BEAT AND DOWNBEAT TRACKING IN PERFORMANCE MIDI USING AN END-TO-END TRANSFORMER ARCHITECTURE

Sebastian MURGUL (sebastian.murgul@klang.io) [1,2] and
Michael HEIZMANN (michael.heizmann@kit.edu) [2]

[1] **Klangio GmbH**, Karlsruhe, Germany
[2] *Institute of Industrial Information Technology*, **Karlsruhe Institute of Technology**, Karlsruhe, Germany

ABSTRACT

Beat tracking in musical performance MIDI is a challenging and important task for notation-level music transcription and rhythmical analysis, yet existing methods primarily focus on audio-based approaches. This paper proposes an end-to-end transformer-based model for beat and downbeat tracking in performance MIDI, leveraging an encoder-decoder architecture for sequence-to-sequence translation of MIDI input to beat annotations. Our approach introduces novel data preprocessing techniques, including dynamic augmentation and optimized tokenization strategies, to improve accuracy and generalizability across different datasets. We conduct extensive experiments using the A-MAPS, ASAP, GuitarSet, and Leduc datasets, comparing our model against state-of-the-art hidden Markov models (HMMs) and deep learning-based beat tracking methods. The results demonstrate that our model outperforms existing symbolic music beat tracking approaches, achieving competitive F1-scores across various musical styles and instruments. Our findings highlight the potential of transformer architectures for symbolic beat tracking and suggest future integration with automatic music transcription systems for enhanced music analysis and score generation.

1. INTRODUCTION

Beat tracking aims to detect the underlying rhythmic grid within a musical performance [1]. This rhythmic grid consists of downbeats, beats, and tatum subdivisions. Downbeats refer to the first beat within a bar and therefore indicate the beginning of a new bar. In a notation-level music transcription system, we implicitly or explicitly need the rhythmical grid in order to be able to derive discrete note values from the detected tones [2]. In general, we have two input options in a transcription setting: We can either use the original input audio or the output performance MIDI data of a previous note tracking step.

Audio-based beat tracking has seen substantial progress, particularly with the introduction of deep learning techniques. Early approaches relied on statistical models, but modern methods increasingly leverage neural networks to improve accuracy and robustness. One of the first notable deep learning approaches was proposed by Böck et al. (2011), who pioneered the use of bidirectional Long Short-Term Memory (LSTM) networks to classify beats from an audio spectrogram, smoothing predictions using autocorrelation [3]. This work evolved in 2016 when Böck et al. proposed an RNN-based beat tracking method that outputs beat and downbeat features directly from magnitude spectrograms. A Dynamic Bayesian Network (DBN) was then used to model variable-length bars and align the detected beats [4]. Davies et al. enhanced Böck's approach in 2019 by replacing LSTMs with a Temporal Convolutional Network (TCN) featuring dilated convolutions along the temporal axis [5]. Zhao et al. introduced Beat Transformer in 2022, employing time-wise and instrument-wise attention mechanisms alongside dilated self-attention and demixed spectrograms to improve beat detection [6]. Foscarin et al. proposed Beat This! in 2024, an advanced transformer-based beat tracker demonstrating high accuracy and generality across diverse musical styles. Since Beat This! does not rely on DBN postprocessing, it is also suitable for pieces with time-signature changes or high tempo variations [7].

Unlike audio-based beat tracking, which has been extensively researched, MIDI-based methods have remained relatively scarce. Traditional approaches to MIDI-based beat tracking often relied on rule-based heuristics and statistical models, but recent research has begun incorporating deep learning techniques. Cambouropoulos et al. proposed a system for joint beat detection and rhythm quantization in 2000 [8]. Their approach clustered inter-onset intervals for beat detection, followed by assigning note onsets to the closest points on a metrical grid and assigning note values based on inter-onset intervals. Temperley's 2007 book 'Music and Probability'extended Bayesian probabilistic approaches to infer complete metrical grids rather than score positions relative to a bar [9]. Cogliati et al. presented an HMM-based system in 2016 for joint estimation of meter, harmony, and stream separation, combined with a distance-based quantization algorithm [10]. Foscarin et al. introduced a parse-based system in 2019 employing weighted context-free grammars (WCFGs) for joint rhythm quantization and music score production [11]. Shibata et al. proposed a piano transcription system in 2021 that incorporated HMMs and Markov Random Fields (MRFs) for rhythm quantization, leveraging non-local musical statistics to infer global parameters [12]. Liu et al. proposed a Convo-

