameters, including the number of heads, layers, Mel bin size, and short-time Fourier transform granularity. Parameters not directly correlated to model outcomes, such as batch size and sampling rate, were excluded from the scope of this study.

First, to explore the effect of Conformer layers, the number of heads was fixed at 4 and the input dimension at 512, while gradually increasing the number of Conformer blocks. Based on the MAESTRO V3 dataset, the F1 Score for each experiment was systematically recorded (Fig. 5a), revealing a significant trend: when the number of layers was less than 8, the model's performance improved significantly with increasing layers; however, once the number of layers exceeded 8, the model's performance began to decline significantly. This performance degradation directly indicates the phenomenon of overfitting.

Next, the impact of Conformer heads on model performance was investigated. In this experiment, the number of layers was fixed at 8, and the number of Mel bins at 512, while varying the number of heads. The experimental results (Fig. 5b) showed that when the number of heads was less than 4, the model's performance improved as the

number of heads increased. However, once the number of heads exceeded 4, the phenomenon of overfitting was observed.

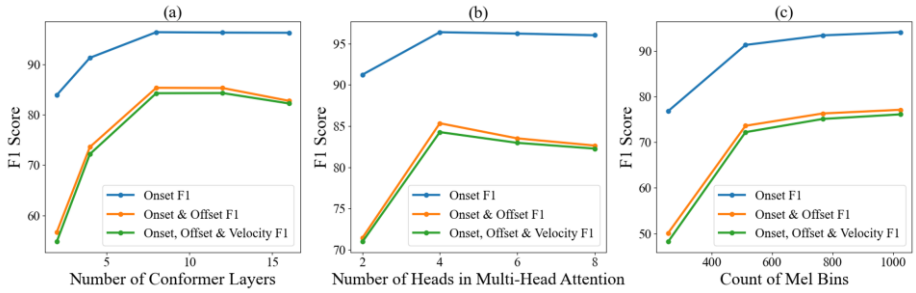

Fig. 5. Impact of Conformer heads, layers, and Mel bins.

After that, by adjusting the number of Mel bins, which determines the input dimension of the model, the frequency resolution of the audio signals was effectively controlled, and the model's performance was evaluated under different settings. The results revealed that as the number of Mel bins increased from a lower value, the model's performance significantly improved, attributable to the higher resolution frequency information enabling the model to capture key features in the audio more accurately. However, when the number of Mel bins count surpassed 512, a diminishing impact on model performance was observed with further increases in audio layer resolution. This indicates that while incorporating more Mel bins may yield marginal performance gains, it also substantially elevates computational costs and resource requirements, necessitating a balance between resolution enhancement and computational efficiency.

4 Discussions

4.1 The Advantage of Conformer

The reason why Conformer outperforms other models is due to several optimizations and improvements, particularly in the Conformer block:

- **Attention modules:** After the introduction of attention mechanisms following the Transformer model [26], they have found widespread application in the field of speech recognition. Experimental findings in the Conformer model underscore the pivotal role of attention mechanisms. Devoid of attention mechanisms, the model faces challenges in accurately transcribing musical scores, highlighting the indispensable nature of attention mechanisms in this context.
- **Relative position encoding and Macaron feed-forward [27]:** Conformer adopts relative position encoding, inspired by Transformer-XL. Relative position encoding helps to better handle tokens of different lengths, enhancing the model's versatility. Inspired by Macaron-Net [28], Conformer replaces the single-layer feed-forward in the Transformer with two half-step feed-forward. This improvement, validated in other studies [29,30], contributes to enhanced model performance. While these two

enhancements have demonstrated positive effects in the field of speech recognition, their impact on music transcription has not resulted in a comprehensive improvement across all aspects.

- **Convolutional modules:** Conformer introduces convolutional modules, which go beyond simple convolutional layers. It incorporates multi-layer convolutions with gating mechanisms and employs depth-wise separable convolutions [31,32], reducing the computational complexity of the attention mechanism and accelerating model convergence. In the context of music transcription, the application of CNNs to Seq2Seq models has primarily been within models that combine CNNs with LSTMs. However, there has been limited exploration into incorporating CNNs into Transformer-based architectures to enhance accuracy. Experiments reveal that the integration of CNNs into Seq2Seq models with Transformer as the main core leads to a partial improvement in accuracy. This fusion demonstrates enhanced accuracy for evaluation metrics based on both 'onset-only' and the combined consideration of 'onset, offset, and velocity.' This novel integration opens promising avenues for improving the performance of music transcription models.

