TISMIR

Cross-Modal Approaches to Beat Tracking: A Case Study on Chopin Mazurkas

RESEARCH ARTICLE

ubiquity press

CHING-YU CHIU
LELE LIU
CHRISTOF WEIß
MEINARD MÜLLER

*Author affiliations can be found in the back matter of this article

ABSTRACT

Within music information retrieval (MIR) research, numerous beat-tracking systems have been developed, targeting either audio recordings or symbolic representations such as MIDI files. However, the differences between these approaches, their respective strengths and weaknesses, and the potential for combining them have received limited attention. In this article, we compare two conceptually different beat trackers: an audio-based model that operates frame by frame and a symbolic-based model using an event-driven approach. Specifically, we analyze the performance of two pretrained systems: the audio beat tracker `madmom` and the symbolic beat tracker Performance MIDI-to-Score (`PM2S`). Our evaluation is based on a cross-modal dataset of Chopin's Mazurkas (Maz-5), which includes multiple audio recordings and MIDI representations automatically transcribed from audio. As a key contribution, we standardize the post-processing pipelines for the frame-based and event-based beat trackers to ensure comparability and explore various late-fusion methods within a unifying framework. Our results highlight the effectiveness of these fusion strategies in leveraging the strengths of both modalities while providing valuable insights into the performance of existing beat-tracking models.

CORRESPONDING AUTHOR:
Ching-Yu Chiu
International Audio Laboratories Erlangen, Germany
ching-yu.chiu@audiolabs-erlangen.de

KEYWORDS:
Audio beat tracking, symbolic beat tracking, cross modalities

TO CITE THIS ARTICLE:
Chiu, C.-Y., Liu, L., Weiß, C., & Müller, M. (2025). Cross-Modal Approaches to Beat Tracking: A Case Study on Chopin Mazurkas. *Transactions of the International Society for Music Information Retrieval, 8*(1), 55–69.
DOI: https://doi.org/10.5334/tismir.238

1 INTRODUCTION

Given a piece of music, the goal of beat tracking is to find the time positions humans would tap along with while listening. Existing beat-tracking systems generally consist of two parts, an *activity estimator* and a *post-processor*. The activity estimator is usually a feature learning network, which processes the input representation and generates an activation function, indicating the likelihood or pseudo-probability of each time position to be a beat or not. The post-processor is typically a dynamic programming-based model that takes the activation function and determines beat positions through optimization. On the basis of the input representation, existing beat-tracking systems can be categorized as *audio beat trackers*, which take as input an audio representation (Böck and Davies, 2020; Fuentes et al., 2018), or *symbolic beat trackers* (*SBTs*), which take as input a symbolic representation (Chuang and Su, 2020; Liu et al., 2022). While both audio and symbolic beat trackers aim to detect the underlying beat structure of music, their differing input representations lead to distinct mechanisms. Audio-based approaches focus on recovering onset information hidden in frame-based input representations, while symbolic approaches, which have access to explicit onset information encoded in event-based representations, can concentrate on higher-level metrical aspects. As shown in Figure 1, given a piece of music, the audio waveform and symbolic MIDI file encode the music in different formats, leading to beat activation estimation approaches based on different principles. The audio waveforms encode both strong and subtle changes in amplitude and frequency, which the audio activity estimator uses to identify relevant note events. In contrast, symbolic MIDI files explicitly specify note event timing, allowing the symbolic activity estimator to focus directly on these specific positions. The differences mentioned above may lead to distinct model behaviors, yet few studies investigate these and their potential complementarity.

From a higher-level viewpoint, beat-tracking errors can be attributed to both *data-related factors* (e.g., musical properties determined by the music scores or physical properties of specific performances (Grosche et al., 2010)) and *model-related factors* (e.g., the decision mechanisms of the activity estimators or the assumptions involved in the post-processors). The choice of evaluation metrics can also bias observations of the model's true behavior.[1] Without systematic control of these factors, the insights we can derive from experiments may be limited. For example, despite the fact that existing works generally report beat-tracking *F*1 scores; correct metrical level, with the total of continuous segments (CMLt); and allowed metrical level, with the total of continuous segments (AMLt) (Davies et al., 2009) for various datasets (Chang and Su, 2024; Cheng and Goto, 2023; Hung et al., 2022; Pinto et al., 2021; Zhao et al., 2022), their reliance on the dynamic Bayesian network (DBN) built into `madmom` (Böck et al., 2016a) or similar approaches (Yamamoto, 2021) imposes strong tempo assumptions, limiting our understanding of what their neural-network-based activity estimators truly learn. *F*1 scores, CML_t, and AML_t also do not allow us to progressively evaluate how well the models handle longer musical contexts. Although researchers have recently recognized the influence of post-processors and purposely adopted peak-picking-based methods (Chiu et al., 2023; Foscarin et al., 2024) without strong assumptions, the impact of peak-picking thresholds in combination with the overall amplitudes of activation functions has rarely been investigated in detail. Moreover, despite the release and adoption of the Aligned Scores and Performances (ASAP) for the piano transcription dataset (Foscarin et al., 2020), which provides a comprehensive understanding of the limitations of existing deep learning (DL)-based methods in the context of classical piano music analysis, the musical complexities in the dataset also hinder our understanding of model failures (Chiu et al., 2023; Foscarin et al., 2024).

Considering the issues mentioned above, the goal of this article is to investigate and analyze both symbolic and audio beat trackers, with a better control of factors related to data and models. To the best of our knowledge, the only existing work that compares the two modalities is by Schwarzhuber (2024). However, despite conducting comprehensive experiments regarding network architectures and various input representations, the insights in the work by Schwarzhuber (2024) are also limited by the abovementioned factors: adoption of post-processors with strong assumptions (i.e., DBN and rule-based method by Liu et al. (2022)) and limited reported evaluation metrics (i.e., only *F*1 scores without reporting or discussing precision, recall, or other evaluation metrics). Additionally, there are two types of input data representations used in the beat trackers: spectrograms with evenly distributed time stamps along the time axis (referred to as "frame-based representation") and event sequences with unevenly distributed time stamps (referred to as "event-based representation"). For both representations, there are corresponding post-processors with very different assumptions, complicating the comparison.

In contrast to existing works, we address the aforementioned factors by establishing a unified post-processing pipeline to better understand and directly compare the properties of activation functions without altering them significantly, and to derive beat estimations without imposing strong assumptions beyond the mechanisms within the activity estimators. We focus on a cross-version dataset of five Chopin Mazurkas (Sapp, 2007; 2008) featuring expressive tempo changes (Schreiber et al., 2020; Shi, 2021) and a variety of specified musical properties, including non-event beats, boundary

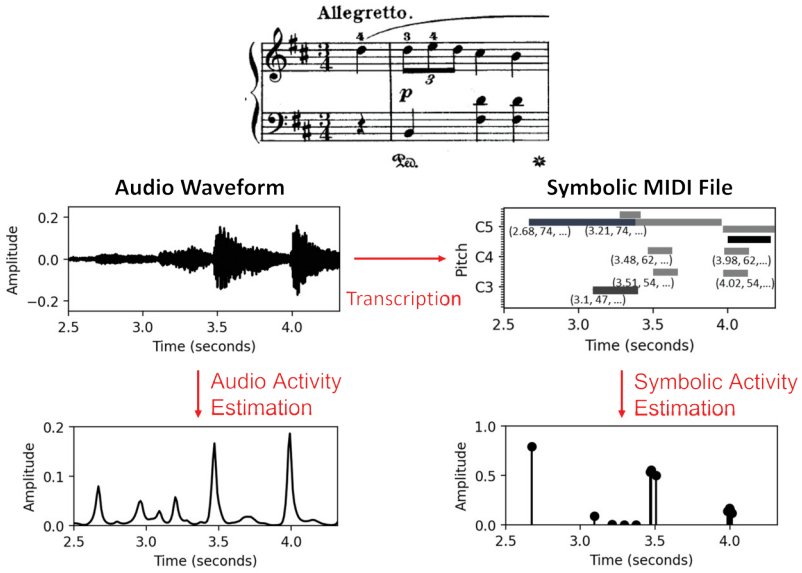

Figure 1 Beat activity estimation of an audio representation using a frame-based approach **(left)** and a symbolic representation using an event-based approach **(right)**.

beats, ornamented beats, constant harmony beats, and weak bass beats that complicate beat tracking (Grosche et al., 2010).[2] Furthermore, we propose representation conversion methods combined with various fusion techniques to explore potential ways to leverage the complementarity between audio and symbolic modalities. We start with existing pre-trained models: `madmom` (Böck et al., 2016a,b) as a frame-based audio beat tracker and `PM2S` (Liu et al., 2022) as an event-based symbolic beat tracker. More specifically, the `madmom` activity estimator employs a Bidirectional Long Short-Term Memory (BiLSTM) network to process audio spectrograms and generate frame-based activations (as shown in Figure 1, left bottom), while `PM2S` uses a convolutional recurrent neural network (CRNN) to process MIDI note sequences and generates event-based activations (as shown in Figure 1, right bottom). We also retrain the BiLSTM and CRNN using the five Mazurkas (Maz-5) to further investigate the capability of these models to adapt to a specific music scenario and to validate the effectiveness of various late-fusion methods.