lutional Recurrent Neural Network (CRNN)-based system in 2022 for MIDI-to-score conversion, incorporating onset-based beat detection and rhythm quantization [13]. Kim et al. developed a transformer and Convolutional Neural Network (CNN)-based guitar transcription model in 2023 that produced note-level transcriptions from spectrograms using beat information [14]. Beyer et al. introduced a performance MIDI-to-score conversion approach in 2024 based on the Roformer architecture. Their encoder-decoder model directly generated MusicXML tokens while implicitly performing beat estimation and rhythm quantization on MIDI token sequences [15].

Despite these efforts, modern transformer architectures have seen limited application in symbolic beat tracking. Existing methods either rely on traditional probabilistic models or use deep learning approaches not specifically optimized for MIDI beat tracking. This gap presents an opportunity to explore more advanced techniques for symbolic music.

In this work, we introduce a novel end-to-end transformer-based approach for beat tracking in MIDI performances, achieving state-of-the-art performance and surpassing existing symbolic beat tracking methods. By leveraging modern transformer architectures, our model effectively captures temporal dependencies and outperforms previous HMM-based and neural network-based systems.

2. METHODOLOGY

Our proposed approach for performance MIDI beat tracking is visualized in Figure 1. The model follows an encoder-decoder transformer architecture, designed to translate an input MIDI segment into the corresponding beat sequence. Since transformers operate on text-based token sequences, MIDI data must first be preprocessed and tokenized before being fed into the model. The following sections detail the preprocessing pipeline, data augmentation strategies, encoding schemes, and model architecture.

2.1 Preprocessing

The data processing is shown in the flow chart in Figure 2. As input, we use the MIDI files of the A-MAPS dataset. Firstly, we extract the notes and the beat annotations from the MIDI files using the PrettyMIDI library [17]. The extracted annotations are then split into segments of 10 s with a hop size of 1 s. In the next step, the segments are cleaned, and all examples with less than one beat are dropped. Finally, the resulting examples are stored in a CSV file for access during training.

2.2 Data Augmentation

The data augmentation is done dynamically in the data loading process during the training. To enrich the diversity of the examples, we perform pitch transposition and time manipulation. The transposition augmentation shifts all the pitches of the piece within a given pitch range from A_0 to C_8. The shift is drawn from a uniform distribution. This ensures that there is no bias for specific pitches in the training data. One main advantage of using symbolic input over audio in a beat tracking deep learning model is the ease of manipulating the temporal properties of the data in order to increase the covered range. For timing manipulation, we apply a random shift within the range of $[-1\,\text{s}, +1\,\text{s}]$ and a randomly drawn scaling factor between 0.9 and 1.1

2.3 Data Encoding

In order to be able to efficiently tokenize the MIDI and beat segments for use with a transformer model, the used vocabulary should be as slim as possible. Therefore, we first quantize the absolute time values to 10 ms steps. This leads to $1,000$ tokens for segments of 10 s length. The amount of time tokens can be further reduced by encoding relative instead of absolute time information, but our early experiments showed that this comes with the downside of error propagation and leads to overall worse results. To evaluate the effects of data encoding on the beat tracking performance, we conducted an experiment with four different encoding strategies. Each encoding was designed to capture the essential musical information, and we experimented with different levels of abstraction and granularity.