These optimizations, when combined, make the Conformer model excel in music score transcription tasks, demonstrating higher efficiency and accuracy compared to traditional Transformer architectures. These enhancements position Conformer as an ideal choice for processing both audio data, yielding satisfactory experimental results.

4.2 Possible Reasons for Mistakes

As shown in Fig. 6, this is an example with five notes as ground truth results, corresponding to a set of ten note events. During prediction, Seq2Seq models may encounter two situations: 'music event mistakes' (*e.g.*, y_1) and 'absence' (*e.g.*, y_6). When situation y_1 occurs, the note corresponding to y_1 (*e.g.*, $C\#7$) will be activated, but the actual result should be $E3$. However, since there is a 50ms tolerance during matching, the note event-based evaluation results will not be significantly affected, and the impact on frame-wise results will also be minimal.

There will also be cases of 'absence' note events, where the current note (*e.g.*, $B4$) will remain activated until it encounters a note start event y_{11}. Due to the large time gap between these two notes, it can lead to inaccurate onset evaluation metrics and severe false positives in the frame-wise results. Therefore, the absence of note events can significantly affect evaluation metrics.

To address this issue, this paper performs a simple post-processing step by adding the most likely note duration for missing note end events. Experiments were conducted by comparing Conformer models with different possible note lengths from 4 to 10 seconds on the MAESTRO V1 dataset. The results evaluated the improvement in model frame-wise evaluation with different possible note lengths and their impact on factors such as note start times, end events, note velocity, *etc*. The results are shown in Table 5. The same treatment also leads to a similar performance improvement in the Transformer model.

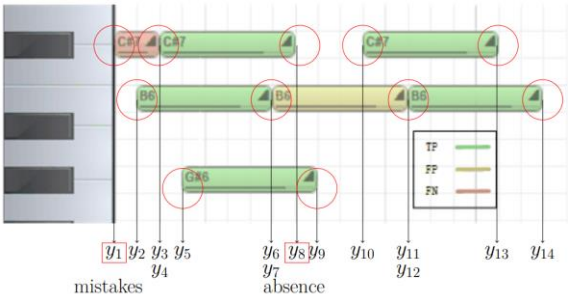

Fig. 6. An example of prediction, with true positive note frames in green, false positive note frames in blue, and false negative note frames in red.

Table 5. Results with different maximum note lengths.

Maximum Note Length	Onset F1	Onset & Offset F1	Onset, Offset & Velocity F1	Frame F1
0	96.69	86.46	85.53	70.52
5	96.69	86.40	85.48	85.66
6	96.69	86.44	85.51	85.03
7	96.69	86.47	85.54	84.40
8	96.69	86.48	85.56	83.80
9	96.69	86.48	85.56	83.21
10	96.69	86.47	85.56	82.66

5 Conclusions

Successfully adapting the best-performing models from the field of speech recognition for music transcription with minor modifications. This enables us to convert spectrograms into human-readable music scores efficiently. The integration of the best-performing models from speech recognition into music transcription was achieved seamlessly through the commonalities in these domains. This approach involves directly applying models of similar scale, specifically trained on music data, and has proven to yield the best results in the field.

Furthermore, our experiments shed light on why previous efforts struggled to achieve good frame-wise performance. It was due to the omission of note-off events during prediction, causing notes to extend into the next note, resulting in numerous false positives. Assigning a maximum possible note length to each note helps preserve note duration for note-related evaluation metrics and substantially enhances the accuracy of frame-wise evaluations in music transcription. The experimental results clearly demonstrate that a slightly modified Conformer model can outperform all existing models. The inclusion of convolution layers allows the model to capture local details better, resulting in superior performance over models trained on various versions of the MAESTRO dataset. Assigning a possible note length to ongoing notes improved the frame-wise evaluation metric from 70.52% to an impressive 85.66%. This enhancement

greatly boosts the performance of Seq2Seq models in frame-wise evaluations while having no impact on evaluation metrics based on note events.

Due to time constraints, determining the length of missing note-off events should ideally be a function or model related to factors like playing intensity. The paper addresses this issue through a simple post-processing approach. Future work will involve the design of models or the discovery of functions to accurately predict the length of missing note-off events, making post-processing more scientifically grounded and precise. Additionally, it was found that training models as streaming models is more practical, as the plan is to apply streaming models from speech recognition to music transcription for real-time transcription.

Acknowledgments. The work was supported by the National Natural Science Foundation of China (No. 62203237), the National Key R&D Program of China (2021YFB0300104), and Haihe Lab of ITAI.