Figure 2 provides a conceptual overview of our cross-modal approach, featuring two branches: the audio/frame-based branch (left) and the symbolic/event-based branch (right). Specifically, in the case of a frame-based audio beat tracker (Figure 2, left), the audio waveform from a recording is converted to a frame-based feature representation X^{A} (e.g., a spectrogram) and processed by an audio activity estimator (purple square) to derive the audio activations Δ^{A}. A frame-based post-processor (green square) then converts the activations Δ^{A} into beat estimations. Similarly, in the case of an event-based symbolic beat tracker (Figure 2, right), a sequence of event tuples X^{S}, obtained from a symbolic representation (e.g., a MIDI file), is processed by a symbolic activity estimator (purple square) to derive the corresponding symbolic activations Γ^{S}. An event-based post-processor (green square) then converts the symbolic activations Γ^{S} into beat estimations. In principle, as indicated in Figure 2, one may leverage frame-to-event (F2E) and event-to-frame (E2F) conversion to bridge the two modalities and explore various fusion methods (blue squares) in both representations, including frame-based and event-based fusion.[3] In this paper, we focus on the frame-based representation (left)[4] and establish a frame-based unified post-processing pipeline (green square) to handle activations from all modalities (i.e., audio, symbolic, and late fusion) consistently, converting them into beat estimations for further analysis.

The remainder of this paper is organized as follows: In Section 2, we introduce the dataset of Chopin Mazurkas (Maz-5) and our experimental scenarios. In Section 3, we present the mathematical definitions and notations for existing beat-tracking methods of different modalities. In Section 4, we describe the representation conversion methods. In Section 5, we outline the unified post-processing pipeline and the peak-picking method considered. Then, in Section 6, we present the experiment results. In Section 7, we extend the experiment to downbeat tracking and discuss the preliminary results. Finally, we conclude this article in Section 8. We provide the Python source code and supplementary discussions for this paper at GitHub repository: https://github.com/SunnyCYC/CrossModalBeat.

2 SCENARIO: CHOPIN MAZURKAS

The dataset for our experiments consists of five Chopin Mazurkas (Maz-5), selected from the Mazurka Project's

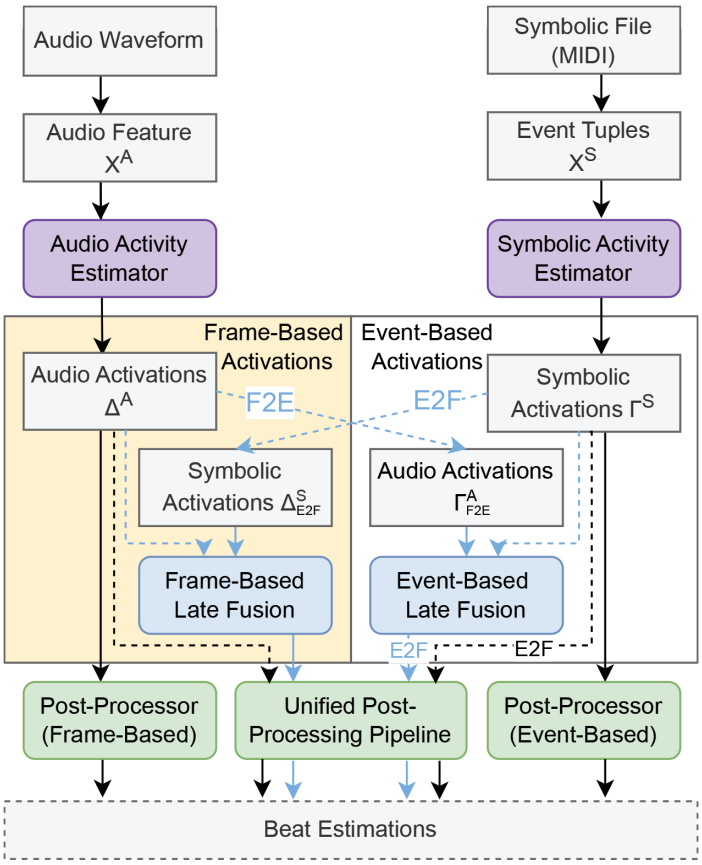

Figure 2 Overview of the system. F2E: frame-to-event conversion; E2F: event-to-frame conversion.

ID	Piece	Number (Beats)	Number (Perf.)	Dur. (h)
M17-4	Op. 17, No. 4	396	64	4.62
M24-2	Op. 24, No. 2	360	64	2.44
M30-2	Op. 30, No. 2	193	34	0.80
M63-3	Op. 63, No. 3	229	88	3.15
M68-3	Op. 68, No. 3	181	51	1.43

Table 1 The five Chopin Mazurkas and their identifiers used in our study. The last three columns indicate the number of beats, performances, and total duration (in hours) available for the respective piece. Dur.: duration; h: hours; ID: identifier; Perf.: performances; Op.: Opus.

collection of 49 Mazurkas.[5] For each of the five Mazurkas, Sapp (2007; 2008) manually annotated beat and downbeat positions in 34–88 audio recordings of real performances (Table 1). As a piano dataset featuring expressive tempo changes (e.g., *rubato* or *ritardando*), Maz-5 is typically used for analysis of beat-tracking errors (Grosche et al., 2010), local tempo (Schreiber et al., 2020), or expressive timing (Shi, 2021). Moreover, the cross-version scenario (i.e., having multiple recorded performances for each of the Mazurkas) of Maz-5 allows us to separate musical factors (e.g., musical complexities determined by scores) and physical factors (e.g., audio quality or interpretations of pianists) for beat tracking to gain more insights. Although no performance MIDI files exist for the 301 recordings, recent advances in DL-based piano transcription enable the creation of MIDI files with sufficient quality for further analysis (e.g., for beat tracking) and to serve as symbolic input representations. For our experiments, we use the transcriber by Kong et al. (2021) to derive MIDI files (without further manual correction). In Section 6, we present the insights gained from these key features of Maz-5.

3 BEAT TRACKING IN DIFFERENT MODALITIES

In this section, we present the key concepts, terminology, and mathematical definitions related to audio and symbolic beat-tracking methods.

3.1 AUDIO BEAT TRACKERS

Commonly, an audio beat tracker (ABT) takes as input a spectrogram-like time–frequency representation (e.g., a Mel spectrogram) calculated from an audio

recording and outputs the estimated beat positions. Specifically, an audio beat tracker consists of an audio activity estimator, which processes the input representation and generates an activation function, and a post-processor, which converts the activation function into beat positions. In this case, the time axis of input data is organized on the basis of regularly sampled frames. From this point forward, we refer to this as the *frame-based* organization of the time axis.

In the following, we use the shorthand mathematical formulas based on these two notations: $[0 : N-1] := \{0, 1, ..., N-1\}$ for $N \in \mathbb{N}$ and $[0,1] := \{v \in \mathbb{R} \mid 0 \leq v \leq 1\}$. Let F_s denote the frame rate (given in frames per second (FPS)), which specifies the number of frames per second determined by the sampling method. Given a music piece with a duration of t seconds, we assume that the (spectrogram-like) audio feature representation is provided by a matrix encoded as follows:

$$X^{\mathrm{A}} : [0 : M-1] \times [0 : K-1] \to \mathbb{R}, \tag{1}$$

where $M = \lfloor t \cdot F_s \rfloor + 1$ denotes the total number of frames and K represents the number of features. An audio beat activity estimator takes X^{A} as input and generates the following activation function:

$$\Delta^{\mathrm{A}} : [0 : M-1] \to [0,1], \tag{2}$$

which maps each time frame to its corresponding likelihood (or pseudo-probability) of being a beat. See Figure 3a for an example of Δ^{A}. A frame-based post-processor can then analyze the activation function Δ^{A} and determine the estimated beats (as frame indices) $B = (b_0, b_1, \ldots, b_{L-1})$. Here, B is a sequence of length $L \in \mathbb{N}$ consisting of strictly monotonically increasing beat positions $b_\ell \in [0 : M-1]$ for $\ell \in [0 : L-1]$. Common frame-based post-processors include predominant local pulse (PLP) (Grosche and Müller, 2011; Meier et al., 2024), dynamic programming-based approaches (Ellis, 2007) such as dynamic Bayesian networks (DBNs) (Böck et al., 2016b), and conditional random field-based methods (CRFs) (Fuentes et al., 2019).