The input sequence of encoding *v1* uses ON⟨#⟩ tokens to denote the pitch of a note's onset at the absolute time which is denoted by the T⟨#⟩ token. Similarly, the target sequence uses B⟨#⟩ tokens to encode the beat counter value at the beat time T⟨#⟩. Here, B⟨1⟩ specifies the first beat, which corresponds to the downbeat. Version *v2* adds a note offset token OFF⟨#⟩ and *v3* also a token for the note's velocity value VEL⟨#⟩ to the input sequence. The input encoding in version *v4* is the same as in *v3* but the target encoding uses only downbeat DB and beat B tokens instead of an explicit beat counter. Lastly, encoding *v5* mirrors *v2* but adopts the target representation of *v4*, removing explicit beat counters while maintaining note offset encoding. This version seeks to balance structural clarity with a compact target format. Table 1 shows short examples of the input and target sequence for each of the text encoding versions.

2.4 Model

The model uses the T5 transformer architecture [16] since the architecture already proved to be suited in other Music Information Retrieval (MIR) tasks like transcription [18]. The Hugging Face Transformers package [19] is used for implementing the network. We employ a reduced architecture of the T5 model, halving the configuration of *t5-small* with a model dimension $d_{model} = 128$, feedforward dimension $d_{ff} = 1024$, three encoder-decoder layers and four attention heads.

This model is trained from scratch, utilizing the Adafactor optimizer [20] with a self-adaptive learning rate. The models are trained for 50 epochs with a batch size of 32. Training the model on an NVIDIA Tesla V100 takes about 6 hours on average.

For the autoregressive sampling during inference, we found out that using a beam search with 5 beams leads to the best results. We also specify that all ngrams of size 2 can only occur once.

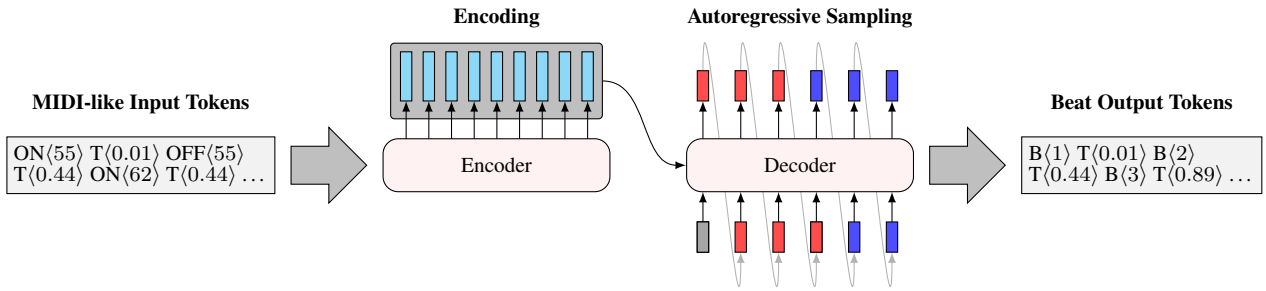

Figure 1. Our model is based on the T5 encoder-decoder transformer architecture [16] and uses MIDI-like tokens as input and outputs tokenized beat and downbeat information. During inference, autoregressive sampling with beam search is used.

ID	Input	Target
v1	ON⟨55⟩ T⟨0.01⟩ ON⟨62⟩ T⟨0.44⟩ ...	B⟨1⟩ T⟨0.01⟩ B⟨2⟩ T⟨0.44⟩ B⟨3⟩ T⟨0.89⟩ ...
v2	ON⟨55⟩ T⟨0.01⟩ OFF⟨55⟩ T⟨0.44⟩ ON⟨62⟩ T⟨0.44⟩ ...	B⟨1⟩ T⟨0.01⟩ B⟨2⟩ T⟨0.44⟩ B⟨3⟩ T⟨0.89⟩ ...
v3	ON⟨55⟩ T⟨0.01⟩ VEL⟨80⟩ OFF⟨55⟩ T⟨0.44⟩ ON⟨62⟩ T⟨0.44⟩ ...	B⟨1⟩ T⟨0.01⟩ B⟨2⟩ T⟨0.44⟩ B⟨3⟩ T⟨0.89⟩ ...
v4	ON⟨55⟩ T⟨0.01⟩ VEL⟨80⟩ OFF⟨55⟩ T⟨0.44⟩ ON⟨62⟩ T⟨0.44⟩ ...	DB T⟨0.01⟩ B T⟨0.44⟩ B T⟨0.89⟩ ...
v5	ON⟨55⟩ T⟨0.01⟩ OFF⟨55⟩ T⟨0.44⟩ ON⟨62⟩ T⟨0.44⟩ ...	DB T⟨0.01⟩ B T⟨0.44⟩ B T⟨0.89⟩ ...