References

1. Poliner, G.E., Ellis, D.P.: A discriminative model for polyphonic piano transcription.
2. Nam, J., Ngiam, J., Lee, H., Slaney, M.: A classification-based polyphonic piano transcription approach using learned feature representations. In: Ismir. pp. 175–180. Citeseer (2011)
3. Böck, S., Schedl, M.: Polyphonic piano note transcription with recurrent neural networks. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). pp. 121–124. IEEE (2012)
4. Hawthorne, C., Elsen, E., Song, J., Raffel, A.R.I.S.C., Eck, J.E.S.O.D.: Onsets and frames:Dual-objective piano transcription
5. Kim, J.W., Bello, J.P.: Adversarial learning for improved onsets and frames music transcription. In: 20th International Society for Music Information Retrieval Conference, ISMIR 2019. pp. 670–677. International Society for Music Information Retrieval (2019)
6. Kong, Q., Li, B., Song, X., Wan, Y., Wang, Y.: High-resolution piano transcription with pedals by regressing onset and offset times. IEEE/ACM Transactions on Audio, Speech, and Language Processing 29, 3707–3717 (2021)
7. Wei, W., Li, P., Yu, Y., Li, W.: Hppnet: Modeling the harmonic structure and pitch invariance in piano transcription. In: Ismir 2022 Hybrid Conference (2022)
8. Gardner, J.P., Simon, I., Manilow, E., Hawthorne, C., Engel, J.: Mt3: Multi-task multitrack music transcription. In: International Conference on Learning Representations (2021)
9. Hawthorne, C.G.M., Simon, I., Swavely, R.J., Manilow, E., Engel, J.: Sequence-to-sequence piano transcription with transformers (2021)
10. Nguyen, T.Q., Salazar, J.: Transformers without tears: Improving the normalization of self-attention. In: Proceedings of the 16th International Conference on Spoken Language Translation (2019)
11. Sigtia, S., Benetos, E., Dixon, S.: An end-to-end neural network for polyphonic piano music transcription. IEEE/ACM Transactions on Audio, Speech, and Language Processing 24(5), 927–939 (2016)
12. Awiszus, B.S.M.: Automatic music transcription using sequence to sequence learning. Ph.D. thesis, Master's thesis, Karlsruhe Institute of Technology (2019)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–

1780 (1997)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems 30 (2017)
15. Kwon, T., Jeong, D., Nam, J.: Polyphonic piano transcription using autoregressive multi-state note model (2020)
16. Gulati, A., Qin, J., Chiu, C.C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., Pang R.: Conformer: Convolution-augmented transformer for speech recognition.Interspeech 2020 (2020)
17. Raffel, C., McFee, B., Humphrey, E.J., Salamon, J., Nieto, O., Liang, D., Ellis, D.P., Raffel, C.C.: Mir_eval: A transparent implementation of common mir metrics. In: ISMIR. vol. 10, p. 2014 (2014)
18. Owens, F., Murphy, M.: A short-time fourier transform. Signal Processing 14(1), 3–10 (1988)
19. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems 30 (2017)
20. Huber, D.M.: The MIDI manual. Sams (1991)
21. Ramachandran, P., Zoph, B., Le, Q.V.: Searching for activation functions (2017)
22. Loshchilov, I., Hutter, F.: Fixing weight decay regularization in adam (2017)
23. Edmonds, J.: Maximum matching and a polyhedron with 0, 1-vertices. Journal of research of the National Bureau of Standards B 69(125-130), 55–56 (1965)
24. EURASIP Journal on Advances in Signal Processing 2007, 1–9 (2006)
25. Xiao, Z., Chen, X., Zhou, L.: Polyphonic piano transcription based on graph convolutional network. Signal Processing 212, 109134 (2023)
26. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems 30 (2017)
27. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., Salakhutdinov, R.: Transformer-xl: Attentive language models beyond a fixed-length context. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics (2019)
28. Lu, Y., Li, Z., He, D., Sun, Z., Liu, T.Y.: Understanding and improving transformer from a multi-particle dynamic system point of view (2019)
29. Nguyen, T.Q., Salazar, J.: Transformers without tears: Improving the normalization of self-attention. In: Proceedings of the 16th International Conference on Spoken Language Translation (2019)
30. Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D.F., Chao, L.S.: Learning deep transformer models for machine translation. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. (2019)
31. Wu, Z., Liu, Z., Lin, J., Lin, Y., Han, S.: Lite transformer with long-short range attention. In: International Conference on Learning Representations (2019)
32. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M.,Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications (2017)