3.2 SYMBOLIC BEAT TRACKERS

For a symbolic beat tracker (SBT) that consists of a symbolic activity estimator and a post-processor, the symbolic input can be organized in either a frame-based manner (as introduced in Section 3.1) using piano-roll-like input representations (Chuang and Su, 2020), or using an event-based representation (e.g., an event sequence). In this study, we focus on `PM2S`,[6] which takes as input an event sequence obtained from MIDI files (Liu et al., 2022) and generates an event-based activation function. Specifically, for an event-based method, the time axis of input data is organized on the basis of a list of unevenly distributed time stamps. These time stamps could be the timing of any musical attributes encoded by a set $\mathcal{A}$, such as pitches, durations, note velocity, instrumentation, or others (Liu et al., 2022). For instance, in the context of MIDI, $\mathcal{A} = [0 : 127] \times [0 : 127]$ may represent the set of possible MIDI pitch and velocity values for a note. Given a list of N time stamps (in seconds) noted as $T = (t_0, t_1, \ldots, t_{N-1})$, with $t_n \in \mathbb{R}_{\geq 0}$ and $t_0 \leq t_1 \leq \ldots \leq t_{N-1}$, the input representation of an event-based symbolic activity estimator can be organized as a list

$$X^{\mathrm{S}} = ((t_0, a_0), \ldots, (t_{N-1}, a_{N-1})) \tag{3}$$

where attribute $a_n \in \mathcal{A}$, and $n \in [0 : N-1]$. A tuple $(t_n, a_n) \in \mathbb{R} \times \mathcal{A}$ is also called an event.

An event-based symbolic activity estimator takes X^{S} as input and generates as output the following activation function:

$$\Gamma^{\mathrm{S}} : [0 : N-1] \to \mathbb{R} \times [0,1]. \tag{4}$$

Specifically, the event-based symbolic activation function Γ^{S} maps a time index n to a tuple $\Gamma^{\mathrm{S}}(n) = (t_n, p_n)$ with a time stamp t_n and a pseudo-probability p_n for that time position to be a beat. See Figure 3e for an example of Γ^{S}.

An event-based post-processor (Liu et al., 2022) can analyze the event-based activation function Γ^{S} and generate estimated beat positions $B = (b_0, b_1, \ldots, b_{L-1})$, where B is a sequence of length $L \in \mathbb{N}$ consisting of strictly monotonically increasing beat positions with $b_\ell \in T$ for $\ell \in [0 : L-1]$.

4 EVENT-TO-FRAME ACTIVATION CONVERSION

In this section, we describe our approach to event-to-frame (E2F) conversion, which not only allowd us to bridge the frame-based audio and the event-based symbolic activations for further fusion but also enables us to process activations from all modalities consistently for subsequent analysis.

Given a list of monotonically increasing time points $T = (t_0, t_1, \ldots, t_{N-1})$ with $t_n \in \mathbb{R}_{\geq 0}$ and the corresponding event-based activation function $\Gamma^{\mathrm{S}} : [0 : N-1] \to \mathbb{R} \times [0,1]$, with $\Gamma^{\mathrm{S}}(n) = (t_n, p_n)$, we can convert it into a frame-based representation $\Delta^{\mathrm{S}}_{\mathrm{E2F}} : [0 : M-1] \to [0,1]$ based on a specified frame rate F_s. Let $M \in \mathbb{N}$ with $M = \lfloor t_{N-1} \cdot F_s \rfloor + 1$ be the total number of frames, then we define

$$s_n := \lfloor t_n \times F_s \rfloor, \tag{5}$$

for $n \in [0 : N-1]$. Note that $s_n \in [0 : M-1]$ is the frame index corresponding to the time stamp t_n. To account for the case where several time stamps are assigned to the same frame index, we further define

$$p^*_n := \max\{p_\ell \mid s_\ell = s_n, \ell \in [0 : N-1]\} \tag{6}$$

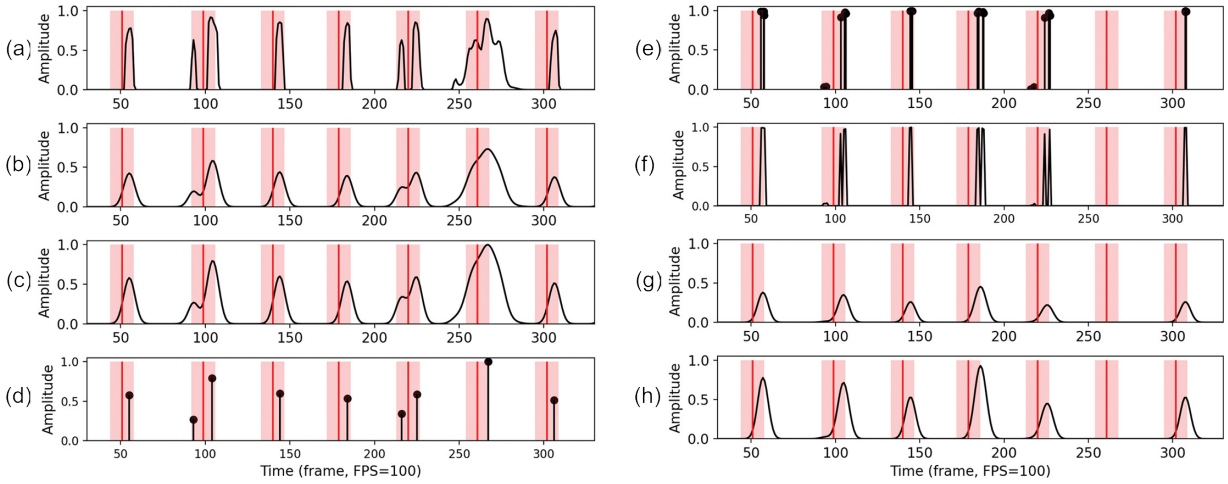

Figure 3 Processing of beat activation functions. **(a)** Frame-based activation function from an audio activity estimator. **(b)** Gaussian smoothing of (a). **(c)** Max normalization of (b). **(d)** Peak-picking results of (c). **(e)** Event-based activation function from a symbolic activity estimator. **(f)** Event-to-frame conversion of (e). **(g)** Gaussian smoothing of (f). **(h)** Max normalization of (g). Red vertical lines indicate reference annotated beats. Red regions highlight the ±70 ms tolerance window.

to keep the maximum probability p_n^* among those stamps. The converted event-based activation function is then defined as follows:

$$\Delta^S_{\text{E2F}}(m) := \begin{cases} p_n^*, & \text{for } m = s_n, \\ 0, & \text{otherwise}, \end{cases} \tag{7}$$

for $m \in [0 : M-1]$. Figure 3e and 3f provides an example of E2F conversion.

5 UNIFIED POST-PROCESSING PIPELINE

As mentioned in Section 1, we establish a unified post-processing pipeline to handle activations of all modalities (i.e., audio, symbolic, and fused) consistently, aiming to preserve the original properties of the activations without imposing strong assumptions. This is mainly achieved by converting all event-based activations to frame-based activations using E2F conversion, smoothing and normalizing all derived frame-based activations, and then applying peak-picking with explicit and transparent settings. In this section, we describe our methods for smoothing and normalizing (Section 5.1), and peak-picking methods with three types of threshold settings (Section 5.2 and Section 5.3). The effectiveness of these methods are then justified in Section 6.2.

5.1 SMOOTHING AND NORMALIZATION

As pointed out by Müller and Chiu (2024), novelty enhancement strategies, such as smoothing and normalization, are essential before peak-picking. By applying Gaussian smoothing and max normalization, we can reduce spurious peaks caused by irrelevant fluctuations, filter out peaks occurring at a frequency too high for natural beat tracking, and standardize the activation values to make them comparable and treatable as pseudo-probabilities.

