



UK Research
and Innovation



PHD THESIS

Automatic Audio-to-Score Piano Transcription
with Deep Neural Networks

Lele Liu

Supervisors:

Dr. Emmanouil Benetos

Dr. Veronica Morfi

Prof. Simon Dixon

Independent Assessor:

Dr. Marcus Pearce

May 8, 2026

Centre for Digital Music

*Submitted in partial fulfilment of the requirements of the Degree of Doctor of
Philosophy*



Queen Mary
University of London

Statement of Originality

I acknowledge the ethical use of Generative Artificial Intelligence to support my polishing and proofreading in this thesis. I confirm that I have kept and can provide (if requested) detailed records of my input into Generative Artificial Intelligence tools, the outputs I received, and how I used these outputs. I used the following GenAI programme: ChatGPT.

I accept that Queen Mary University of London has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

The copyright of this thesis rests with the author and no quotation from it, or information derived from it may be published without the prior written consent of the author.

[Section 1.4](#) of [Chapter 1](#) describes the details of collaboration and publications related to this thesis.

Signature: Lele Liu

Date: May 8, 2026

Abstract

Automatic music transcription (AMT) is a core task in music information retrieval, aiming to convert audio recordings of musical performances into human- or machine-readable score formats. AMT has various applications, including music search, analysis, tutoring, and generation. While AMT research has traditionally focused on note-level transcription, i.e., generating mid-level representations such as piano rolls or note sequences, recent years have seen growing interest in score-level transcription. This seeks to produce a musical representation that includes not only notes but also rhythm, voice, and other score annotations.

In this thesis, we present our work on automatic audio-to-score transcription for piano performances. We begin by preparing two datasets, each representing a distinct musical style and level of expressive performance. We then explore two approaches: pipeline-based methods that first predict a note-level representation and subsequently convert it into a score-level format; and holistic methods that directly transcribe audio recordings into score format.

For the pipeline-based methods, we focus on the second stage: converting a note sequence into a score format. We propose a convolutional-recurrent neural network to track beats and extend the model to predict a MIDI score. The model outperformed two commercial software solutions, highlighting the advantage of tracking expressive temporal changes in musical performances.

For holistic methods, we use sequence-to-sequence models, which convert an audio spectrogram into a symbolic music score representation. We first explore RNN-based models using a LilyPond score, and demonstrate that incorporating multitask learning by jointly predicting a piano roll representation improves model performance. We then examine Transformer models,

adopting an event-based symbolic score representation. Our results show that Transformers outperform RNNs, and using a long short-term decoding strategy boosts the model’s capacity in both note-level transcription and capturing long-term musical features in score-level transcription.

In summary, this thesis presents our exploration towards audio-to-score piano transcription by investigating two approaches. We hope our work can inspire future work in AMT and related fields such as music generation, music education, and performance analysis.

Acknowledgements

At the completion of this thesis, I would like to express my sincere thanks to my academic supervisors. Emmanouil Benetos, for bringing me to the wonderful world of MIR research and your continuous support throughout my master's degree to the PhD. I would like to express my deepest thanks to you for encouraging and guiding me into the research world, helping me shape the underlying logic of academic thinking. You have been a beacon of guidance in both my academic and personal growth. Veronica Morfi, for your constructive and invaluable input throughout my PhD. Having you in my PhD journey has greatly broadened my research horizons and also encouraged my personal growth. Your continued encouragement and understanding have been crucial to me during the hard times. Simon Dixon, for your deep research insights, guidance and support over the years. What's more, the research environment you encouraged in C4DM has been fantastic. It made me feel truly at home within the group. And to Qiuqiang Kong, for encouraging me into the internship and for your incredible patience and guidance in our research discussions. It was a great pleasure working with you, which reshaped my understanding of mentoring, collaboration, and industry-level research.

My sincere thanks to my examiners Islah Ali-MacLachlan and Robin Laney, for your careful reading of my thesis and the detailed feedback you gave me to improve the manuscript. Markus Pearce, for your suggestions, care and support during my PhD milestones. Alvaro Bort, one of the most supportive research managers I could have hoped for, for your incredible support during my PhD journey.

I am deeply grateful to José Javier Valero Mas, Jorge Calvo-Zaragoza, María Alfaro-Contreras, and Juan Carlos Martinez-Sevilla, for your fruitful

discussions and companionship during my research visit. It was really an enjoyable research stay! To Adrien Ycart, Ching-Yu Chiu, Hyon Kim, and Hanwen Zhang, for the enjoyable research discussions and the time we worked together.

I would also like to thank Christof Weiß and Goran Glavaš for your reassuring understanding and support during my thesis writing-up. Meinard Müller, for your encouragement and empowering words. Your dedication to education and the way you interact with your students have reshaped my understanding of what education truly means. Cynthia Liem, for your extraordinary guidance and empathy during the Dagstuhl seminar. You helped me find my way through the confusion and uncertainty I faced during my personal growth. And my WIMIR mentors, Chris Donahue, Alexander Lerch, Simon Durand, Christof Weiß, Ichiro Fujinaga, and Anna Kruspe. You brought me to the warmest corner of our MIR community. I got to learn what the MIR community looks like through our meetings. I'd like to thank you for sharing your experiences and reflections of personal journeys, and for challenging my thinking through sharp questions, which helped me a lot in shaping my path.

My warm thanks to the fellow students in C4DM, particularly to Yukun, Elona, Mary, Ilaria, Saurjya, Cyrus, Pedro, Dave, Jiawen, Jingjing, Yixiao, Huan, Jinhua, Yinghao, Yazhou, Yin-Jyun, Sungkyun. It was a great opportunity to share our PhD journey together. Thank you for the inspiring discussions. It has been a real privilege to be in such a vibrant research group.

I'd like to express my sincere thanks to my friends who have accompanied me throughout different stages of my life, for all the joy and memories we have shared together: Changhong, Jiawen, Yiwei, Heping, Jiajie, Jiayu, Simin, Jingjing, Peiling, Juanjuan, Aiqi, Elona, Mary, Junfeng, and those from earlier years whom I have not named here. It has truly been a blessing to have you in my life through all the ups and downs, and to share many precious moments together. My sincere thanks to my online friends from weekly review meetings, for accompanying and encouraging me throughout my growth over the years.

Special thanks to my family, whose understanding and support have been crucial to me during my life journey.

Finally, I would like to express my heartfelt gratitude to music for entering my life—not only shaping my research path, but also giving my soul a true sense of belonging.

Funding

L. Liu is a research student at the UKRI Centre for Doctoral Training in Artificial Intelligence and Music, supported jointly by the China Scholarship Council and Queen Mary University of London. A part of Liu's work in this thesis is supported by The Alan Turing Institute through an Enrichment Scheme.

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 19 |
| 1.1 | Motivation and aim | 19 |
| 1.2 | Thesis outline | 22 |
| 1.3 | Contributions | 23 |
| 1.4 | Associated publications | 25 |
| 2 | Background | 30 |
| 2.1 | Introduction | 30 |
| 2.2 | AMT overview | 31 |
| 2.2.1 | Sub-tasks | 34 |
| 2.2.2 | The AMT pipeline | 39 |
| 2.2.3 | Holistic approach | 40 |
| 2.3 | Symbolic music representations | 42 |
| 2.3.1 | Music encoding formats | 42 |
| 2.3.2 | MIDI-derived score representations | 45 |
| 2.4 | Datasets and Evaluation Metrics | 50 |
| 2.4.1 | Datasets | 50 |
| 2.4.2 | Evaluation Metrics | 53 |
| 2.5 | State of the art | 56 |
| 2.5.1 | Overview | 56 |
| 2.5.2 | Neural Networks for AMT | 57 |
| 2.5.3 | Multi-task learning methods | 60 |
| 2.5.4 | Early-stage music language models | 61 |
| 2.6 | More on Deep Learning in Sequential Problems | 62 |
| 2.6.1 | RNN-based sequence-to-sequence models | 62 |
| 2.6.2 | Early-stage attention mechanisms | 63 |

| | | |
|----------|--|------------|
| 2.6.3 | Multi-head self-attention and the Transformer architecture | 64 |
| 2.7 | Conclusions | 65 |
| 3 | Data Collection and Preparation | 67 |
| 3.1 | Introduction | 67 |
| 3.2 | Dataset content | 68 |
| 3.2.1 | MuseSyn dataset: A dataset synthesised from piano sheet music collected at the MuseScore website | 68 |
| 3.2.2 | ACPAS dataset: A dataset of aligned classical piano audio and scores | 70 |
| 3.3 | Dataset statistics | 75 |
| 3.3.1 | Dataset splits | 75 |
| 3.3.2 | Other statistics | 76 |
| 3.4 | Discussion and conclusion | 83 |
| 4 | Pipeline-based Methods for Audio-to-Score Piano Transcription | 85 |
| 4.1 | Introduction | 85 |
| 4.2 | Performance MIDI-to-score conversion | 88 |
| 4.2.1 | Introduction | 88 |
| 4.2.2 | Beat tracking from performance MIDIs | 89 |
| 4.2.3 | From beat tracking to rhythm quantisation | 93 |
| 4.2.4 | Prediction of hand parts, time signatures and key signatures | 95 |
| 4.2.5 | Experiments | 97 |
| 4.3 | The audio-to-score transcription pipeline | 106 |
| 4.3.1 | Introduction | 106 |
| 4.3.2 | Method | 106 |
| 4.3.3 | Experiments | 107 |
| 4.4 | Conclusion | 110 |
| 5 | Holistic Methods for Audio-to-Score Piano Transcription | 113 |
| 5.1 | Introduction | 113 |

| | | |
|----------|---|------------|
| 5.2 | Joint multi-pitch detection and score transcription | 115 |
| 5.2.1 | Introduction | 115 |
| 5.2.2 | Data representations | 116 |
| 5.2.3 | Model architecture | 119 |
| 5.2.4 | Experiments | 121 |
| 5.2.5 | Conclusion | 127 |
| 5.3 | Joint note- and score-level transcription via long short-term decoding | 129 |
| 5.3.1 | Introduction | 129 |
| 5.3.2 | Event-based music representation | 130 |
| 5.3.3 | Long short-term decoding strategy | 134 |
| 5.3.4 | Experimental setup | 136 |
| 5.3.5 | Results | 139 |
| 5.3.6 | Summary of findings | 148 |
| 5.4 | Conclusion | 148 |
| 6 | Conclusion and Future Work | 150 |
| 6.1 | Summary of contributions | 150 |
| 6.2 | Continuing the trajectory | 152 |
| 6.2.1 | Towards instrument agnostic score-level transcription | 153 |
| 6.2.2 | Domain adaptation and pre-training strategies | 154 |
| 6.2.3 | Linking with other fundamental MIR tasks | 155 |
| 6.2.4 | Cross-modal and cross-disciplinary directions | 156 |
| 6.3 | Further challenges and opportunities | 157 |
| 6.3.1 | Datasets | 157 |
| 6.3.2 | Evaluation metrics | 158 |
| 6.3.3 | Non-Western music | 159 |
| 6.3.4 | Score-level transcription | 160 |
| 6.4 | Final remarks | 161 |
| | Bibliography | 162 |
| | A Appendices | 184 |
| A.1 | Detailed results for time–frequency representation comparison | 184 |

| | |
|--|-------|
| A.2 Detailed results of repeat runs for Transformer-based models | . 185 |
|--|-------|

List of Figures

| | | |
|-----|---|----|
| 2.1 | Typical stages of an AMT system: (a) input waveform; (b) time-frequency representation; (c) output piano-roll representation; (d) output music score, typeset using Musescore. The example corresponds to the first 4 seconds of W.A. Mozart’s Piano Sonata no. 11, 3rd movement. | 33 |
| 2.2 | (a) The pitch salience representation for the excerpt of Figure 2.1 using the method proposed by Benetos & Weyde (2015) ; (b) The corresponding binarized piano-roll representation. . . . | 35 |
| 2.3 | The rhythm-quantized transcription of the excerpt of Figure 2.1 , automatically transcribed using the method by Benetos & Weyde (2015) and typeset using Musescore (https://musescore.org). | 38 |
| 2.4 | Overview of the traditional AMT pipeline. | 39 |
| 2.5 | General structure for an end-to-end AMT system using encoder-decoder architecture. | 42 |
| 2.6 | Different music encoding formats for the same music snippet in staff notation. | 43 |
| 2.7 | Model architecture for the convolutional neural network by Kelz et al. (2016) for polyphonic piano transcription. The depicted network corresponds to the “ConvNet” architecture by Kelz et al. (2016) | 59 |
| 2.8 | Pitch salience representation for the excerpt of Figure 2.1 , using the deep salience method (Bittner et al., 2017). | 60 |
| 3.1 | Notes per segment in different datasets. Red dashed line: Estimated 95% cutoff points of the number of notes per segment over different segment lengths, computed using linear regression. | 77 |

| | | |
|-----|---|-----|
| 3.2 | Representable scores in terms of inter-onset intervals in musical time and note values by different resolutions (in subbeats per beat). | 78 |
| 3.3 | Inter-onset intervals in different datasets. | 79 |
| 3.4 | Note duration (in seconds) and note value (in beats) in different datasets. | 80 |
| 3.5 | Proportion of frequent inter-onset intervals and note values in subbeats (with a resolution of 24 subbeats per beat) across different datasets. Green blocks: IOIs that fall within the cutoff ratio among all IOIs. Orange blocks: Note values that fall within the cutoff ratio among all note values. We use a lower cutoff ratio for note value than for inter-onset intervals, since we need lower resolution for note values. | 81 |
| 3.6 | Pitch distributions per hand part in different datasets. | 81 |
| 3.7 | Ratio of non-aligned notes per piece in ACPAS-ASAP. | 82 |
| 4.1 | Illustration of onset beats and non-onset beats. Music example: the first five bars of “The Seasons - June (Barcarolle), by P. I. Tchaikovsky”. | 89 |
| 4.2 | Model architecture for onset beat prediction. | 91 |
| 4.3 | Model architecture for joint (onset) beat, (onset) downbeat and tempo estimation from performance MIDIs. The <code>ConvBlock</code> and <code>GRUBlock</code> have the same architecture as in Figure 4.2 | 92 |
| 4.4 | Training the rhythm quantisation model with the help of beat tracking. The <code>ConvBlock</code> and <code>GRUBlock</code> have the same model architecture and size as those in the beat tracking model. | 95 |
| 4.5 | Model architectures for the prediction of hand parts (a), time signature changes (b) and key signature changes (c). The <code>ConvBlock</code> and <code>GRUBlock</code> have the same model architecture as in the beat tracking model. | 96 |
| 4.6 | Baseline beat tracking model architecture using a temporal convolutional network. | 103 |