Table 1. Examples of the different input MIDI-to-text and target beat-to-text encodings. ON / OFF define the pitch and T the time of the event. Additionally, the VEL token defines the velocity of the played note. The B and DB tokens define the beat and downbeat positions in the target sequence.

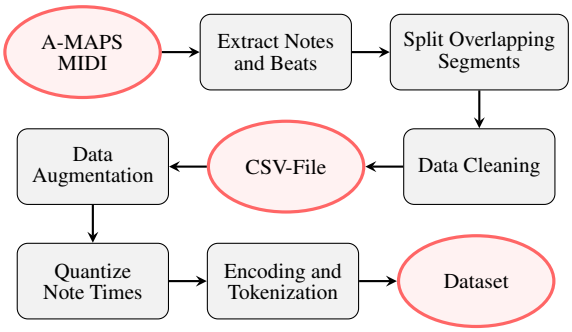

Figure 2. Flowchart of the data processing steps

3. EXPERIMENTS

In this section, we describe the datasets used for training and evaluation, followed by the metrics used to assess model performance.

3.1 Datasets

For the training and evaluation of the model, datasets containing synchronized MIDI and beat annotations are essential. The A-MAPS dataset [21] is an extension of the MAPS database [22] which consists of 270 piano pieces with audio and corresponding MIDI annotations. The original MAPS dataset has been widely used to train and evaluate piano transcription performance. The A-MAPS dataset augments the MIDI annotations by adding rhythm (including beat and downbeat positions) as well as key annotations. Because of the significant large size and well-aligned annotations, it is used as main dataset in our studies. The ASAP dataset [23] is a dataset of aligned musical scores and MIDI performances with additional beat and time signature annotations, amongst others. Therefore, it is used as a second piano dataset in Section 4.2. For a more diverse quantitative evaluation, we also look at the guitar datasets GuitarSet [24] and the Leduc dataset [25]. While GuitarSet comes with beat and downbeat annotations in the JAMS files, the orginal Leduc dataset is focused solely on the guitar transcription task. The Leduc dataset comprises 239 jazz guitar performances with accompanying high-quality transcriptions written by François Leduc in the GuitarPro [1] format. The scores are converted into MIDI and aligned with the original audio using the approach described by Riley et al. [26]. This alignment process is further refined to also adjust downbeat and beat information from the GuitarPro files according to the resulting note mapping between the score and the performance. Using this extension, we also get beat annotations alongside the aligned MIDI transcriptions. The created beat annotations are available online and can be used alongside the original Leduc dataset files [2].

3.2 Evaluation Metrics

To quantitatively assess model performance in both studies and comparative evaluations, we rely on F1-scores for beat and downbeat detection, along with continuity metrics.

- **Beat F1-score (f_b):** Measures the accuracy of detected beat positions relative to ground-truth annotations.
- **Downbeat F1-score (f_{db}):** Evaluates the model's ability to correctly identify downbeats, which mark the beginning of musical bars.

[1] https://www.guitar-pro.com
[2] https://github.com/klangio/midi-beat-tracking

We use the standard tolerance window of 70 ms for both beats and downbeats, as defined by the mir_eval Python library [27]. Unlike some previous studies, we do not exclude the first five seconds of each sequence before evaluation, as our model operates on relatively short input sequences.

4. RESULTS

This section presents the results of our proposed method, evaluated using the experimental setup described in Section 3. We first analyze the impact of various hyperparameters and modeling choices in an ablation study, followed by a comparative evaluation against existing state-of-the-art approaches.