For each frame-based activation function, following Müller and Chiu (2024), we first apply a one-dimensional Gaussian filter, defined in Equation (8), to smooth out irrelevant local fluctuations. Let

$$g_\sigma(u) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{u^2}{2\sigma^2}), \tag{8}$$

be a truncated Gaussian function, where $u \in [-3\sigma : 3\sigma]$. Taking into account the conventional tolerance window of ±70 ms for beat-tracking evaluation, we empirically fixed $\sigma = 3$ (frames) at a frame rate of 100 FPS. After smoothing, we apply max normalization[7] by dividing each activation function by its maximum value to scale all values to the range $[0, 1]$. Figure 3 illustrates the effects of Gaussian smoothing and max normalization.

5.2 GLOBAL THRESHOLD

One of the simplest methods to convert activation functions into estimated beat positions is to apply peak-picking with a global fixed threshold τ, e.g., $\tau = 0.5$. Despite being simple and commonly adopted (Böck et al., 2016b; Chiu et al., 2023; Foscarin et al., 2024), this method is sensitive to the threshold values. While a high threshold may remove too much information from the activation functions and lead to a low recall value, a low threshold may retain noise or irrelevant activation peaks, and lead to a low precision value. The optimal threshold for each track depends on both the data and the model (i.e., activity estimator) and is generally unknown or not specified in many use cases. In this study, we experiment with thresholds $\tau \in \{0.01, 0.0625, 0.1, 0.125, 0.25, 0.5, 0.75, 0.875\}$.

Additionally, given the reference annotated beat positions, one can also investigate the upper bound performance of an activation function by performing a grid search. Specifically, a range of global peak thresholds (i.e., $\tau \in \{0.01, 0.0625, 0.1, 0.125, 0.25, 0.5, 0.75, 0.875\}$) is defined, and peak-picking is applied with each threshold to each test track, reporting only the highest *F*1 scores. We refer to this reference-informed method as "oracle threshold."

5.3 ADAPTIVE LOCAL THRESHOLD

In contrast to a fixed global threshold, an adaptive threshold that adjusts on the basis of the local properties of the activation function is generally more robust and less sensitive to the hyperparameter settings (Müller and Chiu, 2024). In this study, we determine a local average function $\mu : [0 : M-1] \to \mathbb{R}$ defined by

$$\mu(m) := \frac{1}{2W+1} \sum_{\ell=-W}^{W} \Delta(m+\ell), \tag{9}$$

and use it as the local threshold of the peak-picking. The $m \in [0 : M-1]$ and the parameter $W \in \mathbb{N}$ sets the size of the averaging window. Zero padding is applied to avoid summation over negative indices. In this study, we experiment with *W* corresponding to 5, 10, or 20 seconds.

6 EXPERIMENTS

In this section, we first describe the evaluation metrics (Section 6.1), then report our experiment results of pretrained models (Section 6.2) and retrained models (Section 6.3), followed by a discussion of insights derived from the Maz-5 scenario (Section 6.4).

6.1 EVALUATION METRICS

Following the conventional evaluation setting for beat tracking, we adopt from `mir_eval` the metrics of *F*1 score (*F*1), precision (P), and recall (R) with an error tolerance of ±70 ms. Furthermore, we also compute the *L*-correct metric (Chiu et al., 2022; Grosche and Müller, 2011) to evaluate how well the beat trackers can handle a longer musical context. Rather than evaluating each annotated reference beat individually, the *L*-correct metric requires that at least *L* consecutive reference beats are correctly detected, where $L \in \mathbb{N}$ is an adjustable parameter. In this work, we report the *F*-measure of *L*-correct with $L \in \{2, 3, 4\}$, denoted as *F*-*L**.

6.2 PRETRAINED BEAT TRACKERS

In this section, we evaluate the pretrained activity estimators (i.e., `madmom` for audio modality and `PM2S` for the symbolic one) using our unified post-processing pipeline (Section 5) and the Maz-5 dataset. Since neither of these two systems was trained on the Maz-5 dataset, the evaluation fairly and accurately reflects their performance on unseen classical music. We also experiment with late fusion of the two modalities to investigate the underlying complementarity between the two types of models.

6.2.1 Activity estimators and post-processors

Both `madmom` and `PM2S` can jointly generate beat and downbeat activations. In this work, we focus more on the beat activations. For the audio modality (A) we adopt the `RNNDownBeatProcessor` of `madmom` to derive the beat activations Δ^{A}. For the symbolic modality (S), we first transcribe the 301 recordings using the model by Kong et al. (2021) to derive MIDI files. We then apply the CRNN activity estimator by Liu et al. (2022) (referred to as `PM2S`) to derive the beat activation functions Γ^{S}. Finally, we conduct the E2F conversion to derive $\Delta^{\mathrm{S}}_{\mathrm{E2F}}$. As described in Section 5, these frame-based activation functions will be smoothed and normalized before peak-picking. We further experiment with two simple late-fusion methods: addition-based (A+S) and (element-wise) multiplication-based (A×S) methods. Specifically, we take the smoothed and normalized activation functions of single modalities (i.e., A and S), add them up or multiply them point-wise, and smooth and normalize the fused activations again before peak-picking.

As described in Section 5, for all the derived frame-based activations mentioned above, we apply peak-picking with three types of threshold settings, including a global (GLB) threshold, an adaptive local (LOC) threshold, and an oracle threshold, respectively.

6.2.2 Effects of peak-picking thresholds

Table 2 presents the beat-tracking results of pretrained single-modality models (ABTs and SBTs) derived using three types of peak-picking settings (GLB, LOC, and oracle).[8] We specify the hyperparameters using a dash symbol; for example, GLB-0.1 indicates a global threshold with $\tau = 0.1$, and LOC-20 indicates a local threshold using a window length of 20 seconds. The following observations can be made: Under the GLB threshold setting, the performances of ABTs and SBTs across all evaluation metrics show high sensitivity to the threshold values. For example, increasing the global threshold from 0.25 to 0.5 reduces the *F*1 score for ABTs from 0.886 to 0.660 (−0.226) and for SBTs from 0.775 to 0.662 (−0.113). Using the oracle threshold setting reveals upper bound *F*1 scores for ABT (0.892) and SBT (0.855). Under the LOC threshold setting, while the achieved *F*1 scores may be slightly below the GLB best case (e.g., for ABT, GLB-0.25: 0.886 vs. LOC-5: 0.835), they remain comparable and exhibit substantially lower sensitivity to hyperparameter settings. Specifically, varying the window length of LOC threshold setting has only a small impact across all evaluation metrics.

Threshold	*F*-measure			*L*-correct		
	F1	P	R	*F-L2*	*F-L3*	*F-L4*
			Audio beat trackers (ABTs)			
GLB-0.01	0.757±0.054	0.632±0.071	**0.968**±0.020	0.549±0.116	0.476±0.130	0.409±0.140
GLB-0.1	0.825±0.053	0.730±0.074	0.963±0.021	0.698±0.101	0.639±0.119	0.565±0.140
GLB-0.25	**0.886**±0.048	0.877±0.056	0.901±0.053	**0.831**±0.080	**0.795**±0.098	**0.746**±0.125
GLB-0.5	0.660±0.108	**0.955**±0.037	0.518±0.118	0.487±0.160	0.372±0.174	0.284±0.161
Oracle	0.892±0.045	0.866±0.061	0.923±0.044	0.842±0.071	0.809±0.089	0.759±0.120
LOC-5	0.835±0.047	0.748±0.066	0.958±0.024	0.745±0.080	**0.696**±0.098	**0.628**±0.121
LOC-10	**0.840**±0.048	**0.755**±0.067	0.958±0.024	**0.746**±0.082	**0.696**±0.100	0.627±0.124
LOC-20	0.838±0.048	0.751±0.068	**0.960**±0.024	0.737±0.084	0.686±0.102	0.616±0.125
			Symbolic beat trackers (SBTs)			
GLB-0.01	0.823±0.045	0.741±0.064	**0.937**±0.038	0.700±0.092	0.606±0.123	0.497±0.148
GLB-0.1	**0.836**±0.056	0.877±0.064	0.804±0.072	**0.723**±0.101	**0.648**±0.129	**0.568**±0.149
GLB-0.25	0.775±0.070	0.913±0.057	0.679±0.091	0.589±0.135	0.488±0.161	0.400±0.175
GLB-0.5	0.662±0.092	**0.935**±0.052	0.522±0.107	0.371±0.163	0.258±0.174	0.201±0.168
Oracle	0.855±0.048	0.845±0.072	0.870±0.052	0.767±0.084	0.698±0.114	0.621±0.138
LOC-5	0.841±0.056	**0.915**±0.055	0.782±0.069	0.745±0.100	0.676±0.131	0.602±0.152
LOC-10	0.842±0.056	**0.915**±0.055	0.783±0.068	0.746±0.099	0.677±0.129	0.604±0.152
LOC-20	**0.844**±0.055	**0.915**±0.055	**0.786**±0.067	**0.750**±0.097	**0.682**±0.128	**0.610**±0.150

Table 2 Work-wise average beat-tracking results for pretrained models. (Top) `madmom`-based audio beat trackers. (Bottom) `PM2S`-based symbolic beat trackers. The best results are highlighted in bold. GLB: global; LOC: local.