| | | |
|-----|--|-----|
| 5.1 | Example music score and corresponding LilyPond and Reshaped representation. | 118 |
| 5.2 | Model structure with layer sizes. | 120 |
| 5.3 | Example velocity-valued piano roll targets and outputs from model with the best input representation. Left column: ground truth piano rolls, Right column: transcribed piano rolls (non-binary, as predicted probabilities). | 125 |
| 5.4 | Example transcriptions. GT: ground truth music score. TR: transcribed music score. Ground truth key signatures are used for visualisation purposes. Transcription outputs for all pieces in the test set are available in our online repository at https://github.com/cheriell/ICASSP2021-A2S | 128 |
| 5.5 | An example overview of the proposed event-based music representation. | 131 |
| 5.6 | Baseline model architecture. PE is short for positional encoding. We start the output sequence with \mathbf{n}_0 to account for the <code><sos></code> token in practice. | 134 |
| 5.7 | Long short-term decoding strategy with a short-term decoder and a long-term decoder. Similar to Figure 5.6 , \mathbf{n}_0^s is used to account for the <code><sos></code> token in practice. | 135 |
| 5.8 | Example score transcriptions by the <i>LS-Decoding</i> model. GT: ground truth music score. TR: transcribed music score. We use MuseScore 4 to format the transcribed MIDI scores, where the ground truth key signatures are used for visualisation purposes. | 146 |
| 5.9 | Example note-level transcriptions by the <i>LS-Decoding</i> model presented in a piano-roll format. | 147 |

List of Tables

| | | |
|-----|--|-----|
| 3.1 | Dataset statistics across splits. Performances from the MAPS dataset are not counted towards distinct MIDI performance since they are derived from the CPM database. Pf: piano font. | 75 |
| 3.2 | Comparison on different AMT datasets. | 76 |
| 4.1 | Dataset Statistics. Performance from the MAPS dataset (Emiya et al., 2010) and the CPM database (CPMDatabase, 2021) for the same piece are not counted as different performances, since MAPS pieces are originally obtained from the CPM database. | 97 |
| 4.2 | Description of different input data encodings. | 99 |
| 4.3 | Note-level F-measure results for onset-beat tracking using different input data encoding combinations. | 100 |
| 4.4 | Note-level F-measure results on ablation study for input features. | 101 |
| 4.5 | Note-level F-measure results on the ablation study for data augmentation methods. | 102 |
| 4.6 | Beat-level F-measure results on the baseline and proposed models. | 104 |
| 4.7 | MV2H evaluation on performance MIDI-to-score conversion. | 104 |
| 4.8 | Note-level F-measure results for onset beat tracking with different MIDIs for model training, evaluated on the audio-to-score transcription pipeline. | 108 |
| 4.9 | MV2H evaluation on the ACPAS dataset using the audio-to-score conversion pipeline. The results of HighRes+Finale are labelled as “-” since Finale is no longer supported at the time of our experiments. | 109 |

| | | |
|-----|---|-----|
| 5.1 | Statistics of the MuseSyn dataset. For polyphony levels, numbers out of brackets are calculated without adding piano pedals; numbers in brackets are calculated with piano pedals. | 121 |
| 5.2 | F-measure of piano-roll prediction on different input representations and models. F_f : frame-level, F_{on} : note-level onset only, F_{onoff} : note-level onset and offset. | 124 |
| 5.3 | Word error rates and MV2H results in percentage for different score representations. LilyPond: Score-only model with LilyPond representation; Reshaped: Score-only model with Reshaped representation; Joint: Joint model with Reshaped representation. ϵ_{right} and ϵ_{left} refer to the word error rates for the transcriptions for the right-hand part and left-hand part. . . . | 126 |
| 5.4 | Model performances on single-task and multi-task models. . . . | 126 |
| 5.5 | Time and memory used in training and inference for the Score-only model with LilyPond and Reshaped representations. . . . | 127 |
| 5.6 | Score transcription results using MV2H metric on the MuseSyn test set. HRes: The high-resolution piano transcription system proposed by (Kong et al., 2021). RNN*: The method we developed in Section 5.2; RNN-Hierarchical*: The hierarchical method introduced by (Zeng et al., 2024); we use * to indicate music segments are pre-processed into bars using ground truth downbeats. | 140 |
| 5.7 | Frame-level and note-level transcription results on the MuseSyn dataset. RNN*: The method we developed in Section 5.2, where the results are evaluated based on the piano-roll output. On the contrary, in the latter two rows (new results in this experiment), results are evaluated based on the note sequence output from the decoder. | 141 |
| A.1 | F-measure of piano-roll prediction on different input representations and parameters. N_w : window length, N_m : number of mel bands, N_b : number of bins per octave, N_o : number of octaves, N_h : number of harmonics. | 186 |

| | | |
|-----|--|-----|
| A.2 | Score transcription results using MV2H metric on the MuseSyn test set. | 187 |
| A.3 | Frame- and note-level transcription results on the MuseSyn test set. | 187 |

List of Abbreviations

| | |
|------|---|
| A2PM | Audio-to-Performance MIDI (transcription) |
| A2S | Audio-to-Score (music transcription) |
| ADT | Automatic Drum Transcription |
| AI | Artificial Intelligence |
| AMT | Automatic Music Transcription |
| ASAP | Aligned Scores and Performances (dataset) |
| ASR | Automatic Speech Recognition |
| NBT | Neural Beat Tracking |
| CNN | Convolutional Neural Network |
| CQT | Constant-Q Transform |
| CRNN | Convolutional-Recurrent Neural Network |
| CTC | Connectionist Temporal Classification |
| CWMN | Common Western Music Notation |
| DBN | Deep Belief Network |
| DMRN | Digital Music Research Network |
| E2E | End-to-End |
| ELU | Exponential Linear Unit |
| GRU | Gated Recurrent Unit |
| HCQT | Harmonic Constant-Q Transform |
| HMM | Hidden Markov Model |
| IBI | Inter-Beat Interval |

| | |
|--------|---|
| ICASSP | International Conference on Acoustics, Speech and Signal Processing |
| ICLR | International Conference on Learning Representations |
| IOI | Inter-Onset Interval |
| ISMIR | International Society for Music Information Retrieval |
| LSTM | Long Short-Term Memory |
| MEI | Music Encoding Initiative |
| MIDI | Musical Instrument Digital Interface |
| MIR | Music Information Retrieval |
| MIREX | Music Information Retrieval Evaluation eXchange |
| MLM | Music Language Model |
| MPE | Multi-Pitch Estimation |
| NMF | Non-negative Matrix Factorization |
| OMR | Optical Music Recognition |
| PM2S | Performance MIDI-to-Score (conversion) |
| REMI | Revamped MIDI-derived events |
| RNN | Recurrent Neural Network |
| SED | Sound Event Detection |
| STFT | Short Time Fourier Transform |
| SVM | Support Vector Machine |
| TCN | Temporal Convolutional Network |
| TISMIR | Transactions of the International Society for Music Information Retrieval |
| VQT | Variable-Q Transform |
| XML | Extensible Markup language |

Chapter 1

Introduction

This thesis describes our work towards automatic audio-to-score (A2S) piano transcription using deep neural networks. We begin by providing our motivation and aim of our project in [Section 1.1](#). [Section 1.2](#) outlines the thesis contents and [Section 1.3](#) lists our main contributions in this thesis. Associated publications with this thesis are provided in [Section 1.4](#).

1.1 Motivation and aim

Automatic Music Transcription (AMT) is a core problem in the field of Music Information Retrieval (MIR), it is the process of converting music audio into human or machine-readable music scores using computer algorithms ([Benetos et al., 2019](#)). The use of AMT systems is not limited to creating music notation from music recordings, but goes to a wider field of music-related tasks. Automatic music transcription can be used in various applications such as music analysis, music education, music search/recommendation, and music generation (creation). Within the field of music information retrieval, automatic music transcription can be considered one of the “enabling” tasks, which benefits a list of other tasks, including source separation, expressive performance analysis, automatic music generation or accompaniment, performance error detection, and so on.

Research in automatic music transcription dates back to the 1970s, when the problem was solved using signal processing methods ([Klapuri & Virtanen,](#)

1977). In recent years, various methods have been used for automatic music transcription. Two of the most widely used methods are non-negative matrix factorisation (NMF) and deep learning. Since the introduction of deep learning (Goodfellow et al., 2016), there has been a large body of research on deep learning methods for automatic music transcription. In recent years, convolutional neural networks (CNNs), recurrent neural networks (RNNs), and the Transformer architecture have been widely used in automatic music transcription research (Böck & Schedl, 2012; Sigtia et al., 2016; Carvalho & Smaragdis, 2017; Kelz & Widmer, 2019). Over the years, a larger amount of research work has been focused on frame-level and note-level transcription, which aims to get a pianoroll-like output by multi-pitch estimation (frame-level) or a list of note sequences by including note tracking (note-level). In subsection 2.2.2 of Chapter 2, we explain these tasks in detail within the “traditional AMT pipeline”.

State-of-the-art note-level piano transcription systems already achieve very good results (Hawthorne et al., 2018; Kong et al., 2021; Toyama et al., 2023). As a result, people have been increasingly interested in *audio-to-score transcription* (A2S) (or *complete music transcription* in some literature) (Benetos et al., 2019). In audio-to-score transcription, the target is to estimate a human- or machine-readable music score notation from music audio (Román et al., 2020). Compared to frame- and note-level transcription, audio-to-score transcription is an under-explored problem and a challenging task because it includes transcription subtasks beyond detecting pitch or performed notes, but includes tasks such as meter estimation, rhythm quantisation, note duration estimation, key estimation, and voice separation. In Chapter 2, we provide a closer view of what audio-to-score transcription is all about, including its limitations and challenges.

In this thesis, we describe our work to explore deep learning methods for audio-to-score piano transcription. We chose to work on piano music because note-level piano transcription has already achieved good results (Kong et al., 2021), allowing an easier move forward, also because compared with other instruments, there is more data available to support our exploration of deep learning methods.

In this work, we aim to address several key research questions in the field of audio-to-score transcription. Specifically, we are interested in understanding the gap between note-level and score-level transcription, and in evaluating the relative advantages and limitations of pipeline-based versus holistic systems. We also investigate how different input–output representations influence score-level transcription, and whether multitask learning—jointly training note-level and score-level transcription—provides measurable benefits. Furthermore, we seek to explore the role of beat tracking and rhythm quantisation in relation to score-level transcription, and how these processes can be effectively integrated. Another line of inquiry concerns the comparative performance of different sequential modelling approaches, such as RNNs and Transformers, for this task. Finally, we examine how sequence-to-sequence models handle long musical sequences, and consider strategies for enabling more robust modelling of long-term dependencies in extended recordings.

To explore these questions, we focus on transcribing audio piano performance recordings into the format of a MIDI score (which includes rhythm, key signature, and time signature annotations), a text-based symbolic music notation LilyPond (Nienhuys & Nieuwenhuizen, 2003), or an event-based symbolic music representation. We explore a pipeline-based method which first does note-level transcription and then convert the transcribed note sequence into a MIDI score format, using a convolutional recurrent neural network architecture (see Chapter 4), as well as holistic methods which directly convert audio performance recordings into a score, using sequence-to-sequence models based on recurrent neural networks and Transformers (Vaswani et al., 2017) (see Chapter 5). Our study is grounded in two datasets: a collection of expressive classical piano recordings (the ACPAS dataset) and a corpus of popular music synthesised from scores (the MuseSyn dataset; see Chapter 3). We employ the ACPAS dataset to investigate pipeline-based methods, while the MuseSyn dataset serves as the basis for our exploration of holistic approaches. Across these studies, we propose several sequential modelling strategies that draw on convolutional neural networks, recurrent neural networks, and Transformers, and we evaluate different learning paradigms, including beat tracking, multi-task learning, and long short-term decoding. Through this work, we aim to

provide insights that advance the development of audio-to-score piano transcription systems suitable for real-world applications, while also contributing to and inspiring future research in the broader field of music information retrieval.

1.2 Thesis outline

The rest of the thesis is organised as follows:

In the latter part of **Chapter 1** (this chapter), we provide an overview of our main contributions in this thesis and the associated publications.

Chapter 2 provides a literature review of the transcription topic. Although the thesis topic is audio-to-score transcription, we opt not to exclude literature on frame-level and note-level transcription, since these serve as the basics of a complete transcription. In this chapter, we provide a detailed explanation of the stages of automatic music transcription and relevant sub-tasks. We also review the relevant datasets and evaluation metrics, as well as recent advancements in the field, which serve as the basis of this PhD project.

Chapter 3 describes the two datasets we propose and use in this thesis, namely, the MuseSyn dataset and the ACPAS dataset. These two datasets represent different music styles and levels of expressive performance, which is essential in the results analysis in the following chapters. In this chapter, we describe how the two datasets were collected and created and what is included in the datasets. We also provide a detailed dataset statistic analysis. In the conclusion section, we provide a brief discussion on the limitations of the datasets, while leaving the broader discussion of the limitations and future work to the last chapter (**Chapter 6**).

Chapter 4 presents our exploration of pipeline-based methods for audio-to-score piano transcription. Based on existing methods for note-level transcription, we propose a method for converting performance MIDI into score MIDI based on neural beat tracking. We combine our proposed methods for performance MIDI-to-score conversion with an existing audio-to-performance MIDI

transcription system into an audio-to-score transcription pipeline system. We evaluate this system on the ACPAS dataset and compare the results with two public commercial software and two previous statistical methods.