4.1 Ablation Study

In the ablation study, we show the effect of various hyperparameters, encoding schemes, and data augmentation strategies. Each model is trained for 50 epochs on the training split of the A-MAPS dataset, while the results are obtained by evaluating on the test split of the same dataset. If not specified otherwise, we use the T5 architecture with a segment length of 10 s, a quantization of 10 ms, *v3* encoding, and no data augmentations.

4.1.1 Data Encodings

The choice of the text encoding is crucial for the model's ability to capture the context and rules the beat tracking estimation implicitly underlies. In Table 2 we show the downbeat and beat tracking performance for the encoding schemes introduced in Section 2.3. It can be observed that the model benefits from the more information it receives as input. Adding velocity as well as offset information leads to an 18 % higher f_b score than when only relying on note onset information. This improvement aligns with the expectation that downbeats are typically emphasized, often through increased velocity values, making them more distinguishable. However, a key limitation of this approach is that velocity values are not always available, particularly for instruments like the guitar, which may reduce its applicability in certain contexts.

The best results for the downbeats are achieved with the *v4* encoding. Here, we observe a great impact by the removal of the beat counter. We assume that the model already counts the beats implicitly and that having to output a specific counter value leads to confusion when the MIDI segment is cropped in a way that it does not begin with the first beat. Interestingly, the f_b score drops slightly in comparison to the *v3* encoding. In version *v5*, on the other hand, we do not see that effect in f_db.

4.1.2 Segment Length

The choice of the segment length does not only have a direct impact on the sequence lengths the model has to process, but also affects the vocabulary size via the number of time tokens needed. Therefore, having a shorter segment length reduces complexity with the disadvantage of reducing the context window, too. Table 3 shows the results for different segment lengths. For the beat F1-score, the segment length of 10 s leads to the best results. Since the 15 s length performs better than the 5 s length, it can be assumed that the sweet spot for the aforementioned tradeoff lies somewhere in the interval $[10\,\mathrm{s}, 15\,\mathrm{s})$.

Encoding Scheme	f_b	f_db
v1	81.06 %	34.75 %
v2	90.11 %	47.69 %
v3	**96.03 %**	59.52 %
v4	94.84 %	**67.16 %**
v5	81.23 %	47.31 %

Table 2. Comparison of beat (f_b) and downbeat (f_db) F1-scores across different encoding schemes. The results highlight the impact of incorporating velocity, offset information, and the removal of the explicit beat counter on beat tracking performance.

Segment Length	f_b	f_db
5 s	91.77 %	**65.54 %**
10 s	**96.03 %**	59.52 %
15 s	94.99 %	57.10 %

Table 3. Impact of segment length on beat (f_b) and downbeat (f_db) F1-scores. The results illustrate the tradeoff between shorter segments, which reduce computational complexity, and longer segments, which provide a broader temporal context for beat tracking.

4.1.3 NLP Task Interpretation

Our implementation using the T5 transformer interprets the task of beat tracking as a translation between the MIDI language and the rhythm language. Alternatively, the text completion interpretation can also be applied by using the GPT2 model [28]. Here, the model is trained to generate beat tokens for a given sequence of MIDI tokens in a text completion manner. Therefore, a 'MIDI:' token followed by the MIDI notes and a 'Beat:' token is used as a primer sequence. The GPT2 model now completes the text by adding the beat token sequence. A comparison of the results from the GPT2 model with the T5 model is shown in Table 4. We can see that the T5 clearly outperforms the GPT2 model in terms of beat and downbeat F1-scores.

Model Architecture	f_b	f_db
T5	**96.03 %**	**59.52 %**
GPT2	88.69 %	45.22 %

Table 4. Comparison of beat (f_b) and downbeat (f_db) F1-scores for different NLP task interpretations. The results demonstrate the superiority of the T5 transformer model, which frames beat tracking as a translation task, over the GPT-2 model, which treats it as a text completion problem.