Moreover, as mentioned in Section 5.2, selecting an optimal global threshold involves a precision–recall trade-off that can obscure the models' original behaviors. For instance, adjusting the global threshold from 0.01 to 0.5 raises ABT precision from 0.632 to 0.955 (+0.333) but decreases recall from 0.968 to 0.518 (−0.450). Opposed to GLB, the LOC method determines the threshold on the basis of the average amplitude within each window, thus compensating for local changes in intensity or dynamics. Under this LOC setting, differences between ABTs and SBTs become more apparent: ABTs tend to achieve higher recall, while SBTs generally attain higher precision, highlighting the potential complementarity between the two methods based on different modalities and concepts. Lastly, the adaptation capability of LOC also stabilizes the *L*-correct results. When *L* increases from 2 to 4, the GLB-based methods often show a dramatic drop in *F*1 scores. For example, for SBT, GLB-0.01 decreases from 0.700 to 0.497 (−0.203) and GLB-0.1 decreases from 0.723 to 0.568 (−0.155), while LOC-20 shows a slightly smaller decrease, from 0.750 to 0.610 (−0.140). This may be because expressive music often demonstrates dramatic musical changes (e.g., in volume or tempo), which fail the global settings (since any single error can break the continuity of *L*-correct). However, these dramatic changes may remain locally consistent, allowing adaptive methods to partially capture them.

In summary, we can see that, despite being simple and free from strong assumptions, the use of GLB still requires a careful selection of the threshold, which could potentially obscure the properties of activation functions. We therefore suggest LOC as a neutral post-processor to better reflect the behavior of the activation functions.

6.2.3 Effects of late fusion

With the overall good performance using LOC and its stability across the hyperparameter *W*, we now fix the peak-picking setting to LOC with a window length of 20 seconds (LOC-20) and investigate the behavioral differences between single-modality beat trackers and late fusion-based models. Table 3 (top) presents the work-wise average beat-tracking results for audio recordings of Maz-5. From the *F*1 scores, we can observe improvements made by both addition-based ($A + S$) and multiplication-based ($A \times S$) methods. For example, the *F*1 score increases from 0.844 for *S* to 0.885 for $A + S$ and to 0.850 for $A \times S$.

Activation	*F*-measure			*L*-correct		
	F1	P	R	*F-L2*	*F-L3*	*F-L4*
			Pretrained			
A	0.838±0.048	0.751±0.068	**0.960**±0.024	0.737±0.084	0.686±0.102	0.616±0.125
S	0.844±0.055	0.915±0.055	0.786±0.067	0.750±0.097	0.682±0.128	0.610±0.150
A+S	**0.885**±0.044	0.838±0.063	0.947±0.027	**0.823**±0.071	**0.790**±0.086	**0.744**±0.109
A×S	0.850±0.051	**0.959**±0.035	0.765±0.067	0.749±0.093	0.684±0.121	0.610±0.142
			Retrained			
A	0.816±0.038	0.716±0.054	**0.965**±0.014	0.696±0.068	0.606±0.087	0.468±0.100
S	0.927±0.037	0.919±0.042	0.938±0.037	0.902±0.053	0.882±0.066	0.858±0.082
A+S	0.860±0.036	0.786±0.054	0.962±0.015	0.772±0.064	0.712±0.079	0.625±0.107
A×S	**0.937**±0.034	**0.943**±0.031	0.933±0.041	**0.917**±0.047	**0.899**±0.059	**0.882**±0.069

Table 3 Work-wise average of beat-tracking results (including late-fusion approaches). Beat-tracking results (including late-fusion approaches) were derived using peak-picking with local average threshold with a window length of 20 seconds (LOC-20). The best results are highlighted in bold.

Specifically, $A+S$ and $A\times S$ represent two ways to leverage the complementarity between A and S: recall-oriented and precision-oriented, respectively. While $A \times S$ retains only the activation peaks agreed upon by both A and S, leading to much higher precision than A and S ($A \times S$: 0.959; A: 0.751; S: 0.915), $A + S$ retains all strong activation peaks from either A or S, leading to a high recall ($A+S$: 0.947; A: 0.960; S: 0.786).

Figure 4 (left) visualizes the behaviors of all these pretrained models. Comparing the musical score as a reference, we can observe the different behaviors between the four methods: When there are multiple note onsets close together (e.g., around the frame 4700 and the frame 4800, between the last beat of measure 40 and the second beat of measure 41 in the score), A and S generate different peaks in the activation function that do not correspond to the annotations. As these non-beat peaks are weak, the addition, smoothing, and normalization process of addition-based fusion can remove them, improving the performance of $A + S$. On the other hand, $A \times S$ eliminates the non-beat peaks around frame 4700 via the multiplication process. Figure 4 (left) also explains why the L-correct scores can be largely improved for $A+S$ in Table 3: While $A\times S$ will fail if either A or S makes a false-negative prediction, breaking the continuity required by L-correct, $A + S$ will retain strong activation peaks from either A or S while smoothing out weak predictions.

6.3 RETRAINED BEAT TRACKERS

The pretrained models were trained on datasets independent of Maz-5; for example, the `madmom` BiLSTM was primarily trained on pop, rock, and dance music, while the `PM2S` CRNN was trained on classical piano music by various composers, excluding Maz-5. In the next experiment, we retrained the same models (i.e., BiLSTM and CRNN) from scratch using Maz-5 with five-fold cross-validation. This approach not only enables model evaluation with limited data but also ensures that all performances (versions) in the Maz-5 dataset appear at least once in both the training and test sets. Specifically, we split the five Mazurkas into five folds. For each split, three Mazurkas are used for training, one for validation, and one for testing. In general, retraining allows us to inspect the capability of a model to adapt to a specific music scenario (e.g., Chopin Mazurkas). In this section, we describe our retraining methods, conduct experiments similar to those for the pretrained models, and discuss the consistencies and differences between the pretrained and retrained results.

6.3.1 Activity estimators and post-processors

Taking the retrained single-modality models, we apply them to the corresponding test split to derive the activations. We then apply similar late-fusion method and unified post-processing (as those used for the pretrained models) to the activations to derive beat estimations for further analysis.

6.3.2 Effects of peak-picking thresholds

Figure 5 visualizes and summarizes the observed effects of peak-picking thresholds. Despite that fact that both pretrained (Figure 5a) and retrained (Figure 5b) models show a strong dependency on GLB threshold values (solid curves), this dependency is less pronounced