Chapter 5 focuses on holistic methods for audio-to-score piano transcription. In this chapter, we focus on the use of sequence-to-sequence models for audio-to-score piano transcription, considering the music audio spectrogram as the input sequence and a symbolic score representation as the output sequence. We first describe our experiments to use multitask learning for audio-to-score piano transcription, in which we build a model to jointly learn a piano-roll and a LilyPond-based music score representation. Results show a positive effect on using multitask learning. Based on these findings, we continued our exploration by jointly learning a descriptive representation and a prescriptive representation. We compare the use of recurrent neural networks (RNN) with Transformers ([Vaswani et al., 2017](#)), and show in our results that Transformers outperform RNNs, particularly in the ability to process long recordings. We propose to use a long short-term decoding strategy, which enhances the learning of both short-term musical elements (multi-pitch estimation) and long-term musical elements (metrical alignment in score transcription).

Chapter 6 concludes the thesis. We first provide a summary of the contributions described in this thesis, followed by a discussion on the limitations of current work and opportunities for future work. In the end, we provide some remarks about our thoughts on this research topic and our next steps.

1.3 Contributions

The contributions of this thesis can be listed as follows:

Chapter 2 provides a detailed literature review on automatic music transcription, including:

- A tutorial-like introduction to automatic music transcription, with a special focus on the stages in the traditional transcription pipeline and

various sub-tasks. We specifically define the pipeline-based methods and holistic methods for audio-to-score transcription.

- A review of the datasets and evaluation metrics for note-level and score-level transcription.
- A review of the state-of-the-art methods for automatic music transcription.

Chapter 3 describes the two datasets we created in the PhD project. This includes:

- The MuseSyn dataset for popular piano music, whose musical scores are collected online from the MuseScore website, and audio recordings are synthesised from the collected scores.
- The ACPAS dataset for Western classical piano music, which is collected and cleaned from three classical piano databases.
- We additionally provide a detailed statistical analysis for both datasets.

Chapter 4 introduces our proposed methods for pipeline-based methods for audio-to-score piano transcription. We include:

- A system for neural beat tracking using convolutional recurrent neural networks.
- A system for performance MIDI-to-score conversion utilising the beat tracking system above.
- An exploration of different input encoding methods for MIDI note sequences for tracking beats in performance MIDIs.
- A pipeline-based method for audio-to-score piano transcription.

Chapter 5 describes our proposed methods for holistic methods for audio-to-score piano transcription. The contributions include:

- We propose the first holistic model that transcribes polyphonic piano music into both a piano-roll format (corresponding to a descriptive notation of the music audio) and a score in Western staff notation (corresponding to a prescriptive notation of the musical audio).
- We propose a modified score representation based on the LilyPond format for modelling polyphonic music that learns and predicts seven times faster and performs better than the LilyPond format score representation on this model.
- We evaluate the effect of using different input time-frequency representations, including the Short Time Fourier Transform (STFT), Mel Spectrogram, Constant-Q Transform (CQT), Variable-Q Transform (VQT) and Harmonic Constant-Q Transform (HCQT).
- We design an event-based symbolic score representation that encodes both note-level and score-level information, which enables the joint learning of a performance MIDI and a score MIDI using a sequence-to-sequence network.
- We use a Transformer decoder for audio-to-score transcription, which outperforms a recurrent neural network decoder.
- We propose to use a long short-term decoding strategy which enhances the learning of both short-term musical features in multi-pitch estimation (i.e., pitch, onset and offset) and long-term musical features in score-level transcription (i.e., metrical alignment).

Chapter 6 provides a discussion on the limitations of current research and points out opportunities in this research topic.

1.4 Associated publications

In this section, we list the associated publications to this PhD thesis. This covers the publications that contribute to the main content of this thesis, as well as other publications from the PhD period that are relevant to the PhD

topic. We provide a brief description of each publication and how it relates to this thesis, as well as how the PhD candidate contributed to the publication. Additionally, we provide the relevant public online code repositories and datasets associated with the publications.

Below are the associated publications ordered by publication date:

1. Adrien Ycart, Lele Liu, Emmanouil Benetos and Marcus T. Pearce. “Investigating the perceptual validity of evaluation metrics for automatic piano music transcription,” *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 3(1):68-81, 2020.

This is a collaborative work with Dr Adrien Ycart, who was working on his PhD at that time. This work describes an investigation of the perceptual validity of the benchmark evaluation metrics for automatic piano transcription, namely, the F-measure (Hawthorne et al., 2018) via a listening test, and proposes a new perceptually-meaningful evaluation metric for future usage. In this work, I mainly collaborated to discuss, implement, and test the musical features used in developing the proposed evaluation metric. One of the main findings in this paper, that “onset-only notewise F-measure is the benchmark metric that correlates best with human judgement”, is particularly relevant to the result analysis in this PhD work, where we therefore prioritise onset-only metrics for model comparison.

The online code repository for the listening test webpage is available at https://github.com/adrienycart/AMT_perception_website.

A Python implementation of the used music features and the pre-trained metric can be found at <https://github.com/adrienycart/PEAMT>.

2. Lele Liu, Veronica Morfi and Emmanouil Benetos, “Joint multi-pitch detection and score transcription for polyphonic piano music,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Canada, Jun 2021.

This paper presents our first working system for audio-to-score piano transcription during the PhD project. The main contents of this paper

are described in Section 5.2 of Chapter 5. In this paper, we propose to jointly do multi-pitch estimation and audio-to-score transcription in a multitask system. We use a multitask model combined with a convolutional recurrent neural network and sequence-to-sequence models with recurrent neural networks and attention mechanisms. In this paper, we propose a reshaped score representation that outperforms a LilyPond representation in terms of both prediction accuracy and time/memory resources and compare different input audio spectrograms. Additionally, we present the first dataset we use in the PhD project, the MuseSyn dataset, which is a synthesised dataset from collected online musical scores. The MuseSyn dataset is described in Chapter 3.

The public code repository for the experiments in this paper can be found at <https://github.com/cheriell/ICASSP2021-A2S>.

The MuseSyn dataset is available at Zenodo: <https://zenodo.org/records/4527460>.

3. Lele Liu and Emmanouil Benetos, “From audio to music notation,” in *Handbook of Artificial Intelligence for Music*, E. Miranda (ed.), pp. 693-714, Springer, 2021 (ISBN: 978-3-030-72115-2).

This is a review book chapter on the automatic music transcription task. The main text is written jointly by me and my primary PhD supervisor, Dr. Emmanouil Benetos. This review article contributes to the main part of the literature review in this thesis in Chapter 2.

4. Lele Liu, Veronica Morfi and Emmanouil Benetos, “ACPAS: A Dataset of aligned classical piano audio and scores for audio-to-score transcription,” *International Society for Music Information Retrieval Conference Late-Breaking Demos Session (ISMIR-LBD)*, Nov 2021.

In this short paper, we describe our second dataset in this PhD thesis, the ACPAS dataset. This dataset provides a different perspective than the MuseSyn dataset by targeting a different music style and level of expressive performance. The ACPAS dataset features Western classical music and rich expressive performance and works in this PhD project to

better understand the model performance in real scenarios. The details of this dataset are described in [Chapter 3](#).

The dataset is available online at: https://cheriell.github.io/research/ACPAS_dataset.

5. Lele Liu, Qiuqiang Kong, Veronica Morfi and Emmanouil Benetos, “Performance MIDI-to-score conversion by neural beat tracking,” in *Proceedings of the 23rd International Society for Music Information Retrieval (ISMIR) Conference*, Bengaluru, India, Dec 2022. **Best paper award**

This paper is the outcome of an internship at ByteDance, under the mentorship of Dr Qiuqiang Kong and my PhD supervisors. In this work, we focus on the conversion from a performance MIDI into a score format. We propose to use neural beat tracking for rhythm quantisation with a convolutional recurrent neural network. This network is further extended to predict other score components, including key signatures, time signatures, and hand parts in piano music. The main contents of this paper are included in [Chapter 4](#) in this thesis. In this thesis, we further extend the work in the paper to combine the proposed system with an existing performance MIDI transcription system ([Kong et al., 2021](#)) as a pipeline-based system for audio-to-score piano transcription.

We released our experimental code online at <https://github.com/cheriell/PM2S>.

Additionally, below is a list of other publications and pre-print reports during the PhD time, also ordered by publication date:

1. Lele Liu and Emmanouil Benetos, “Automatic music accompaniment with a chroma-based music data representation,” *DMRN+14: Digital Music Research Network One-day Workshop*, London, Dec 2019.
2. Adrien Ycart, Lele Liu, Emmanouil Benetos and Marcus T. Pearce. “Musical features for automatic music transcription evaluation,” *arXiv preprint arXiv:2004.07171*, 2020.

3. Lele Liu, Veronica Morfi and Emmanouil Benetos, “Joint piano-roll and score transcription for polyphonic piano music,” *DMRN+15: Digital Music Research Network One-day Workshop*, London, Dec 2020.
4. Giovanni Bindi, Nils Demerlé, Rodrigo Diaz, David Genova, Aliénor Golvet, Ben Hayes, Jiawen Huang, Lele Liu, Vincent Martos, Sarah Nabi, Teresa Pelinski, Lenny Renault, Saurjya Sarkar, Pedro Sarmiento, Cyrus Vahidi, Lewis Wolstanholme, Yixiao Zhang, Axel Roebel, Nick Bryan-Kinns, Jean-Louis Giavitto, Mathieu Barthet, “AI (r)evolution – where are we heading? Thoughts about the future of music and sound technologies in the era of deep learning,” *arXiv preprint arXiv:2310.18320*, 2023.
5. Lele Liu and Christof Weiß, “Utilizing cross-version consistency for domain adaptation: A case study on music audio,” in *Tiny Papers Track at International Conference on Learning Representations (Tiny Papers @ ICLR 2024)*, May 2024. **Notable (oral)**
6. Lele Liu and Christof Weiß, “Unsupervised domain adaptation for music transcription: Exploiting cross-version consistency,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025.
7. Ching-Yu Chiu, Lele Liu, Christof Weiß and Meinard Müller, “Cross-modal approaches to beat tracking: A case study on Chopin Mazurkas,” *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 8(1):55–69, 2025.

Chapter 2

Background

2.1 Introduction

The field of Music Information Retrieval (MIR) focuses on creating methods and practices for making sense of music data from various modalities, including audio, video, images, scores, and metadata (Serra et al., 2013). Within MIR, a core problem which to the day remains open is automatic music transcription (AMT), the process of automatically converting an acoustic music signal into some form of musical notation. The creation of a method for automatically converting musical audio to notation has several uses including but also going beyond MIR: from software for automatic typesetting of audio into staff notation or other music representations, to the use of automatic transcriptions as a descriptor towards the development of systems for music recommendation, to applications for interactive music systems such as automatic music accompaniment, for music education through methods for automatic instrument tutoring, and towards enabling musicological research in sound archives, to name but a few. Interest in AMT has grown during recent years as part of recent advances in artificial intelligence and in particular deep learning, which have led to new applications, and systems, as well as have led to a new set of technical, methodological and ethical challenges related to this problem (Benetos et al., 2019).

The topic of this PhD thesis lies within the scope of AMT, where we focus on the transcription of a single instrument (piano) and a specific output

format (music score notation). Despite this, in this chapter, we present a literature survey of state-of-the-art research and open topics for AMT in general, focusing on methods based on AI and deep learning in particular. The focus of the chapter is on automatic transcription of pitched sounds. For relevant topics, we refer to a recent review by [Wu et al. \(2018\)](#) on the related task of automatic drum transcription (ADT); for a detailed look at signal processing and statistical methods for AMT, we refer to the book by [Klapuri & Davy \(2006\)](#); for a discussion related to the challenges of AMT methods relying on signal processing or statistical methods, please see the review paper by [Benetos et al. \(2013\)](#). A recent tutorial-like overview of both “traditional” machine learning and deep learning methodologies for AMT is presented in the survey by [Benetos et al. \(2019\)](#). For this thesis in particular, we also look into relevant fields such as music encoding and sequential modelling beyond AMT, since they are relevant to the methods we used.

The rest of this chapter is organized as follows. [Section 2.2](#) provides a general overview of the AMT task. [subsection 2.2.1](#) provides a concise definition of various problems that have been posed under AMT. [Section 2.3](#) reviews various music encodings. An overview of commonly used datasets and evaluation metrics in AMT is presented in [Section 2.4](#). An overview of the state-of-the-art in AMT is presented in [Section 2.5](#), including a more detailed look at deep learning methods for the task. In [Section 2.6](#), we specifically look into deep learning methods in sequential modelling. Finally, conclusions are presented in [Section 2.7](#). Most of this chapter is modified from our previous book chapter ([Liu & Benetos, 2021](#)).

2.2 AMT overview

The first attempts to address the AMT problem date back to the 1970s and the dawn of the field of computer music (e.g., ([Piszczałski & Galler, 1977](#))), while the problem faced a resurgence in the mid-2000s with the development of methods for audio signal processing and pattern recognition, and encountered a second wave of popularity in recent years following the emergence of deep learning methods. Irrespective of the methodologies used to investigate and develop tools and practices for AMT, researchers addressing this

task draw knowledge from several disciplines, including digital signal processing, machine learning, music perception and cognition, musical acoustics, and music theory. There are also strong links with other problems both within and beyond MIR, including optical music recognition (OMR), which is the counterpart of AMT but for printed music or manuscripts instead of recorded audio (Rebelo et al., 2012), automatic speech recognition (ASR) and speaker detection (Yu & Deng, 2015), sound event detection (SED) for everyday and nature sounds (Virtanen et al., 2018), and object recognition and tracking in video (Chen, 2016). AMT is also closely related to the fields of music language modelling and symbolic music processing (Boulangier-Lewandowski et al., 2012), bridging the acoustic and symbolic domains in music.