4.1.4 Effect of Augmentation

One of the main advantages of using symbolic MIDI input over audio is that it is fairly easy and efficient to augment the examples dynamically during training. In this experiment, we evaluate the effect of different augmentation methods for pitch and time. As shown in Table 5, the transposition of the notes' pitches leads to a minor improvement of both downbeat and beat F1-scores. Applying a randomly drawn time shift alone does not have a noticeable positive effect. Although, in combination with the time scaling augmentation, the best results can be achieved with an f_b of over 98 %. But this improvement comes at the price of a significantly decreased f_{db} score.

Augmentation Methods	f_b	f_{db}
None	96.03 %	59.52 %
Transpose	96.22 %	**59.91 %**
Transpose / Shift	96.06 %	58.67 %
Transpose / Shift / Scale	**98.10 %**	52.25 %

Table 5. Impact of different data augmentation strategies on beat (f_b) and downbeat (f_{db}) F1-scores. The results highlight the effectiveness of pitch transposition (Transpose), time shifting (Shift) and time scaling (Scale), while also showing the tradeoff between improved beat tracking accuracy and decreased downbeat performance.

4.1.5 Time Quantization

The temporal resolution has a significant impact on the vocabulary size and offers a tradeoff between precise beat time output and the balancing of the dataset. With a higher temporal resolution, the individual time tokens are less frequently present in the training dataset. Another effect of quantizing the time tokens is that this way beat annotation inaccuracies get reduced. Therefore, we highlight the results for different time quantization settings in Table 6. The evaluation shows that there is a noticeable impact on the beat F1-score and a big impact on the downbeat F1-score. By increasing the time steps to 50 ms, we get the best results for f_b with 2 % increase over the result for 10 ms. By increasing the time steps even to 100 ms, we see a 34 % increase in f_{db}. But these results should be seen with caution since they also benefit from the relatively loose tolerance of 70 ms used by mir_eval. By increasing the quantization step further to 200 ms, we see a rapid drop of both scores, which matches our expectation since the temporal resolution is too coarse for the evaluation tolerance window now.

Temporal Resolution	f_b	f_{db}
5 ms	94.52 %	54.36 %
10 ms	96.03 %	59.52 %
20 ms	96.98 %	65.17 %
50 ms	**97.88 %**	73.58 %
100 ms	97.68 %	**80.00 %**
200 ms	71.85 %	55.35 %

Table 6. Effect of different time quantization settings on beat (f_b) and downbeat (f_{db}) F1-scores. The results demonstrate how increasing the quantization step can enhance downbeat detection but also highlight the performance drop when the resolution becomes too coarse.

4.2 Comparison with Baselines

Using the results of the ablation study, we optimized our final model for a comparison with other state-of-the-art methods. We choose a segment length of 10 s, a temporal resolution of 50 ms and we apply all three augmentation methods. The model is trained on a combined training dataset containing the respective train splits of the A-MAPS, ASAP, GuitarSet, and Leduc datasets. Since velocity is not included in the guitar datasets, the encoding *v2* is used.

We compare our model's performance with two state-of-the-art MIDI beat tracking approaches: The strongest HMM-based approach [12] and the PM2S deep learning model [13] which relies on neural beat tracking. For reference, we also add the evaluation results of Beat This!, the currently best-performing audio-based beat tracking transformer model. The scores for Beat This! are directly taken from the 8-fold cross-validation results of the original paper and are therefore intended to give an impression rather than a fair comparison. The results of the comparative evaluation on the test splits of the four datasets are shown in Table 7.

Our proposed transformer-based model outperforms PM2S and the HMM on almost every dataset for beat as well as downbeat performance. Only on the ASAP dataset, the beat tracking results of PM2S are 6 % better compared to our results. In general, we can see a significant drop in downbeat accuracy when comparing the results for the A-MAPS dataset with the others. This is an indicator that the data is easier to learn, which is caused by the way the dataset has been generated (see Section 4.3). We also see a significant drop in f_b when comparing guitar with piano dataset results. This indicates that detecting beats in guitar music is more difficult, since we have generally a lower note density and less strong downbeat indication, especially in guitar solos. Since the pieces in the ASAP dataset consist of more complex time-signatures than in the A-MAPS dataset, we see a much higher drop in the downbeat than in the beat F1-score.