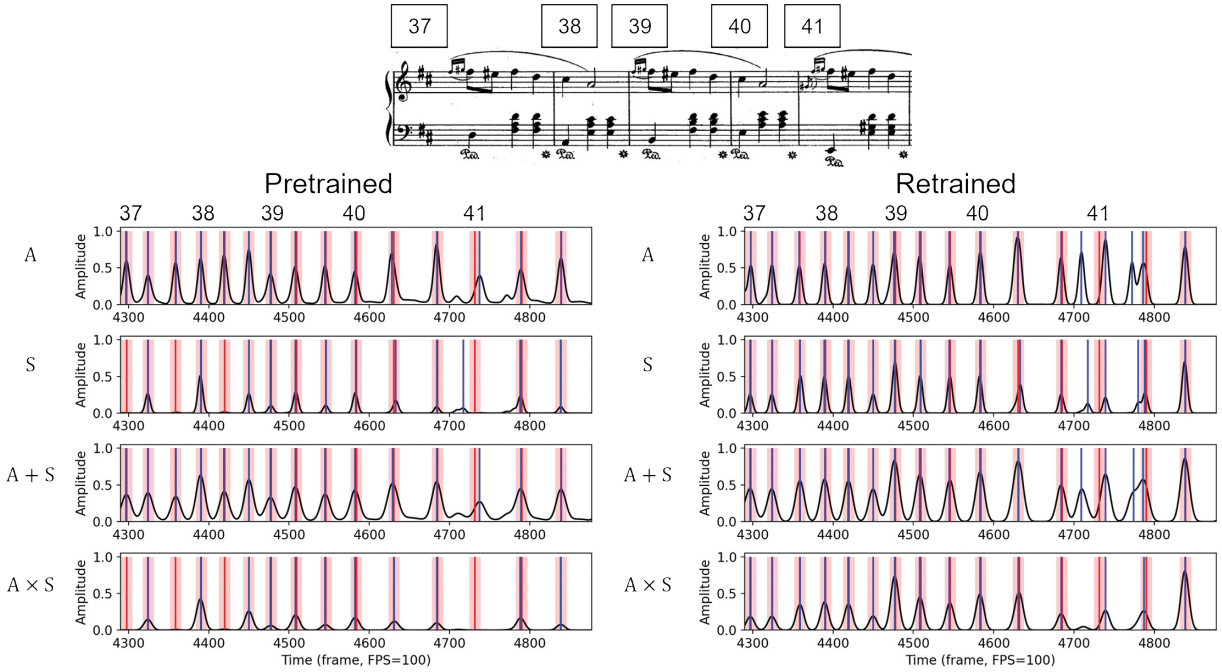

Figure 4 Comparison of four types of activations. **(top)** Music score of Op. 30, No.2. **(left)** Activation functions from pretrained models. **(right)** Activation functions from retrained models. Red regions highlight the ±70 ms tolerance window. Blue vertical lines indicate the beat estimations derived using peak-picking with the LOC-20 threshold setting. Op.: Opus.

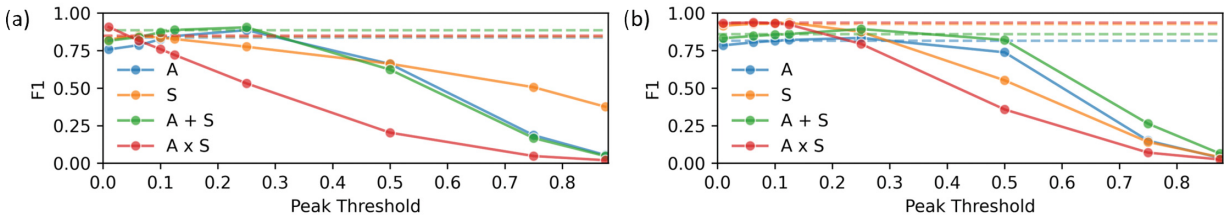

Figure 5 Effects of peak-picking thresholds on beat-tracking *F*1 scores. **(a)** Pretrained models. **(b)** Retrained models. Dashed lines indicate the *F*1 scores of the corresponding results derived using local average threshold (LOC-20). Solid dots indicate the *F*1 scores derived using global threshold values $\tau \in \{0.01, 0.0625, 0.1, 0.125, 0.25, 0.5, 0.75, 0.875\}$.

for the retrained models, particularly for small thresholds. Furthermore, compared with the pretrained models, the retrained models show an overall improvement in *F*1 scores, along with shifts in optimal global thresholds (e.g., pretrained *S*: 0.0625; retrained *S*: 0.125). This again highlights the challenge of identifying the optimal global threshold in practical applications. However, the local threshold-based results (dashed lines), though not the best, generally perform close to the best case of GLB settings.

6.3.3 Effects of training data

Going beyond Figure 5, Table 3 reveals further differences between pretrained (top) and retrained (bottom) models. While the retrained ABT behaves similarly to a pretrained ABT (retrained *F*1: 0.816; pretrained *F*1: 0.838), the retrained SBT achieves much higher scores for all metrics compared with the pretrained SBT (e.g., retrained *F*1: 0.927; pretrained *F*1: 0.844), indicating that the event-based CRNN model may have higher capability to learn (or, in some way, overfit to) the musical patterns in Maz-5. Figure 4 illustrates the differences between the pretrained (left) and retrained (right) models. For audio activations (*A*), both pretrained and retrained BiLSTMs show a tendency to detect onsets and generate activation peaks at non-beat positions (e.g., between frames 4700 and 4800). However, the retrained BiLSTM produces sharper, higher peaks, suggesting greater sensitivity to Maz-5 note events. For symbolic activations, both pretrained and retrained CRNNs produce less pronounced activation peaks compared with the BiLSTMs (e.g., around frames 4700–4800), yet the retrained CRNN appears to capture different musical patterns and retains more accurate peaks than the pretrained version (e.g., around frames 4350–4450).

6.3.4 Effects of late fusion

The improvement in retrained SBT noted above also influences the effects of late-fusion methods. In Table 3 (bottom), we see that addition-based fusion ($A + S$) no longer

outperforms SBT (*S*), whereas multiplication-based fusion ($A \times S$) still does. This shift is primarily due to two factors. First, the recall of retrained *S* increases significantly, from 0.786 to 0.938, which also substantially boosts the recall of retrained $A \times S$ (from 0.765 to 0.933). Second, the retrained *A* appears more sensitive to Maz-5 note events, generating stronger peaks at non-beat positions, which reduces its precision (from 0.751 to 0.716) and, in turn, lowers the precision of retrained $A + S$ as well. Additionally, late-fusion methods prove to be less effective than in the pretrained case. Specifically, as retrained *S* becomes much stronger and retrained *A* slightly weaker, the reduced complementarity between the two modalities limits the benefits of fusion. In summary, similar to our findings regarding optimal global threshold, the optimal fusion method is also dependent on the model and the data, which are mostly unknown for real use cases. This raises the question of whether DL-based methods can learn a more effective approach to fusing the two modalities based on the provided musical content, which is beyond the scope of this article.[9]

6.4 INSIGHTS FROM MAZ-5 SCENARIO

We further color-code the *F*1 scores of Maz-5, as shown in Figure 6. One can observe that the effectiveness of all models is version-dependent; i.e., they depend on the properties of specific versions. For example, versions #31, #40, #50, and #71 of M63-3 are consistently more challenging for all models compared with other versions. A possible explanation is the poor audio quality of the recordings and the varied interpretations by pianists (e.g., left-hand onsets may intentionally not align with right-hand onsets). Despite the version dependency, one can still observe certain trends of the average (right side after the black vertical line). Consistent with our observations in Table 3, for the pretrained cases, $A + S$ typically works better than *A* and *S*. Interestingly, in the retrained case, the improvement of *S* becomes more evident in the color-coded plot, shown as a horizontal blue line (for M63-3) that contrasts with the original results. Aside from cases of poor audio quality (e.g., version #50), which introduce errors during transcription, retrained *S* appears to adapt well to the Maz-5 scenario and to overcome certain version-wise challenges (e.g., for M63-3, retrained *S* performs well on versions #35, #36, and #71–73, which are challenging for all pretrained models). Lastly, one can also observe the challenges posed by musical properties by comparing across the five Mazurkas. For example, M24-2 and M30-2 are noticeably easier for all models. This is partly because the number of non-beat note events is lower in these Mazurkas. In summary, the cross-version nature of the Maz-5 scenario, combined with the color-coded plot, not only aligns with our previous observations but also offers deeper insights into version dependency, adaptation to specific music scenarios, and model limitations (e.g., challenges in handling low-quality recordings).

7 TOWARD DOWNBEAT TRACKING

We now repeat the experiments for downbeat tracking to assess whether the insights gained from beat tracking extend to downbeats, which rely on higher-level metrical structures rather than just local onset information. While beat tracking primarily detects rhythmic pulses, downbeat tracking requires understanding phrase structures, harmonic cues, and long-term dependencies, making it a substantially more difficult task. Rather than aiming to improve downbeat estimation, this experiment explores the potential of fusing different modalities for this challenging problem.

Using the same experimental pipeline, we evaluate downbeat activations from pretrained and retrained audio beat trackers (*A*) and symbolic beat trackers (*S*), as well as their fusion variants ($A + S$ and $A \times S$). As shown in Table 4, the pretrained models perform poorly in downbeat tracking, with all *F*1 scores below 0.500 and *L*-correct measures under 0.100, indicating low quality in capturing the continuity of downbeat estimations. While the pretrained *A* model maintains high recall (0.897), the pretrained *S* model does not retain its precision advantage from beat tracking (0.309 vs. 0.915 in Table 3), suggesting that downbeat tracking is a task that is more complex for and less effectively learned by these models. Given these limitations, late fusion provides little improvement.