Given the complexity of the problem of AMT, the overarching task is often split into subtasks, including pitch/multi-pitch detection (MPE), onset and offset detection, instrument identification and tracking, meter estimation and rhythm quantization, estimation of dynamics and expression, and typesetting/engraving. However, recent advances in artificial intelligence have promoted the development of “holistic” (or “end-to-end”) methods for AMT, thus often skipping intermediate tasks or steps and directly producing a transcription in a particular notation format. Figure 2.1 shows the typical stages of an AMT system for a short excerpt from a Mozart sonata, starting with the input waveform, the extracted time-frequency representation (in this case a short-time Fourier transform magnitude spectrogram), the output transcription in piano-roll representation, and the output transcription in the form of Western staff notation.

Despite active research on this problem over the years, AMT is still faced with several challenges, both technical and ethical. Broadly, the performance of certain AMT systems can be deemed sufficient for audio recordings containing solo acoustic instruments, within the context of Western tonal music, assuming a relatively moderate tempo and a level of polyphony around three to four. Here, the term “polyphony” refers to the maximum number of concurrent pitches at a given time instant. The problem of automatically transcribing audio recordings which contain sounds produced by multiple instruments, vocals, and percussion with a high polyphony level or a fast tempo, is still rel-

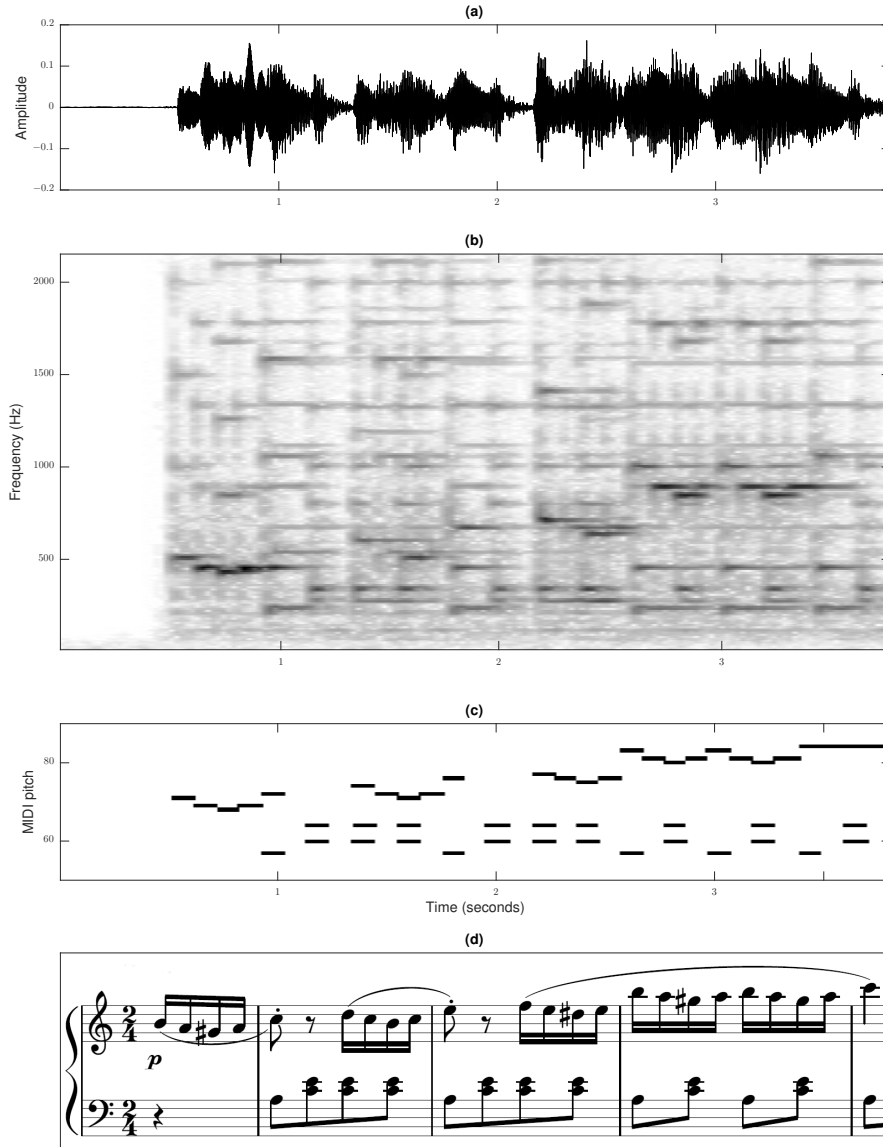


Figure 2.1: Typical stages of an AMT system: (a) input waveform; (b) time-frequency representation; (c) output piano-roll representation; (d) output music score, typeset using Musescore. The example corresponds to the first 4 seconds of W.A. Mozart’s Piano Sonata no. 11, 3rd movement.

actively limited. Other factors that can affect the performance of such systems include the existence of distortions either at the instrumental production stage or at the audio production/mastering stage, or cases where the performance or composition in question does not fall under the auspices of Western tonal music. A relatively new challenge which has emerged with the adoption of data-driven methods for addressing the task is the bias imposed by the algorithms through the choice of datasets. The fact that most datasets for AMT include Western tonal music performed by solo piano or other solo Western orchestral instruments has created certain limits and biases with respect to the range of instruments or to the range of music cultures and styles that state-of-the-art AMT systems can support. Limitations of symbolic representations and encodings for music (MIDI, MEI¹, MusicXML, Lilypond (Nienhuys & Nieuwenhuizen, 2003), etc.) also further constrain the potential of current AI-based AMT systems to support the transcription of music performances that cannot necessarily be expressed through Western staff notation or do not assume 12-tone equal temperament.

2.2.1 Sub-tasks

As mentioned in Section 2.1, AMT is divided into several subtasks, and most approaches have only been addressing a small subset of these subtasks. Perhaps the most essential subtask (especially when referring to the transcription of pitched sounds) is *pitch detection*, or in the case of multiple concurrent sounds, *multi-pitch detection* (MPE). Here, we define pitch in the same way by Hartmann (1996), where a sound has a certain pitch if it can be reliably matched to a sine tone of a given frequency at a sound pressure level of 40 dB. Typically, this task refers to estimating one or more pitches at each time frame (e.g., at 10ms intervals), where pitch is typically expressed in Hz. Given the close links between pitch and the fundamental frequency of periodic signals, this task is often referred to as *multiple-F0 estimation*. This task is publicly evaluated annually as part of the Music Information Retrieval Evaluation eXchange (MIREX)² task on MultiF0 estimation.

¹<https://music-encoding.org/>

²https://www.music-ir.org/mirex/wiki/MIREX_HOME

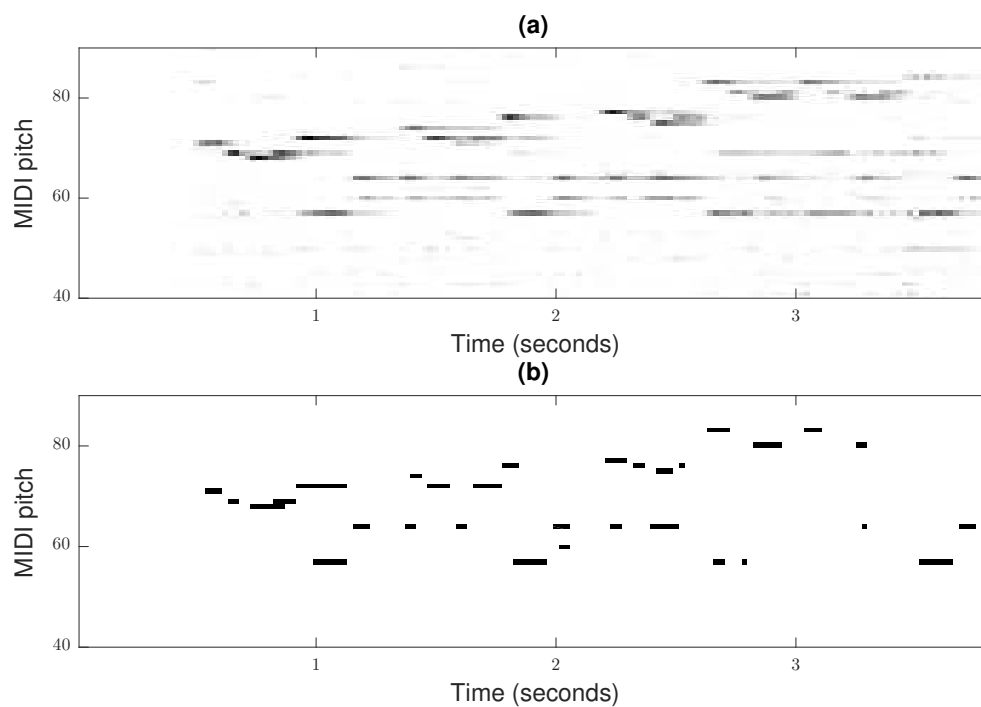


Figure 2.2: (a) The pitch salience representation for the excerpt of [Figure 2.1](#) using the method proposed by [Benetos & Weyde \(2015\)](#); (b) The corresponding binarized piano-roll representation.

It is often useful for multi-pitch detection systems to produce a non-binary representation of estimated pitches over time, which could be used for pitch visualization purposes, or as an intermediate feature for other MIR tasks that rely on an initial pitch estimate (e.g., melody estimation ([Salamon et al., 2014](#)), chord estimation ([McVicar et al., 2014](#))). Often this representation is referred to as *pitch salience*, or a *time-pitch representation*. [Figure 2.2\(a\)](#) shows the pitch salience representation for the excerpt of [Figure 2.1](#) using the method proposed by [Benetos & Weyde \(2015\)](#).

Moving on to a higher level of abstraction, which is closer to how humans might transcribe music, we would need to express notes as characterised by their start time, end time, and pitch, in a similar way as expressed, e.g., in the MIDI format. This task is referred to as *note tracking*, and involves the

subtasks of *onset detection* (i.e. detecting the start of a note), *offset detection* (i.e., detecting the end of a note), and (multi-)pitch detection. A comprehensive tutorial on signal processing-based methods for onset detection can be found in a tutorial by [Bello et al. \(2005\)](#). Approaches for note tracking are publicly evaluated annually as part of the Music Information Retrieval Evaluation eXchange (MIREX) note tracking task ([MIREX](#)). [Figure 2.2\(b\)](#) shows the output of the note tracking process by performing simple thresholding on the pitch salience of [Figure 2.2\(a\)](#).

Over the past decades, research in AMT has been largely focused on frame- and note-level transcription, i. e., *multi-pitch estimation* (MPE) which estimates active pitches from a music recording at a given time frame (frame-level transcription), and *note tracking* which estimates sequences of note events in tuples of note pitch, onset, offset, and velocity (note-level transcription) ([Bentos et al., 2019](#)). People have developed various methods for the AMT task, from non-negative matrix factorisation ([Smaragdis & Brown, 2003](#)) to deep learning methods ([Hawthorne et al., 2018](#); [Kong et al., 2021](#); [Hawthorne et al., 2021](#); [Gardner et al., 2022](#)). Recent years have seen a boost in the development of note-level piano transcription systems, featuring some milestones in deep learning-based AMT algorithms. Related works include the onset and frames framework which uses a multitask learning method that jointly learns note pitch and onset ([Hawthorne et al., 2018](#)), the ADSR framework which uses an estimated piano attack–decay–sustain–release curve to help with note-level transcription ([Kelz et al., 2019b](#)), the high-resolution piano transcription system which uses a new output representation to enable high-resolution note onset, offset and piano pedal prediction ([Kong et al., 2021](#)), and newer Transformer-based architectures ([Hawthorne et al., 2021](#); [Gardner et al., 2022](#); [Toyama et al., 2023](#)). Although there are still challenges such as dealing with music transcription with complex instrumentations (including singing) ([Bittner et al., 2022](#); [Wang & Jang, 2023](#)), transcribing solo piano recordings into performance MIDI tracks has achieved promising results.

In addition to the detection of pitched sounds and their timings, a key element towards a successful musical transcription is on assigning each detected note to the musical instrument that produced it. This task is referred to in the

literature as *instrument assignment*, *timbre tracking*, or *multi-pitch streaming*. A closely related task in the wider field of MIR is that of *musical instrument recognition* from audio, which has received relatively little attention from the research community (we refer readers to a recent overview by [Humphrey et al. \(2018\)](#)).

The above-mentioned note tracking task estimates the start and end times of notes, but in terms of seconds as opposed to beats or any other metrical subdivision. To that end, the task of *rhythm transcription* or *note value recognition* aims to estimate the metrical structure of the music recording in question and estimate the note timings and durations in terms of metrical subdivisions ([Cogliati et al., 2016](#); [Nakamura et al., 2017b](#)). By having estimated pitches with their respective timings in terms of meter, one can typeset the transcribed audio in some form of human-readable music notation, e.g., Western staff notation. This is a task that, depending on the complexity of the music performance in question, might also require splitting the detected stream of notes into multiple music staves (this is referred to as *voice separation* and *staff estimation*). The process of converting music audio into staff notation is sometimes referred to as *audio-to-score transcription* (A2S), or *complete music transcription*, taking into account that such a “complete” transcription might not contain information related to musical instruments, phrasing, expression or dynamics.

[Figure 2.3](#) shows the rhythm-quantised transcription of the excerpt of [Figure 2.1](#) in Western staff notation, automatically transcribed using the method by [Benetos & Weyde \(2015\)](#). While from a first glance there are few similarities with the score of [Figure 2.1\(d\)](#), a close inspection shows that the majority of pitches have been correctly detected, although their respective durations are not properly estimated, which can be attributed to sustaining and pedalling of the piano performance of this piece.