The audio-based method performs generally better in terms of downbeat F1-score, since a lot of expression gets lost when transcribed as a MIDI file. The accents that typically are placed on the downbeat make it a lot easier to determine the beats' positions. The huge differences in the GuitarSet dataset are most likely caused by the inaccuracies of the MIDI annotations and the discrepancy with the beat annotations (see Section 4.3). These might also be responsible for the higher F1-scores for the Leduc guitar transcription dataset. Although the Leduc dataset consists of Jazz pieces with more complex rhythms, the alignment between MIDI and beat annotations is better because of the joint beat and note alignment process described in Section 3.1.

Method	A-MAPS		ASAP		GuitarSet		Leduc	
	f_b	f_{db}	f_b	f_{db}	f_b	f_{db}	f_b	f_{db}
Beat This! [7]	-	-	76.30 %	61.20 %	92.20 %	88.10 %	-	-
HMM (J-Pop) [12]	48.63 %	25.62 %	47.68 %	13.36 %	38.37 %	6.78 %	32.71 %	13.38 %
HMM (classical) [12]	49.85 %	28.23 %	43.60 %	13.67 %	33.65 %	9.88 %	34.34 %	13.78 %
PM2S [13]	83.89 %	68.90 %	**82.95 %**	14.14 %	42.63 %	16.49 %	41.12 %	28.94 %
Ours	**98.01 %**	**76.56 %**	78.13 %	**27.81 %**	**52.38 %**	**23.02 %**	**57.72 %**	**29.75 %**

Table 7. Comparison of beat (f_b) and downbeat (f_{db}) F1-scores between the proposed transformer-based model and state-of-the-art MIDI beat tracking methods, including an HMM-based approach [12] and the PM2S deep learning model [13]. For reference, results from the audio-based Beat This! model are also included. The evaluation is conducted on the test splits of the A-MAPS, ASAP, GuitarSet, and Leduc datasets, highlighting differences in performance across piano and guitar music.

4.3 Discussion

While there are datasets available containing performance MIDI with beat annotations, the quality strongly varies. Since the A-MAPS MIDI files are derived from tempo-varied quantized MIDI files, they consequently do not capture the full range of human timing variations [23]. On the other hand, they offer a perfect alignment between beat and note annotations. The GuitarSet examples consist of actual guitar recordings that have been transcribed semi-automatically using a hexaphonic pickup and manual corrections [24]. While the note annotations have been carefully adjusted, the beat annotations come from the used metronome and do not account for any tempo variations by the human player. Datasets like Leduc and ASAP have an improved alignment workflow and lead to a better agreement between beat and note annotations.

5. CONCLUSION

This research demonstrates the effectiveness of an end-to-end transformer-based approach for beat tracking in MIDI performances. By formulating the task as a symbolic translation problem, our model surpasses existing MIDI-based methods, including HMM-based approaches and PM2S, achieving state-of-the-art performance across most datasets. Key contributions include a data pre-processing pipeline, data augmentations, and tokenization strategies. The model's adaptability to diverse datasets for guitar and piano highlights its generalizability. Augmentation strategies and temporal quantization enhance the beat tracking accuracy. Although its accuracy does not yet match that of audio-based beat tracking methods, the model remains highly effective in scenarios where only MIDI data is available, offering a practical alternative for rhythm analysis in symbolic music.

In future works, this method could be combined with a AMT approach in order to have a more fair comparison with audio based beat-tracking methods. By combining this method with a beat based quantization method, it could be used in a processing pipeline that transcribes audio to sheet music. Expanding the model to multi-instrument performances and applying self-supervised learning techniques could also increase its robustness and versatility in diverse musical settings.

6. REFERENCES

[1] M. E. P. Davies, S. Böck, and M. Fuentes, *Tempo, Beat and Downbeat Estimation*. https://tempobeatdownbeat.github.io/tutorial/intro.html, 2021.

[2] E. Benetos, S. Dixon, Z. Duan, and S. Ewert, "Automatic Music Transcription: An Overview," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 20–30, 2018.