Retraining the models follows a similar trend as in beat tracking. The retrained *A* model exhibits greater sensitivity to note events in Maz-5, improving recall (from 0.897 to 0.961) but at the cost of reduced precision (from 0.301 to 0.254). The retrained *S* model shows substantial improvements in both precision (from 0.309 to 0.561) and recall (from 0.744 to 0.814), significantly boosting *L*-correct scores (from below 0.100 to above 0.390). As in beat tracking, the precision-oriented fusion method ($A \times S$) further enhances the *F*1 score (from 0.657 to 0.671) by slightly lowering recall (from 0.814 to 0.791) while improving precision (from 0.561 to 0.591).

Overall, late fusion is beneficial only when both modalities provide meaningful and complementary outputs. The strong improvements in retrained *S* for both tasks suggest that event-based representations with explicitly defined note events offer a more effective approach for classical music, capturing musical structure and phrasing more effectively than audio-based methods alone.

8 CONCLUSION

In this work, we developed a cross-modal approach to beat tracking, aiming to deepen our understanding

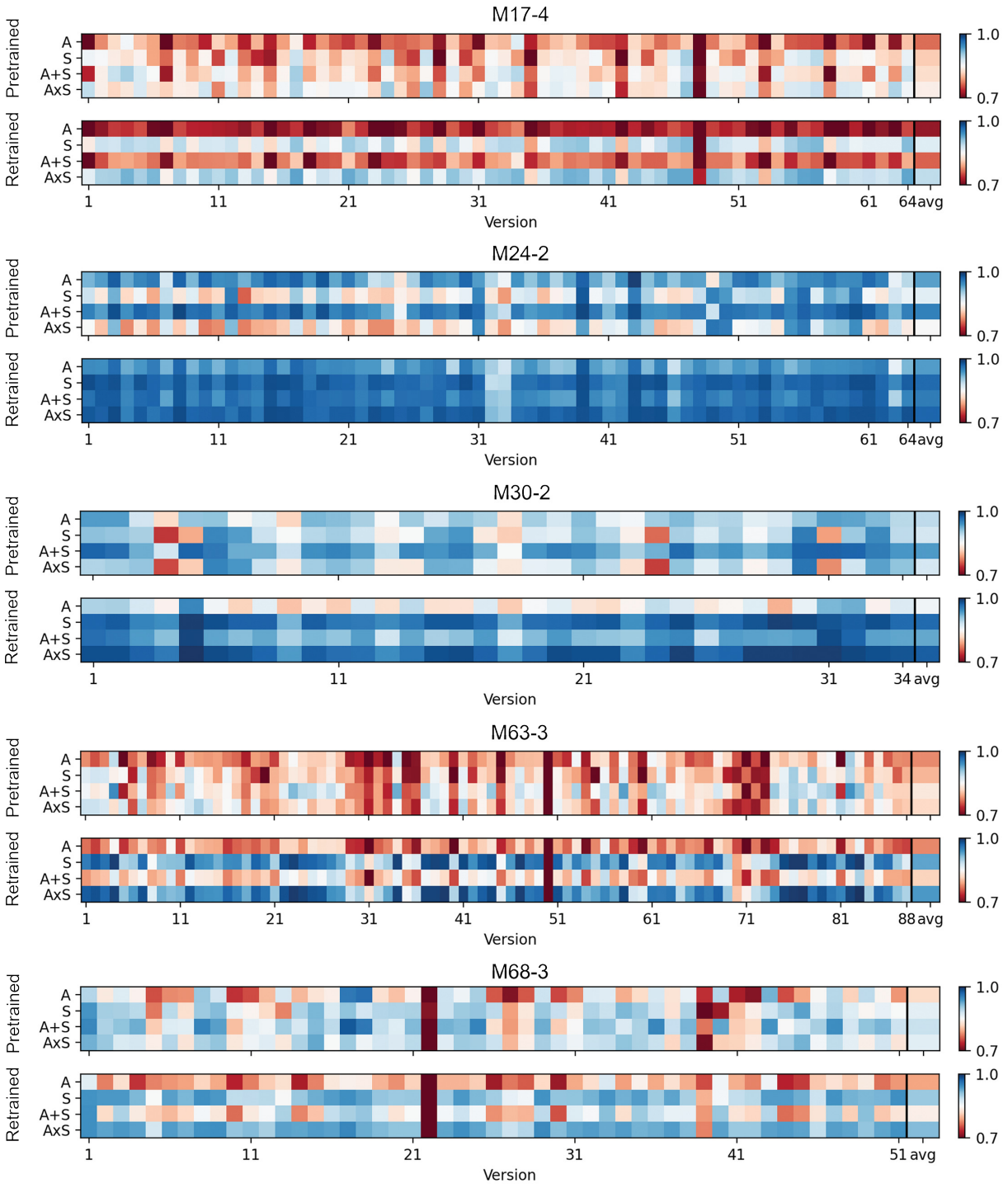

Figure 6 Beat-tracking *F*1 scores of Maz-5.

of audio and symbolic beat trackers while exploring their complementarity through fusion methods. Using a dataset of Chopin Mazurkas (Maz-5), we evaluated pretrained models in a controlled musical scenario and introduced a unified post-processing pipeline to enable direct comparisons, mitigating confounding effects from conventional post-processors. Retraining the models provided insights into their adaptability to Maz-5, revealing that symbolic-based methods benefit more from domain-specific training. Our exploration of late-fusion strategies demonstrated how combining audio and symbolic modalities can enhance performance by balancing precision and recall. To extend our analysis, we also investigated downbeat tracking to assess whether insights from beat tracking generalize to this more complex task. While retraining and fusion exhibited similar trends in both beat and downbeat tracking, the overall performance metrics were notably lower in the latter, highlighting the additional challenges of downbeat tracking. Beyond these findings, our work provides a structured

Activation	*F*-measure			*L*-correct		
	F1	P	R	*F-L2*	*F-L3*	*F-L4*
			Pretrained			
A	0.450±0.039	0.301±0.030	**0.897**±0.061	0.019±0.013	0.008±0.005	0.007±0.002
S	0.435±0.049	0.309±0.036	0.744±0.093	0.027±0.021	0.013±0.014	0.010±0.010
A+S	**0.459**±0.040	0.312±0.029	0.876±0.069	0.019±0.012	0.008±0.006	0.007±0.002
A×S	0.448±0.064	**0.347**±0.050	0.640±0.105	**0.084**±0.047	**0.032**±0.028	**0.018**±0.019
			Retrained			
A	0.401±0.026	0.254±0.020	0.961±0.029	0.010±0.004	0.005±0.001	0.005±0.001
S	0.657±0.071	0.561±0.072	0.814±0.080	0.480±0.100	0.432±0.102	0.396±0.101
A+S	0.434±0.026	0.281±0.021	**0.971**±0.025	0.016±0.007	0.008±0.004	0.007±0.003
A×S	**0.671**±0.073	**0.591**±0.072	0.791±0.082	**0.519**±0.103	**0.471**±0.109	**0.433**±0.111

Table 4 Work-wise average of downbeat-tracking results (including late-fusion approaches). Downbeat-tracking results (including late-fusion approaches) were derived using peak-picking with local average threshold with a window length of 20 seconds (LOC-20). The best results are highlighted in bold.

framework for analyzing beat tracking across different input modalities. Future research could further refine fusion strategies and explore broader datasets to enhance the robustness of rhythm analysis techniques.

FUNDING INFORMATION

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) within the project "Learning with Music Signals: Technology Meets Education" under Grant No. 500643750 (MU 2686/15-1) and the Emmy Noether research group "Computational Analysis of Music Audio Recordings: A Cross-Version Approach" under No. 531250483 (WE 6611/3-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institute for Integrated Circuits IIS.

COMPETING INTERESTS

The authors have no competing interests to declare.

AUTHOR CONTRIBUTION

Ching-Yu Chiu was the primary contributor and lead author of this article. Ching-Yu Chiu and Lele Liu jointly implemented the fusion approach and conducted the experiments. Meinard Müller and Christof Weiß supervised the work and contributed to the writing. All authors were actively involved in developing the conceptual framework and preparing the final manuscript.