Rhythm quantisation is one of the most essential sub-tasks in converting a note-level transcription into a score which has been widely discussed in the literature. Several challenges make this task difficult, including handling expressive tempo changes and note value detection, since note offset (key release time) is not always audible, especially when piano pedals are used. It is also



Figure 2.3: The rhythm-quantized transcription of the excerpt of Figure 2.1, automatically transcribed using the method by Benetos & Weyde (2015) and typeset using Musescore (<https://musescore.org>).

common in piano performances that note offsets are inconsistent with what is written in the music scores, which happens when performers release the key earlier or later. Over the years, people have applied different algorithms to the task. Recent years have seen some advances in statistical methods (Nakamura et al., 2017a,b; McLeod & Steedman, 2018b; Nakamura et al., 2018; Shibata et al., 2021). Nakamura et al. (2017b) proposed an improvement for rhythm transcription using a merged-output hidden Markov model to solve the problem introduced by multiple voices. A note value prediction method using Markov random fields was proposed by Nakamura et al. (2017a). The two methods were further improved for AMT by combining a multi-pitch estimation model (Nakamura et al., 2018; Shibata et al., 2021). McLeod & Steedman (2018b) proposed a hidden Markov model-based meter detection method for aligning MIDI performances. In contrast to rhythm quantisation, score complement spans a wider range of sub-tasks. Hiramatsu et al. (2021) proposed to use a recurrent neural network (RNN) to joint estimate note val-

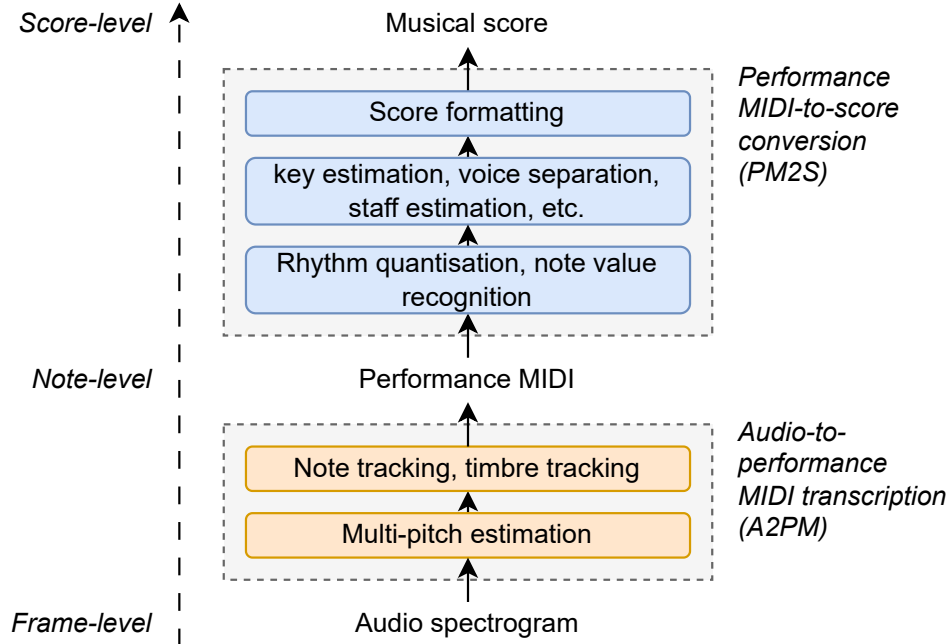


Figure 2.4: Overview of the traditional AMT pipeline.

ues and voices from note pitches and onset times, which in combination with multi-pitch estimation and rhythm quantisation, outperformed previous AMT methods (Cogliati et al., 2016; Shibata et al., 2021). More recently, Suzuki (2021) proposed a Transformer-based solution for generating human-readable scores from quantised MIDI files.

2.2.2 The AMT pipeline

Overall, the traditional AMT pipeline can be divided into two major parts:

1. **Audio to performance MIDI transcription (A2PM)**, or *note-level transcription* which targets a *note-level* output in the form of a performance MIDI, and
2. **Performance MIDI to score conversion (PM2S)**, which targets the conversion from a performance MIDI to a *score-level* output in the form

of a musical score.

In [Figure 2.4](#), we present an overview of the traditional AMT pipeline, organized into two major parts encompassing the sub-tasks detailed in the previous section ([subsection 2.2.1](#)).

Over the years, the first part (audio to performance MIDI transcription) has been the focus of many research works ([Hawthorne et al., 2018](#); [Kong et al., 2021](#); [Bittner et al., 2022](#); [Yan & Duan, 2024](#)). On the contrary, although research on the various sub-tasks of performance MIDI-to-score conversion (PM2S) dates back to earlier decades ([Desain & Honing, 1989](#); [Temperley & Sleator, 1999](#); [Raphael, 2001](#); [Takeda et al., 2002](#)), the first paper that fully brings the PM2S problem into literature was by [Cogliati et al. \(2016\)](#) much later, in which the authors worked on converting a performance MIDI recording into a LilyPond score ([Nienhuys & Nieuwenhuizen, 2003](#)). The authors firstly fixed spurious overlapping notes according to a defined note overlapping ratio, then applied a probabilistic model using hidden Markov models (HMM) based on a tactus-root combination concept ([Temperley, 2009](#)) for meter, harmony, and stream estimation. After that, the note onsets and offsets are quantised to beat subdivisions. Note spellings and staves are determined from the predicted harmony and streams. Cogliati further proposed to use a convolutional sparse coding-based multi-pitch estimation and a Melisma Analyser-based rhythm quantisation method for piano transcription ([Cogliati, 2018](#)). Alternatively, Nakamura et al. proposed an improvement in the hidden Markov model-based rhythm quantisation method, which is used to create a complete audio-to-score transcription system in combination with a multi-pitch estimation system based on probabilistic latent component analysis ([Nakamura et al., 2018](#)). The proposed audio-to-score transcription pipeline was further improved by [Shibata et al. \(2021\)](#) using a Markov random field-based rhythm quantisation and deep learning-based multi-pitch estimation.

2.2.3 Holistic approach

Opposed to the traditional AMT pipeline, there are also works that use a *holistic* (or *end-to-end*) approach, which directly transcribes music record-

ings (or more precisely, in the form of audio spectrograms) into a symbolic score format (Carvalho & Smaragdis, 2017; Román et al., 2018; Román et al., 2019; Román et al., 2020). In this scenario, a deep learning network is used to link the system input and output. A challenge in designing an end-to-end system is that the input and output of the system cannot be aligned directly (one is a time-based representation and the other is a representation in terms of metres or symbolic encoding). As a result, research has focused on encoder-decoder architectures that do not rely on frame-wise aligned annotations between the audio and music score (Carvalho & Smaragdis, 2017; Nishikimi et al., 2019). In Figure 2.5, we provide an encoder-decoder structure commonly used in holistic AMT systems. Recent works have shown the potential of encoder-decoder methods, although their performance on polyphonic music transcription remains less explored in the literature. In 2017, Carvalho and Smaragdis proposed a method for end-to-end music transcription using a sequence-to-sequence architecture combined with CNNs and RNNs (Carvalho & Smaragdis, 2017). The developed system can output a textual music encoding in the Lilypond language from an input audio waveform. However, the work focused mainly on monophonic music (which showed high-level performance), and only a simple scenario of polyphonic music was tested (with two simultaneous melodies within a pitch range of two octaves). Another exploration on singing transcription by Nishikimi et al. (2019) also used a sequence-to-sequence model. A point worth mentioning is that they applied an attention loss function for the decoder, which improved the performance of the singing transcription system. The work, still, focused only on the monophonic singing voice.

Using an encoder-decoder architecture is a simple way of designing end-to-end AMT systems, but there are also other works using Connectionist Temporal Classification (CTC). A recent example combines the use of a CRNN and a CTC loss function (Román et al., 2019; Román et al., 2020). The CTC loss function enables the system to be trained using pairs of the input spectrogram and output textual encoding. In that work, a simple polyphonic scenario is considered where four voices are included in a music piece (in string quartets or four-part Bach chorales). Still, the problem of end-to-end complete

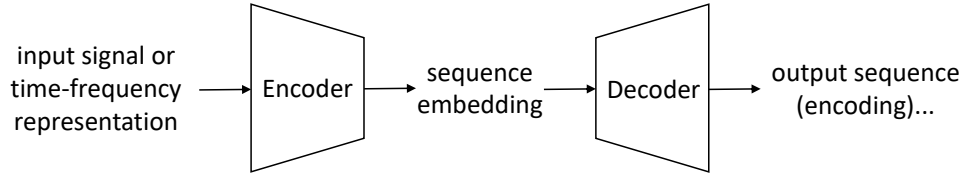


Figure 2.5: General structure for an end-to-end AMT system using encoder-decoder architecture.

music transcription with unconstrained polyphony remains unsolved. Another limitation among those above-mentioned holistic A2S methods is that they only output a beat-quantised musical notation, lacking important descriptive information in music analysis, such as absolute time for note onsets and offsets.

2.3 Symbolic music representations

In this section, we review widely used music score encoding formats alongside several MIDI-derived music representations. These representations are pertinent to our thesis as potential candidates for our chosen music score representation.

2.3.1 Music encoding formats

Widely used music encoding formats include MusicXML, MEI, LilyPond, **kern, ABC notation, and MIDI. In this section, we provide a brief overview of several of these formats and examine their applications as discussed in the literature. Our focus is primarily on music encodings designed for Common Western Music Notation (CWMN). ABC notation is excluded from this discussion, as it is primarily suited for monophonic music, which falls outside the scope of our work. Similarly, a review of MIDI-derived score representations is deferred to the next section. [Figure 2.6](#) shows the different music encodings we discuss in this section.



(a) Musical score

```

<score>
  <scoreDef>
    <staffGrp>
      <staffDef n="1" lines="5" meter.count="4" meter.unit="4">
        <label>Piano</label>
        <labelAbbr>Pno.</labelAbbr>
        <clef shape="G" line="2" />
      </staffDef>
    </staffGrp>
  </scoreDef>
  <section xml:id="s1">
    <pb />
    <measure xml:id="m4y39m0" right="end" n="1">
      <staff xml:id="m1s1" n="1">
        <layer xml:id="m1s1l1" n="1">
          <note xml:id="nsbw652" dur="4" pname="c" oct="4" />
          <note xml:id="nl1nj7f" dur="4" pname="d" oct="4" />
          <rest xml:id="rlpk78z0" dur="2" />
        </layer>
      </staff>
    </measure>
  </section>
</score>

```

(c) MEI

```

\new PianoStaff <<
  \new Staff {
    \clef treble
    \key c \major
    \time 4/4
    \relative c' {
      % Measure 1
      c4 d4 r2
    }
  }
>>

```

(d) LilyPond

```

**kern
*M4/4
*clefG2
*k[]
=1
4c
4d
2r
=
*_

```

(e) **kern

```

<measure number="1" width="197.58">
  <print>
    <system-layout>
      <system-margins>
        <left-margin>50</left-margin>
        <right-margin>814.84</right-margin>
      </system-margins>
      <top-system-distance>70</top-system-distance>
    </system-layout>
  </print>
  <attributes>
    <divisions>1</divisions>
    <key>
      <fifths>0</fifths>
    </key>
    <time>
      <beats>4</beats>
      <beat-type>4</beat-type>
    </time>
    <clef>
      <sign>G</sign>
      <line>2</line>
    </clef>
  </attributes>
  <note default-x="80.21" default-y="-50">
    <pitch>
      <step>C</step>
      <octave>4</octave>
    </pitch>
    <duration>1</duration>
    <voice>1</voice>
    <type>quarter</type>
    <stem>up</stem>
  </note>
  <note default-x="110.6" default-y="-45">
    <pitch>
      <step>D</step>
      <octave>4</octave>
    </pitch>
    <duration>1</duration>
    <voice>1</voice>
    <type>quarter</type>
    <stem>up</stem>
  </note>
  <note default-x="140.99" default-y="-20">
    <rest />
    <duration>2</duration>
    <voice>1</voice>
    <type>half</type>
  </note>
  <barline location="right">
    <bar-style>light-heavy</bar-style>
  </barline>
</measure>

```

(b) MusicXML

Figure 2.6: Different music encoding formats for the same music snippet in staff notation.

MusicXML

MusicXML³ is one of the most widely used music encoding formats in digital music. Built on the extensible markup language (XML), it provides a textual representation of music scores that is both human-readable and machine-readable. MusicXML supports detailed score annotations, including notes,

³<https://www.musicxml.com/>

rhythms, and graphical elements for sheet music (e.g., engraving). However, it does not encompass every detail in a music score; for instance, it lacks full layout specifications and support for certain non-standard notations.

Despite its ability to encode rich musical information, MusicXML is rarely used in deep learning-based music research due to its verbosity, structural complexity, and redundancy. The format's non-uniform data structure and the challenges of parsing its extensive files further limit its applicability in such contexts.

Music Encoding Initiative (MEI)

The Music Encoding Initiative (MEI)⁴ aims to create a versatile digital representation format for encoding and sharing music documents. Like MusicXML, MEI uses XML to encode musical scores, metadata, and related information. However, while MusicXML primarily focuses on Common Western Music Notation (CWMN), MEI supports a broader range of musical notations, including mensural (Renaissance) and neume (Medieval) systems, as well as contemporary scores. This flexibility allows MEI to serve diverse applications, such as musicology, digital archives, score editing, and computational music analysis.

Unlike MusicXML, which is designed primarily for the interchange of notated scores between music notation software, MEI addresses more comprehensive scholarly needs. These include advanced score editing, detailed metadata annotation, and the representation of historical sources. MEI is also widely used in optical music recognition (OMR), where its structured and flexible format is ideal for storing and processing OMR outputs. Moreover, MEI supports the integration of notation with other modalities, such as images and audio recordings, enabling comprehensive documentation of musical works and their performance histories.

Despite its advantages, MEI faces similar challenges to MusicXML as an XML-based format. Its complexity, detailed focus on notation, and verbosity make it less suited for deep learning applications, which often require simpler, more streamlined data structures.

⁴<https://music-encoding.org/>

LilyPond

LilyPond (Nienhuys & Nieuwenhuizen, 2003) is a text-based music engraving system that uses a human-readable plain-text format, enabling users to write and edit music with a simple text editor. It is commonly employed for engraving scores in Common Western Music Notation (CWMN) across genres such as classical, jazz, and contemporary music. Its text-based encoding makes it particularly well-suited for programmatic score generation.