[3] S. Böck and M. Schedl, "Enhanced Beat Tracking with Context-Aware Neural Networks," in *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx)*, 2011.

[4] S. Böck, F. Krebs, and G. Widmer, "Joint Beat and Downbeat Tracking with Recurrent Neural Networks," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[5] M. E. P. Davies and S. Böck, "Temporal Convolutional Networks for Musical Audio Beat Tracking," in *27th European Signal Processing Conference (EUSIPCO)*, 2019.

[6] J. Zhao, G. Xia, and Y. Wang, "Beat Transformer: Demixed Beat and Downbeat Tracking with Dilated Self-Attention," in *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR)*, 2022.

[7] F. Foscarin, J. Schlüter, and G. Widmer, "Beat This! Accurate beat tracking without DBN postprocessing," in *Proceedings of the 25th International Society for Music Information Retrieval Conference (ISMIR)*, 2024.

[8] E. Cambouropoulos, "From MIDI to Traditional Musical Notation," in *Proceedings of the AAAI Workshop on Artificial Intelligence and Music: Towards Formal Models for Composition, Performance and Analysis*, 2000.

[9] D. Temperley, *Music and Probability*. Mit Press, 2007.

[10] A. Cogliati, D. Temperley, and Z. Duan, "Transcribing Human Piano Performances Into Music Notation," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[11] F. Foscarin, F. Jacquemard, P. Rigaux, and M. Sakai, "A Parse-Based Framework for Coupled Rhythm Quantization and Score Structuring," in *Mathematics and Computation in Music*, 2019.

[12] K. Shibata, E. Nakamura, and K. Yoshii, "Non-Local Musical Statistics As Guides for Audio-To-Score Piano Transcription," *Information Sciences*, vol. 566, pp. 262–280, 2021.

[13] L. Liu, Q. Kong, G. Morfi, E. Benetos *et al.*, "Performance MIDI-To-Score Conversion by Neural Beat Tracking," in *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR)*, 2022.

[14] S. Kim, T. Hayashi, and T. Toda, "Note-Level Automatic Guitar Transcription Using Attention Mechanism," in *Proceedings of the 30th European Signal Processing Conference (EUSIPCO)*, 2022.

[15] T. Beyer and A. Dai, "End-to-End Piano Performance-MIDI To Score Conversion with Transformers," in *Proceedings of the 25th International Society for Music Information Retrieval Conference (ISMIR)*, 2024.

[16] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[17] C. Raffel and D. P. Ellis, "Intuitive Analysis, Creation and Manipulation of MIDI Data with pretty_midi," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[18] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. Engel, "Sequence-to-Sequence Piano Transcription with Transformers," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[19] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Hugging Face's Transformers: State-of-the-Art Natural Language Processing," *arXiv preprint arXiv:1910.03771*, 2019.

[20] N. Shazeer and M. Stern, "Adafactor: Adaptive Learning Rates with Sublinear Memory Cost," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4596–4604.

[21] A. Ycart and E. Benetos, "A-MAPS: Augmented MAPS Dataset with Rhythm and Key Annotations," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[22] V. Emiya, R. Badeau, and B. David, "Multipitch Estimation of Piano Sounds Using a New Probabilistic Spectral Smoothness Principle," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1643–1654, 2009.

[23] F. Foscarin, A. Mcleod, P. Rigaux, F. Jacquemard, and M. Sakai, "ASAP: A Dataset of Aligned Scores and Performances for Piano Transcription," in *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[24] Q. Xi, R. M. Bittner, J. Pauwels, X. Ye, and J. P. Bello, "GuitarSet: A Dataset for Guitar Transcription," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[25] D. Edwards, X. Riley, and S. Dixon, "The François Leduc Dataset," Apr. 2024. [Online]. Available: https://doi.org/10.5281/zenodo.10984521

[26] X. Riley, D. Edwards, and S. Dixon, "High Resolution Guitar Transcription via Domain Adaptation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024.

[27] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. Ellis, and C. C. Raffel, "MIR_EVAL: A Transparent Implementation of Common MIR Metrics," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[28] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.