NOTES

1. In general, every evaluation metric has its limitations and provides insights from specific perspectives. While the *F*-measure is simple and concise, it does not reflect a model's ability to make continuous correct estimations (Davies et al., 2021; 2009). CML_t and AML_t (Davies et al., 2009), designed to evaluate models in a continuity-based manner, require at least two consecutive correct estimations. In contrast, *L*-correct (Grosche and Müller, 2011) allows for manual adjustment of the continuity length criterion. Moreover, these metrics do not account for metric-level switching behaviors in beat-tracking models. For a more detailed discussion of the potential inconsistencies and limitations of existing metrics, readers can refer to Chiu et al. (2022).
2. While the use of a single dataset indeed limits the generalizability of this study, analyzing additional datasets would introduce considerable complexities (e.g., changes in time signatures or underlying inconsistencies between musical styles that may influence model learning), requiring discussion beyond the scope of this work. As our goal is to understand existing models in a carefully controlled setting rather than propose an approach aimed at significant performance improvements, we focus on Maz-5 in this study.
3. Given the space limitations and the relatively limited insights gained, we omit the event-based late-fusion experiments in this paper and report the results in our GitHub repository: https://github.com/SunnyCYC/CrossModalBeat.
4. The conversion from a frame-based to an event-based representation involves more technical decisions and may result in greater information loss. Therefore, we adhere to the frame-based representation, as it is more fundamental.
5. The Mazurka Project: http://www.mazurka.org.uk, 2010
6. In Liu et al. (2022), the model is referred to as Performance MIDI-to-Score (`PM2S`), which consists of both an activity estimator and a rule-based post-processor. In this work, we use only the activity estimator and refer to it as `PM2S`.
7. Note that there are more advanced local maxima-based normalization methods to further improve the tracking performance of the activation function (Nunes et al., 2015). However, as our goal is to understand rather than to improve the activation function, we consciously avoid complicated processing methods or fine-tuning of the processing parameters.
8. In this article, we omit the results and discussions of conventional post-processors such as DBN (Böck et al., 2016a; Krebs et al., 2015) and DP (Ellis, 2007; McFee et al., 2015) owing to the confounding effects introduced by their strong tempo assumptions and their poor performance (all *F*1 scores below 0.750; see our GitHub repository for details). For a

detailed discussion on the failure of DBN and DP on the Maz-5 dataset and how they may hinder the properties of the activation functions, readers may refer to Chiu et al. (2023).

9. Preliminary results of training-based fusion can be found in our GitHub repository: https://github.com/SunnyCYC/CrossModalBeat.

AUTHOR AFFILIATIONS

Ching-Yu Chiu
International Audio Laboratories Erlangen, Germany

Lele Liu
Center for Artificial Intelligence and Data Science, University of Würzburg, Germany

Christof Weiß
Center for Artificial Intelligence and Data Science, University of Würzburg, Germany

Meinard Müller
International Audio Laboratories Erlangen, Germany

REFERENCES

Böck, S., and Davies, M. E. P. (2020). Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 574–582).

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., and Widmer, G. (2016a). madmom: A new Python audio and music signal processing library. In *Proceedings of the ACM International Conference on Multimedia (ACM-MM)*, Amsterdam, The Netherlands (pp. 1174–1178).

Böck, S., Krebs, F., and Widmer, G. (2016b). Joint beat and downbeat tracking with recurrent neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, New York City, New York, USA (pp. 255–261).

Chang, C., and Su, L. (2024). Beast: Online joint beat and downbeat tracking based on streaming transformer. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 396–400).

Cheng, T., and Goto, M. (2023). Transformer-based beat tracking with low-resolution encoder and high-resolution decoder. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 466–473).

Chiu, C.-Y., Müller, M., Davies, M. E. P., Su, A. W.-Y., and Yang, Y.-H. (2022). An analysis method for metric-level switching in beat tracking. *IEEE Signal Processing Letters*, *29*, 2153–2157.

Chiu, C.-Y., Müller, M., Davies, M. E. P., Su, A. W.-Y., and Yang, Y.-H. (2023). Local periodicity-based beat tracking for expressive classical piano music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *31*, 2824–2835.

Chuang, Y.-C., and Su, L. (2020). Beat and downbeat tracking of symbolic music data using deep recurrent neural networks. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)* (pp. 346–352). IEEE.

Davies, M. E. P., Böck, S., and Fuentes, M. (2021). Tempo, beat and downbeat estimation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.

Davies, M. E. P., Quintela, N. D., and Plumbley, M. (2009). Evaluation methods for musical audio beat tracking algorithms. In *Queen Mary University of London, Centre for Digital Music, Tech. Rep.* C4DM-TR-09-06.

Ellis, D. P. (2007). Beat tracking by dynamic programming. *Journal of New Music Research*, *36*(1), 51–60.

Foscarin, F., McLeod, A., Rigaux, P., Jacquemard, F., and Sakai, M. (2020). ASAP: A dataset of aligned scores and performances for piano transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 534–541).

Foscarin, F., Schlüter, J., and Widmer, G. (2024). Beat this! Accurate beat tracking without DBN postprocessing. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, San Francisco, CA, United States.

Fuentes, M., McFee, B., Crayencour, H. C., Essid, S., and Bello, J. P. (2018). Analysis of common design choices in deep learning systems for downbeat tracking. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 106–112).

Fuentes, M., McFee, B., Crayencour, H. C., Essid, S., and Bello, J. P. (2019). A music structure informed downbeat tracking system using skip-chain conditional random fields and deep learning. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 481–485).

Grosche, P., and Müller, M. (2011). Extracting predominant local pulse information from music recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, *19*(6), 1688–1701.

Grosche, P., Müller, M., and Sapp, C. S. (2010). What makes beat tracking difficult? A case study on Chopin Mazurkas. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 649–654).

Hung, Y., Wang, J., Song, X., Lu, W. T., and Won, M. (2022). Modeling beats and downbeats with a time-frequency transformer. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 401–405).

Kong, Q., Li, B., Song, X., Wan, Y., and Wang, Y. (2021). High-resolution piano transcription with pedals by regressing onset and offset times. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *29*, 3707–3717.

Krebs, F., Böck, S., and Widmer, G. (2015). An efficient state-space model for joint tempo and meter tracking. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 72–78).

Liu, L., Kong, Q., Morfi, V., and Benetos, E. (2022). Performance midi-to-score conversion by neural beat tracking. In

Proceedings of the International Society for Music Information Retrieval Conference (ISMIR) (pp. 395–402).

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O. (2015). Librosa: Audio and music signal analysis in Python. In *Proceedings the Python Science Conference*, Austin, Texas, USA (pp. 18–25).

Meier, P., Chiu, C.-Y., and Müller, M. (2024). A real-time beat tracking system with zero latency and enhanced controllability. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, *7*(1), 213–227.

Müller, M., and Chiu, C.-Y. (2024). A basic tutorial on novelty and activation functions for music signal processing. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, *7*(1), 179–194.

Nunes, L. O., Rocamora, M., Jure, L., and Biscainho, L. W. P. (2015). Beat and downbeat tracking based on rhythmic patterns applied to the uruguayan candombe drumming. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 264–270).

Pinto, A. S., Böck, S., Cardoso, J. S., and Davies, M. E. P. (2021). User-driven fine-tuning for beat tracking. *Electronics*, *10*(13), 1518.

Sapp, C. S. (2007). Comparative analysis of multiple musical performances. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Vienna, Austria (pp. 497–500).

Sapp, C. S. (2008). Hybrid numeric/rank similarity metrics. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Philadelphia, Pennsylvania, USA (pp. 501–506).

Schreiber, H., Zalkow, F., and Müller, M. (2020). Modeling and estimating local tempo: A case study on Chopin's mazurkas. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Montréal, Canada (pp. 773–779).

Schwarzhuber, T. (2024). Beat tracking on classical music: Audio vs symbolic representations. *Master's thesis*. University Linz. https://epub.jku.at/obvulihs/content/titleinfo/10031600

Shi, Z. (2021). Computational analysis and modeling of expressive timing in Chopin's Mazurkas. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 650–656).

Yamamoto, K. (2021). Human-in-the-loop adaptation for interactive musical beat tracking. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 794–801).

Zhao, J., Xia, G., and Wang, Y. (2022). Beat transformer: Demixed beat and downbeat tracking with dilated self-attention. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 169–177).

TO CITE THIS ARTICLE:
Chiu, C.-Y., Liu, L., Weiß, C., & Müller, M. (2025). Cross-Modal Approaches to Beat Tracking: A Case Study on Chopin Mazurkas. *Transactions of the International Society for Music Information Retrieval, 8*(1), 55–69. DOI: https://doi.org/10.5334/tismir.238

Submitted: 12 November 2024 **Accepted:** 1 April 2025 **Published:** 2 May 2025