In MIR research, LilyPond has been used for tasks such as automatic music transcription (Carvalho & Smaragdis, 2017) and optical music recognition (OMR) (Schneider et al., 2021). However, its complexity in handling polyphonic and multi-instrumental music, along with its primary focus on visual presentation, can pose challenges for training machine learning models.

****kern**

The ****kern** format, developed as part of the Humdrum Toolkit⁵, is designed to represent essential information in Common Western Music Notation (CWMN). It encodes music scores as plain-text symbols, capturing elements such as pitch, duration, meter, and articulation. Compared to formats like LilyPond, ****kern** is simpler, more flexible, and compact, making it well-suited for applications such as musicology research, computational music analysis, optical music recognition, automatic music generation, and transcription tasks (Rahal et al., 2021; Alfaro-Contreras et al., 2024).

However, while ****kern** effectively encodes a wide range of musical score elements, it lacks the capability to capture expressive nuances of performance data. This limitation reduces its applicability for models focused on analysing or replicating expressive performance features.

2.3.2 MIDI-derived score representations

MIDI-derived score representations are most commonly used in music generation tasks, from the time-shift representation in (Oore et al., 2018) to newer

⁵<https://www.humdrum.org/guide/ch02/>

ones by [Zeng et al. \(2021\)](#) and [Hsiao et al. \(2021\)](#). In this section, we go through some of the MIDI-derived score representations in the literature.

Time-shift representation ([Oore et al., 2018](#))

In this work on piano music generation, a time-shift representation is used to encode the scores into one sequence. The work considers a MIDI excerpt as a sequence of events, each from one of the following:

1. 128 NOTE-ON events, corresponding to 128 different MIDI pitches;
2. 128 NOTE-OFF events, one for each MIDI pitch;
3. 125 TIME-SHIFT events, representing time shifts from 8ms to 1s, with a skip time of 8ms;
4. 32 VELOCITY events, which correspond to a velocity change that applies to the following notes.

The data representation is designed to model piano performance and thus does not cover metrical information. All the events in the data representation are considered as word tokens and converted into one-hot vectors for model training, ending up with a vocabulary size of $(128+128+125+32)=413$.

REMI representation ([Huang & Yang, 2020](#))

This work proposes to use the *Revamped MIDI-derived events* (REMI) to represent MIDI data following the way humans read the scores. They define the representation as events including:

1. 32 NOTE-VELOCITY events, where the velocity for notes is quantised into 32 bins;
2. 128 NOTE-ON events, each one corresponds to a MIDI pitch;
3. 64 NOTE-DURATION events, corresponding to different note durations from a 32nd note to two whole notes;
4. one BAR event and 16 POSITION events, where each bar is divided into 16 bins;

5. TEMPO events corresponding to tempo changes from 30 to 209 bpm;
6. 60 CHORD events, each representing a different chord change.

All the described events in one music excerpt are combined into a sequence, where each bar has a BAR event followed by several groups of metrical and note events. Metrical events follow the order of POSITION, CHORD, TEMPO events, where applicable, and note events are added in the order of NOTE-VELOCITY, NOTE-ON, NOTE-DURATION. Compared to the Time-shift representation proposed by [Oore et al. \(2018\)](#), REMI representation covers more information for music scores, including metrical information. It is also proven to be faster in learning and is better for expressive pop piano composition.

MuMIDI representation ([Ren et al., 2020](#))

This Multi-track MIDI (MuMIDI) representation is specifically designed for multi-instrument accompaniment generation. In order to model the dependencies between different instruments, the data representation puts score information for multiple instrument tracks into one sequence. This is different from what most multi-instrument accompaniment systems do. They tend to generate scores for different instruments separately. MuMIDI representation is composed of the following symbols:

1. <Bar> and 32 <Position> symbols, where each bar is divided into 32 bins and note onsets are quantised to the nearest bins;
2. 84 <Chord> symbols, considering 12 chord roots (C, C#, D, D#, E, F, F#, G, G#, A, A#, B) and 7 chord qualities (major, minor, diminished, augmented, major7, minor7, half_diminished);
3. 6 <Track> symbols for different instrument tracks;
4. note symbols including <Pitch>, <Velocity> and <Duration>, which are jointly predicted in one time step;
5. Meta symbols, including tempo and style.

This representation is similar to the REMI representation, both of which have notes encoded by their velocity, pitch and duration and have similar metrical symbols (bar and position) as well as the chord symbols. However, since the score information for multiple instrument tracks is squeezed into a single sequence, the sequence will be very long, and this will make it harder for models to learn. To shorten the sequence length, the paper proposed to predict the three note attributes in one time step. More specifically, they take the sum of the embeddings of the three note symbols as the embedding for the note, and use this embedding as the input to the token embedding for the corresponding time step. For prediction, they use multiple softmax matrices to generate the corresponding attributes for each time step. To better model the long-term dependencies, the work makes use of a recurrent Transformer encoder and recurrent Transformer decoder architecture similar to the Transformer-XL architecture (Dai et al., 2019). This allows the model to deal with an unlimited length of sequences. It turns out that the MuMIDI representation predicts more consistent harmonies than the REMI representation for multiple instrument tracks.

Compound Word representation (Hsiao et al., 2021)

This score representation is an upgraded version based on the REMI representation and the MuMIDI representation. Similar to the two, the Compound Word representation group the tokens into two types of “Compound Words”: Metrical-related word and Note-related word. A Metrical-related word covers a BAR token or a group of POSITON, CHORD and TEMPO tokens (if there are chord and tempo changes, otherwise they are set to IGNORE). A Note-related word is composed of a group of PITCH, DURATION and VELOCITY tokens. At each time step, the model predicts one “Compound Word” and this will not only greatly shorten the sequence length, but also offer a chance to explicitly take into account the different types of tokens. To better model the different token types, the work defines the data representation as having 7 tokens per time step, one family token corresponding to Metrical-related or Note-related symbols, three tokens for Metrical-related tokens and the remaining 3 tokens for Note-related tokens. For each time step, there will be at

least 3 empty tokens, which are set to be IGNORE.

To better model the type-specific information, 7 embedding layers and 7 feed-forward heads in the Transformer model are added to the different types of tokens. The embeddings for each of them are concatenated to a larger embedding vector to be used as the input for each time step. In order to better constrain the prediction of only Metrical-related or Note-related tokens, the model predicts tokens for each time step in two stages: first predict the family token, and then it predicts the Compound Word tokens given the family token. By using the Compound Word representation, the models converge 5-10 times faster than during training, compared to models using REMI or time-shift representations, with comparable quality in generated music.

OctupleMIDI encoding (Zeng et al., 2021)

This paper develops MusicBERT, a large-scale pre-trained model for music understanding. The data representation OctupleMIDI encodes music into a sequence of 8-element octuples, where each element represents one of the different characteristics of a musical note. Each of the eight elements is one of the following:

1. 254 TIME-SIGNATURE elements, the denominator is a power of two in the range of [1, 64] and the numerator is an integer in the range of [1, 128]. The duration of a bar is limited to no more than two whole notes. and otherwise it will be divided into several equal-duration bars;
2. 49 TEMPO elements, quantised from tempo values in the range of [16, 256]. The tempo values are in a geometric progression instead of an arithmetic progression;
3. 256 BAR elements and 128 POSITION elements. This supports music pieces with up to 256 bars. A granularity of a 64th note is used to represent the onset time of notes, starting from 0 in each bar, which corresponds to a total of 128 POSITION elements to represent positions within a bar no longer than two whole notes;
4. 129 INSTRUMENT elements;

5. 128 PITCH elements and 128 PITCH-PERCUSSION elements;
6. 128 DURATION elements, where a high resolution is used when the note duration is small and a low resolution is used when the note duration is large. For percussion instruments, the duration is set to be 0;
7. 32 VELOCITY elements, which are quantised with an interval of 4 in the range of [2, 126].

By covering the eight characteristics in a single time step, the work greatly reduces the length of a music sequence. OctupleMIDI encoding is four times shorter than the REMI representation and two times shorter than the Compound Word representation. It is also note-centric and can be easily adapted to cover more/less information on demand.

Since there are a lot of repetitions in the metrical-related elements, the paper proposed a bar-level masking strategy, which masks all the tokens of the same type in a bar to get rid of information leakage and help with effective representation learning.

2.4 Datasets and Evaluation Metrics

2.4.1 Datasets

As there is an increasing amount of exploration on deep learning methods for AMT, people are using larger datasets to train and evaluate the systems they developed. In the following, we review datasets that are commonly used for AMT problems in the literature:

- **RWC dataset** ([Goto et al., 2003](#))

The RWC dataset is a comprehensive collection of 315 musical pieces divided into six subsets, featuring a wide range of instruments such as guitar, vocal, drums, piano, trumpet, clarinet, and more. It encompasses diverse music styles, including classical, royalty-free, popular, and jazz. The dataset consists of real recordings with non-aligned MIDI files for most subsets. For the classical subset, an enhanced version called

SyncRWC provides automatically aligned MIDI annotations, making it particularly useful for tasks requiring precise synchronisation.

- **MAPS dataset** ([Emiya et al., 2010](#))

The MAPS dataset focuses exclusively on piano music and includes both classical pieces and non-musical material such as individual notes and chords. It features 30 compositions rendered across nine different piano synthesisers in the MUS subset. The dataset contains both synthesised and real piano recordings, offering a rich resource for piano-based studies. Additionally, rhythm and key annotations are provided in the extended A-MAPS dataset, enabling advanced analysis of piano performances.

- **Bach10** ([Duan et al., 2010](#))

The Bach10 dataset consists of 10 real recordings of four-part J.S. Bach chorales. The dataset includes individual instrument stems for violin, clarinet, saxophone, and bassoon, along with fundamental frequency (F0) annotations. Its compact size and detailed labelling make it a valuable resource for studying polyphonic music, especially in the context of classical chorale-style composition.

- **MedleyDB** ([Bittner et al., 2014](#))

MedleyDB comprises 196 royalty-free music tracks featuring a variety of instruments, including piano and vocals. The dataset is notable for its provision of individual stems for each instrument in the recordings, enabling detailed analysis and separation studies. Moreover, 108 of the tracks include melody annotations, making this dataset particularly useful for tasks like melody extraction and multi-instrument music analysis.

- **MusicNet** ([Thickstun et al., 2017](#))

The MusicNet dataset includes 330 classical music recordings performed under various conditions. It features multiple instruments such as piano, violin, and clarinet. Labels in the dataset are aligned using dynamic time warping and manually verified by trained musicians, resulting in a high-quality resource with an estimated labelling error rate of just

4%. MusicNet’s detailed annotations and diverse recordings make it a significant dataset for classical music analysis and modelling.

- **GuitarSet** ([Xi et al., 2018](#))

GuitarSet is a collection of 360 real recordings focused on popular music styles. The dataset exclusively features guitar performances, offering a rich resource for studying this instrument in depth. Its comprehensive scope makes it ideal for tasks like transcription, modelling, and style analysis specific to guitar-based music.

- **MAESTRO** ([Hawthorne et al., 2019](#))

The MAESTRO dataset is a large-scale collection of piano performances drawn from e-piano competitions, with a total duration of 201.2 hours across 1,282 pieces. The dataset primarily features classical music and provides high-quality recordings with fine temporal alignment. Its extensive size and detailed annotations make MAESTRO a valuable resource for tasks such as piano transcription, generation, and performance analysis.

- **Slakh** ([Manilow et al., 2019](#))

The Slakh dataset contains 2,100 synthesised tracks spanning various musical styles, including classical and popular music. It features multiple instruments such as piano, guitar, bass, and drums. The dataset is synthesised using MIDI data from the Lakh MIDI dataset, offering a controlled environment for studying music transcription, generation, and source separation in multi-instrument contexts.

- **ASAP** ([Foscarin et al., 2020](#))

The ASAP (Aligned Scores and Performances) dataset is a collection of piano performances aligned with their corresponding musical scores. It includes recordings of classical piano pieces with detailed performance data such as tempo, dynamics, and expressive timing. The dataset contains 236 distinct musical scores and 1067 performances of Western classical piano music from 15 different composers. It provides precise

alignments between the audio and the symbolic score, making it a valuable resource for tasks like automatic music transcription, performance analysis, and expressive modelling. Its emphasis on alignment and expressive features sets it apart as a significant contribution to piano-based music research.

Although there are plenty of choices of AMT datasets, there are relatively more datasets for piano transcription (given the ease in automatically exporting MIDI annotations from acoustic pianos when using specific piano models such as Disklavier or Bösendorfer), but much less for other instruments, especially non-Western instruments. The biggest challenge of collecting AMT datasets is that annotating music recordings requires a high degree of music expertise and is very time-consuming. Also, there might not be enough music pieces and recordings for some less popular traditional instruments when a large dataset is needed. Moreover, human-annotated transcription datasets are not guaranteed to have a high degree of temporal precision, which makes them less suitable for model evaluation on the frame and note level. [Su & Yang \(2015\)](#) proposed four aspects to evaluate the goodness of a dataset: generality, efficiency, cost and quality. They suggest that a good dataset should not be limited to a certain music form or recording conditions, should be fast-annotated, should be as low-cost as possible, and should be accurate enough. Because of the difficulty in collecting large human-transcribed datasets, researchers have used electronic instruments or acoustic instruments with sensors that can directly produce annotations (e.g., electronic piano, MAESTRO dataset), or synthesised datasets (e.g., Slakh) instead of real recordings. The use of synthesised recordings greatly speeds up dataset collection, but on the other hand, could introduce some bias in model training, limiting the generality of the developed AMT system.

2.4.2 Evaluation Metrics

Frame- and note-level transcription metrics

Beyond collecting datasets, model evaluation is another important process in developing methodologies for AMT problems. Evaluating a music transcrip-

tion can be difficult since there are various types of errors, from pitch errors to missing/extra notes, and each has a different influence on the final evaluation of results. Currently, common evaluation metrics for AMT systems focus mainly on frame/note level transcriptions (Bittner & Bosch, 2019; Bay et al., 2009; Bosch et al., 2016; Kelz et al., 2016; Hawthorne et al., 2018). Much less work has been done on stream and notation level transcriptions (McLeod & Yoshii, 2019; McLeod & Steedman, 2018a; Molina et al., 2014). In the 2019 annual Music Information Retrieval Evaluation eXchange (MIREX), there are three subtasks (MIREX) for music transcription for pitched-instruments - multiple fundamental frequency estimation on frame level, note tracking, and timbre tracking (multi-pitch streaming).

Common multiple fundamental frequency estimation methods (Bay et al., 2009) calculate frame-wise *precision* (P), *recall* (R) and relevant *F-measure* (F1 score, $F1$) values. The three scores are defined as:

$$P = \frac{TP}{TP + FP} \quad (2.1)$$

$$R = \frac{TP}{TP + FN} \quad (2.2)$$

$$F1 = \frac{2 \times P \times R}{P + R} \quad (2.3)$$

The TP , FP , and FN values correspond to *true positives*, *false positives* and *false negatives* respectively, and are calculated from all pitch values and time frames in the piano roll. There are also other methods for evaluating frame-wise transcription, such as separating different types of errors (e.g., missed pitches, extra pitches, false octaves) in multiple F0 estimation. A type-specific error rate is proposed by Poliner & Ellis (2007), where the authors defined a frame-level transcription error score combining different error types. Separating different error types can help with analysing the limitations of music transcription systems.

Note tracking problems usually define transcription results as sequences of notes, characterised by a pitch, onset and offset. A *tolerance* is defined to allow small errors in onset times since it is difficult to estimate the exact time when building an AMT dataset as well as transcribing music with an AMT system. A common *tolerance* is 50ms, which is used in the MIREX

note tracking subtask. There are also some other scenarios where offset times are included (e.g., a 20% tolerance for offset (Benetos & Holzapfel, 2013) and a tolerance of the larger one in 20% of the note length and 50ms (Cogliati et al., 2017)). For any of the above scenarios, note-level precision, recall and F-measure are calculated for a final evaluation. Similar to frame-level F0 estimation, researchers have attempted to include error types in evaluation metrics (Molina et al., 2014).

There are fewer works on *multi-pitch streaming*. The evaluation for *multi-pitch streaming* uses similar metrics like precision and recall. Gómez & Bonada (2013) proposed a simple method of calculating accuracy and false rate to evaluate voice streaming applied to A Capella transcription. In 2014, Duan et al. (2014) used a similar evaluation method to calculate a more general multi-pitch streaming accuracy. The accuracy is defined as:

$$accuracy = \frac{TP}{TP + FP + FN} \quad (2.4)$$

Another work by Molina et al. (2014) proposed to include types of errors in streaming process, and used a standard precision-recall metric.

Score-level transcription metrics

Recent years have seen some introduction of evaluation metrics for *complete music transcription*, given a recent increase in methods that directly transcribe audio to music scores (Cogliati & Duan, 2017; McLeod & Steedman, 2018a; McLeod & Yoshii, 2019). A recent approach for evaluating score transcriptions is proposed by McLeod & Yoshii (2019), which is based on a previous approach called *MV2H* (representing **M**ulti-pitch detection, **V**oice separation, **M**etrical alignment, note **V**alue detection, and **H**armonic analysis) (McLeod & Steedman, 2018a). According to this metric, a score is calculated for each of the five aspects, then the scores are combined into a joint evaluation following the principle that one mistake should not be penalised more than once. More recently, Nakamura and Hiramatsu have proposed *MUSTER*, which is an edit-distance-based metric for evaluating score-level transcription in MusicXML format (Nakamura et al., 2018; Hiramatsu et al., 2021). Similar to the word error rate (WER) used for evaluating automatic speech recognition

systems, MUSTER introduces six sub-metrics (i.e., pitch error rate, missing note rate, extra note rate, onset-time error rate, offset-time error rate, and voice error rate), where each sub-metric evaluates a specific aspect of the music score.

While most evaluation metrics are based on music theory and simple statistical analysis, there are some metrics that contain some considerations of human perception of music transcriptions. In 2008, [Daniel et al. \(2008\)](#) explored the differences of some error types in AMT from the aspect of human perception, and proposed a modified evaluation metric that weights different error types.

2.5 State of the art

In this section, we look into state-of-the-art methodologies for AMT, mainly focusing on Neural Network methods. The section will be structured as follows. In [subsection 2.5.1](#), we provide an overview of the development and common methods for AMT, followed by [subsection 2.5.2](#), where we discuss Neural Network methods used in AMT. The following sections cover more specific topics within AMT: we give a review on *multi-task learning methods* for AMT in [subsection 2.5.3](#); and the use of *music language models* and related works are covered in [subsection 2.5.4](#).

2.5.1 Overview

As the field of MIR has evolved over the past 20 years since the inception of the International Symposium on Music Information Retrieval (ISMIR), so has the topic of AMT. Roughly, proposed methods for AMT in the early 2000's made use of signal processing and statistical machine learning theory ([Klapuri & Davy, 2006](#)). Following the seminal paper by [Smaragdis & Brown \(2003\)](#) on the potential of non-negative matrix factorisation, when applied to the problem of AMT, a series of different methods were proposed for AMT that made use of matrix decomposition approaches. In the early 2010's, following the rise of deep learning methods and the paper by [Humphrey et al. \(2018\)](#) advocating for the use of deep learning methods for MIR, neural network-

based methods started being widely used for AMT and are still in use to date.

In terms of AMT subtasks to be addressed, the vast majority of methods have been and still do focus on (framewise) multi-pitch detection, with a smaller proportion of methods focusing on note tracking or rhythm transcription/typesetting. Due to the emergence of end-to-end deep learning methods for AMT, an increasing trend towards systems producing higher-level representations (such as outputs in MIDI format or in staff notation) can be observed (Benetos et al., 2019). The problem of timbre tracking/instrument assignment is, however, still under-explored.

Current literature for AMT includes a mixture of deep learning and matrix decomposition approaches, with deep learning methods currently being used in the majority of scenarios. Compared to other tasks in MIR, a large proportion of methods still employ matrix decomposition approaches (Benetos et al., 2019), due to their ability to work with limited data, fast learning and inference, and due to the models' interpretability. The remainder of this chapter will focus more on neural network-based methods for AMT, due to their increasing popularity in the research community and also due to certain methodological challenges when using deep learning methods for AMT that are still to be addressed.

2.5.2 Neural Networks for AMT

Research in AMT has increasingly been relying on deep learning models, which use feedforward, recurrent, and convolutional layers as main architectural blocks. An early example of a deep neural model applied to AMT is the work of Nam et al. (2011), which uses a Deep Belief Network (DBN) in order to learn representations for a polyphonic piano transcription task. The resulting learned features are then fed to a Support Vector Machine (SVM) classifier in order to produce a final decision. Another notable early work that made use of deep neural architectures was by Böck & Schedl (2012), where the authors used a bi-directional Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) units, applied to the task of polyphonic piano transcription. Two points are particularly worth mentioning for the work by

Böck & Schedl (2012): (i) the use of two STFT magnitude spectrograms with different window sizes as inputs to the network, in order to achieve both a “good temporal precision and a sufficient frequency resolution”; (ii) The output is a piano-roll representation of note onsets and corresponding pitches, and does not include information on note durations/offsets.

A first systematic study towards the use of various neural network architectures for AMT was done by Sigtia et al. (2016). The study compared networks for polyphonic piano transcription that used feedforward, recurrent, and convolutional layers (noting that layer types were not combined), all using a Constant-Q Transform (CQT) spectrogram as input time-frequency representation. Results by Sigtia et al. (2016) showed that networks that include convolutional layers reported the best results for the task, which is also in line with other results reported in the literature, and with current methodological trends related to neural networks for AMT. The ability of Convolutional Neural Networks (CNNs) to function well for tasks related to multi-pitch detection and AMT stems from the useful property of shift-invariance in log-frequency representations such as the CQT: a convolutional kernel that is shifted across the log-frequency axis can capture spectro-temporal patterns that are common across multiple pitches.

Following the work by Sigtia et al. (2016), Kelz et al. (2016) showed the potential of simple frame-based approaches for polyphonic piano transcription using an architecture similar to the architecture proposed by Sigtia et al. (2016), but making use of up-to-date training techniques, regularizers, and taking into account hyper-parameter tuning. The “ConvNet” architecture (Kelz et al., 2016) can be seen in Figure 2.7.

An influential work that used CNNs for multiple fundamental frequency estimation in polyphonic music was the *deep saliency* representation proposed by Bittner et al. (2017). Contrary to most methods in AMT that produce a binary output, the model (Bittner et al., 2017) produces a non-binary time-pitch representation at 20 cent pitch resolution, which can be useful for both AMT applications and for several downstream applications in the broader field of MIR. A particular contribution of this work was the use of a Harmonic Constant-Q Transform (HCQT) as input representation; the HCQT is a three-

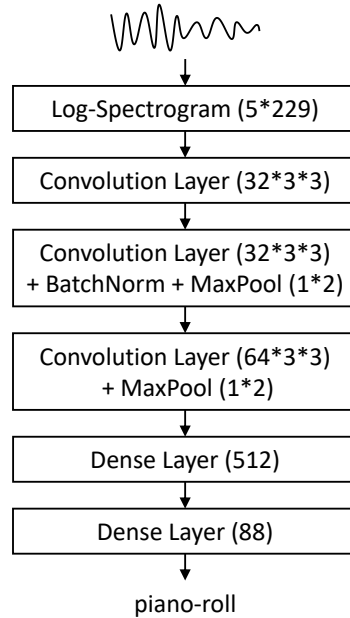


Figure 2.7: Model architecture for the convolutional neural network by [Kelz et al. \(2016\)](#) for polyphonic piano transcription. The depicted network corresponds to the “ConvNet” architecture by [Kelz et al. \(2016\)](#).

dimensional representation over frequency, time and harmonic, produced by computing several versions of the CQT by scaling the minimum frequency used by a harmonic. [Figure 2.8](#) shows the pitch salience representation for the Mozart excerpt of [Figure 2.1](#), computed using the deep salience method ([Bittner et al., 2017](#)).

The ability of CNNs in learning features in time or time-frequency representations keeps them still active in the AMT literature. This includes the work of [Thickstun et al. \(2017\)](#) that was carried out as part of the MusicNet dataset, and compared feedforward and convolutional networks learned on raw audio inputs, as opposed to having a time-frequency representation as input. It is worth noting, however that convolutional, and more broadly neural networks, when trained for AMT as a multi-label classification task, face the issue that they appear to learn combinations of notes exposed to them

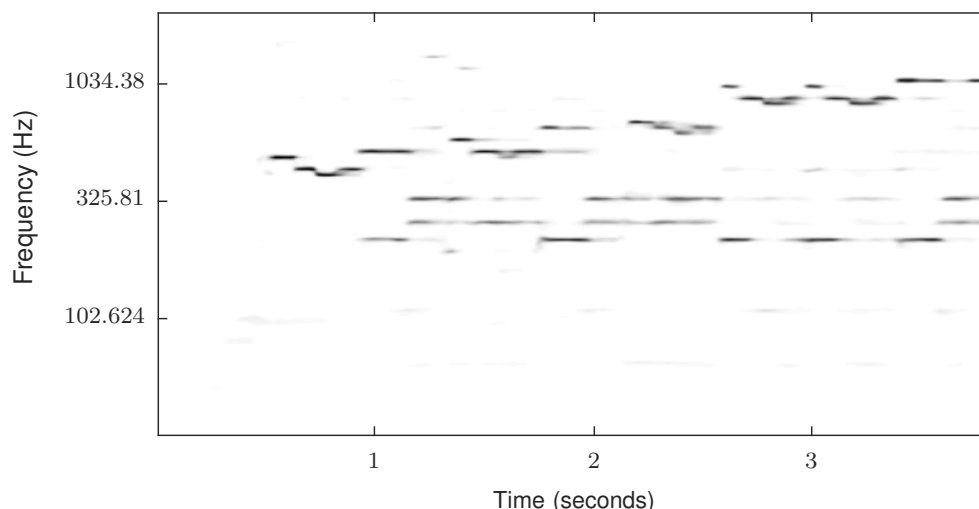


Figure 2.8: Pitch salience representation for the excerpt of [Figure 2.1](#), using the deep salience method ([Bittner et al., 2017](#)).

during training, and are not able to generalise unseen combinations of notes – the so-called *entanglement problem* as discussed by ([Kelz & Widmer, 2017](#)).

2.5.3 Multi-task learning methods

Recent research in machine learning has focused on *multi-task learning* ([Ruder, 2017](#)), where multiple learning tasks are addressed jointly, thus exploiting task similarities and differences. In the context of AMT, multi-task learning has been shown to improve transcription performance in certain cases. Tasks related to AMT, such as note-level transcription, onset detection, melody estimation, bass line prediction and multi-pitch detection (sharing chroma and rhythm features) can be integrated into one model that would exploit task interdependencies.

In the “Onsets and Frames” system by [Hawthorne et al. \(2018\)](#), which is currently considered the benchmark in automatic piano transcription, the authors used a deep Convolutional Recurrent Neural Network (CRNN) to jointly predict onsets and multiple pitches. The onset detection results are fed back

into the model for further improving frame-wise multi-pitch predictions. The Onsets and Frames model was further improved in the work of [Kim & Bello \(2019\)](#), which addresses the problem of expressing inter-label dependencies through an adversarial learning scheme.

[Bittner et al. \(2018\)](#) proposed a multi-task model that jointly estimates outputs for several AMT-related tasks, including multiple fundamental frequency estimation, melody, vocal and bass line estimation. The authors show that the more tasks included in the model, the higher the performance, and that the multi-task model outperforms the single-task equivalents. In another recent work ([Kelz et al., 2019a](#)), the authors designed a multi-task model with CNNs which enables four different transcription subtasks: multiple-f₀ estimation, melody estimation, bass estimation, and vocal estimation. Results on the method proposed by [Kelz et al. \(2019a\)](#) showed an overall improvement in the multi-task model compared to single task models.

2.5.4 Early-stage music language models

Inspired by work in the field of speech processing, where many systems for Automatic Speech Recognition (ASR) benefit from language models that predict the occurrence of a word or phoneme ([Jurafsky & Martin, 2008](#)), researchers in MIR have recently attempted to use Music Language Models (MLMs) and combine them with acoustic models in order to improve AMT performance. While the problem of polyphonic music prediction using statistical machine learning models (such as n-grams and hidden Markov models) is not trivial, the emergence of neural network methods for high-dimensional sequence prediction has enabled the use of MLMs for polyphonic music.

One of the first works to use neural network-based MLMs for polyphonic music prediction and combine them with multi-pitch detection was carried out by [Boulanger-Lewandowski et al. \(2012\)](#). The MLM was based on a combination of a recurrent neural network with a Neural Autoregressive Distribution Estimator (NADE). The same RNN-NADE music language model was also used by [Sigtia et al. \(2016\)](#), which was combined with a CNN as the acoustic model, showing that the inclusion of an MLM can improve transcription performance.

It was shown, however, that the MLMs which operate at the level of a small time frame (e.g., 10 msec) are only able to produce a smoothing effect in the resulting transcription (Ycart & Benetos, 2017). More recently, Wang et al. (2018) used an LSTM-RBM language model as part of their proposed transcription system, but each frame corresponds to an inter-onset interval as opposed to a fixed temporal duration, resulting in improved transcription performance when using note-based metrics. Finally, Ycart et al. (2019) combined an LSTM-based music language model with a feedforward neural blending model, which combines the MLM probabilities with the acoustic model probabilities. In line with past observations, the blending and language models work best when musically-relevant time steps are used (in this case, time steps corresponding to a 16th note).

2.6 More on Deep Learning in Sequential Problems

In this section, we discuss a little more about the use of sequence-to-sequence models and attention mechanisms in sequential problems, since they play a crucial role in the development of our proposed methods. We will briefly introduce basic sequence model structures, including sequence-to-sequence models and the attention mechanism.

2.6.1 RNN-based sequence-to-sequence models

Recurrent neural networks are limited to pre-aligned input and output sequences, but in a more general situation of sequence transduction, we need to deal with non-aligned input/output sequence pairs, and sometimes even free-length output sequences. To solve this problem, people have proposed different approaches. One of the early solutions was the sequence “Transducer” proposed by Graves et al. (2006) and later improved by Graves (2012), which uses connectionist temporal classification (CTC) to label non-aligned sequences. A null symbol is used in an intermediate stage to link an aligned output to the final non-aligned output sequence. Thus, this method is limited to labelling a monotonic sequence, and the output sequence cannot be longer than the original aligned sequence.

Another solution fits a more general situation of free-length sequence transduction. [Sutskever et al. \(2014\)](#) proposed the use of a *sequence-to-sequence* (seq2seq) model. The seq2seq model uses two RNN networks, one as an encoder, another as a decoder. The encoder maps an input sequence into a fixed-dimensional latent vector, and the decoder decodes the vector into an output sequence. This architecture is proven to be useful in many applications, including machine translation ([Sutskever et al., 2014](#); [Bahar et al., 2018](#); [Luong et al., 2016](#)), automatic speech recognition ([Bahar et al., 2019](#)), syntactic constituency parsing ([Vinyals et al., 2015](#)), image captioning ([Zhao et al., 2016](#)) and automatic music transcription ([Ullrich & van der Wel, 2018](#); [Nishikimi et al., 2019](#)). It is worth mentioning that [Sutskever et al. \(2014\)](#) found that by reversing the order of the input sequence, the system’s performance can improve markedly, since the operation introduces many short-term dependencies between the input and output sequences. The seq2seq model has been shown to be effective in improving transcription performance. However, it is still a challenge to improve the performance of the seq2seq model, especially for long sequences.

2.6.2 Early-stage attention mechanisms

Following the seq2seq model, people have proposed to use attention mechanisms to improve model performance. This early-stage exploration of attention mechanism brings wide discussions in the community and were explored in various tasks.

[Chorowski et al. \(2014\)](#) proposed content-based alignment methods to connect input and output sequences. This idea was further explored by [Bahdanau et al. \(2015\)](#), where a penalty was applied to alignments that map to pre-considered inputs, preferring monotonic alignment.

Following this, the addition-based attention was proposed by [Bahdanau et al. \(2015\)](#), and became widely used before the multi-head self-attention mechanism in the Transformer architecture was introduced ([Vaswani et al., 2017](#)). To deal with long utterances, [Chorowski et al. \(2015\)](#) further extended the original content-based mechanism to be location-aware, or more precisely, to take into account the alignment in the previous time steps. In the same

year, [Chan et al. \(2015\)](#) proposed a “Listen, Attend and Spell” model that uses a pyramid structured bi-LSTM encoder (Listener) and an attention-based decoder (Speller) for character-wise speech recognition.

Later, attention and related ideas were further used in various tasks. [Bahar et al. \(2018\)](#) proposed a two-dimensional sequence-to-sequence model to model the dependency between input and output sequences, and tested it in machine translation and automatic speech recognition tasks. Beyond sequential problems, the idea of “attend” to some parts of data was also used in computer vision ([Zhang et al., 2019](#)). Attention has become a commonly used mechanism in neural machine translation ([Bahdanau et al., 2015](#)), automatic speech recognition ([Chorowski et al., 2014, 2015](#)), sentiment analysis ([Shen et al., 2018](#)), and automatic music transcription ([Nishikimi et al., 2019](#)).

2.6.3 Multi-head self-attention and the Transformer architecture

A general attention mechanism for modelling dependencies between input and output sequences usually uses query, key, and value representations to calculate a context vector that indicates how much attention the current output should pay to each input position. This is further extended into *multi-head self-attention* by [Vaswani et al. \(2017\)](#).

In the self-attention setting, for an input sequence representation H , the model computes a query matrix $Q = HW^Q$, a key matrix $K = HW^K$, and a value matrix $V = HW^V$, and then uses scaled dot-product attention to calculate the attention weights,

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V. \quad (2.5)$$

The resulting weighted sum gives a context-aware representation at each time step. Multi-head self-attention computes this operation in parallel with multiple parameter sets,

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \quad (2.6)$$

and then combines them as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O. \quad (2.7)$$

In the Transformer architecture, attention is not only used between encoder and decoder, but also within the encoder and decoder as self-attention layers combined with feed-forward networks. At the same time, they choose to calculate not only one set of attention weights, but multiple sets of attention weights within each self-attention, to allow a word to attend to different words in a sentence.

The idea of multi-head self-attention is further explored in various works, such as (Shen et al., 2018; Salazar et al., 2019; Yang et al., 2019; Voita et al., 2019). Shen et al. (2018) proposed DiSAN, which uses directional self-attention to model token dependencies without RNN/CNN recurrence. Salazar et al. (2019) applied self-attention networks to CTC-based speech recognition, showing that self-attention can be used as an acoustic sequence model in end-to-end automatic speech recognition. Yang et al. (2019) proposed convolutional self-attention networks, combining convolutional layers and self-attention to capture both local and global dependencies. Voita et al. (2019) analysed multi-head self-attention in neural machine translation and showed that a subset of specialised heads contributes most of the useful behaviour, while many other heads can be pruned with limited performance degradation.

Recent AMT studies have also seen an increased use of the Transformer architecture. For note-level piano transcription and multi-instrument transcription, Hawthorne et al. (2021) proposed a sequence-to-sequence Transformer for piano transcription, Gardner et al. (2022) extended this direction to multi-instrument transcription, Toyama et al. (2023) introduced a hierarchical frequency-time Transformer to model dependencies across both frequency and time axes, and recent systems further adopted this architecture (Yan & Duan, 2024; Chang et al., 2024). For audio-to-score transcription, (Alfaro-Contreras et al., 2024) proposed to use the Transformer architecture for polyphonic music transcription.

2.7 Conclusions

AMT is a core problem in the field of Music Information Retrieval (MIR), and has attracted a lot of attention during the past few decades. In this

chapter, we review and discuss some of the main topics within the problem of AMT. We make a concrete definition of the problem of AMT, and describe the main subtasks in the AMT process (see [subsection 2.2.1](#)). We also introduce the problem of complete transcription, which refers to the process of converting music audio into a music score representation. We review commonly used datasets and evaluation metrics for AMT (see [Section 2.4](#)), and look into the state-of-the-art methodologies used in AMT (see [Section 2.5](#)). Current research on AMT has focused on methods using neural networks with promising results. We look into several topics in particular, including the use of commonly used neural network architectures, the use of multi-task learning methods, the use of music language models, and methods for complete transcription.

Chapter 3

Data Collection and Preparation

3.1 Introduction

Due to several difficulties in dataset collection and annotation mentioned in [subsection 2.4.1](#) of [Chapter 2](#), we worked on preparing data for audio-to-score piano transcription as a first step of our project.

In this chapter, we will introduce the two datasets we prepared and used during the PhD project: 1) the MuseSyn dataset ([Liu et al., 2021b](#)) and 2) the ACPAS dataset ([Liu et al., 2021a](#)). For reproducibility purposes, we have made both datasets accessible online:

- MuseSyn dataset: <https://zenodo.org/record/4527460>
- ACPAS dataset: https://cheriell.github.io/research/ACPAS_dataset

We decided to collect the two datasets because at the time of starting this PhD project, there were no publicly available datasets that provided music score ground truth for score-level transcription tasks. Previous research either used internal datasets or synthetic datasets from musical scores ([Carvalho & Smaragdis, 2017](#); [Nakamura et al., 2018](#)). The two datasets we collected represent different music styles and levels of expressive performances. Here, we consider music tracks with “expressive performance” to be those performances

with tempo and dynamic changes, whether introduced by real human performance or by editing to simulate real human performance. In this scenario, the MuseSyn dataset contains only music pieces collected and synthesized from music scores. So, it does not contain any “expressive performance”. The MuseSyn dataset is composed of popular music and a few simplified classical music pieces without expressive performance, and the ACPAS dataset covers classical music with expressive performance. The difference between the two datasets allows us to better investigate the ability of the transcription systems at different levels of transcription difficulty.

In the remainder of this chapter, we describe the dataset contents in [Section 3.2](#) and the dataset statistics in [Section 3.3](#). Lastly, conclusions follow in [Section 3.4](#). Associated publications with this chapter are ([Liu et al., 2021b](#)) and ([Liu et al., 2021a](#)).

3.2 Dataset content

3.2.1 MuseSyn dataset: A dataset synthesised from piano sheet music collected at the MuseScore website

MuseSyn is a dataset (Syn)thesized from piano sheet music collected from the (Muse)Score website¹. Due to the lack of automatic music transcription datasets that provide music score ground truth on both physical and musical time, we collected 210 scores of popular or simplified classical piano music in MusicXML format, converted them into MIDI files, and synthesized the audio recordings using the MIDIs. Some examples of the music pieces we collected include “All Falls Down” by Alan Walker, “A Sky Full of Stars” by Coldplay and a simplified piano version of “Canon in D”. For efficient conversion from MusicXML to MIDI format, we use the Batch Convert plugin for MuseScore.²

During the synthesis process, we used four piano models provided by the Native Instruments Kontakt Player³ in order to provide some variations in

¹MuseScore sheet music for piano: <https://musescore.com/hub/piano>

²Batch Convert plugin: <https://musescore.org/en/project/batch-convert>

³Kontakt 6 Player: <https://www.native-instruments.com/en/products/komplete/samplers/kontakt-6-player/>

the piano font and allow training and evaluation on different piano fonts. The synthesized audio recordings are in .flac format, with a sample rate of 44.1kHz and a bit depth of 16 bits. To speed up the synthesis procedure, we used the python-reapy⁴ package to automate the music synthesis using Reaper⁵. The code of our automatic synthesis pipeline has been made available online for reuse at <https://github.com/cheriell/MuseSyn-dataset-synthesis>.

The MuseSyn dataset contains music pieces organised in three formats: 1) .flac audio recordings; 2) MIDI scores; and 3) MusicXML scores. As the audio files were synthesised directly from the scores, they are synchronised in physical times. It is important to mention that these audio recordings have a constant tempo and lack dynamics, as they are a direct synthesis of the music scores. The MuseSyn dataset comprises music scores presented in piano grand staff format, encompassing diverse key and time signatures. The dataset includes notes limited to semibreve, minim, crotchet, quaver, semiquaver, and demisemiquaver. To keep the scores simple, the scores do not contain grace notes, triplets, arpeggios, or trios. We manually checked each collected music score to remove them if there were any.

We adopt a training/validation/testing ratio of 8:1:1 for dataset splitting, resulting in 168 pieces for training, 21 pieces for validation, and 21 pieces for testing. Each music piece is assigned to one subset only to avoid overlap between subsets. During the synthesis process, we utilize four piano fonts, of which three piano fonts are assigned for training and validation, while all four piano fonts are used for testing.

As MuseSyn is entirely synthetic, there is a risk of overfitting to its synthetic profile. Models trained on this dataset may not generalise well to real-world performances, which contain expressive timing and dynamic variations absent from MuseSyn. To compensate for this limitation, we introduce the ACPAS dataset, which includes both synthesised and real expressive piano performances, allowing models to be further trained and evaluated in more realistic conditions.

⁴python-reapy: <https://pypi.org/project/python-reapy/>

⁵Reaper: <https://www.reaper.fm/>

3.2.2 ACPAS dataset: A dataset of aligned classical piano audio and scores

ACPAS is a dataset of **A**ligned **C**lassical **P**iano **A**udio and **S**cores for Audio-to-Score (A2S) transcription research. It is created from three sources: 1) the ASAP dataset (Foscarin et al., 2020); 2) the MAPS (Emiya et al., 2010) dataset, with score annotations provided in the A-MAPS dataset (Ycart et al., 2018); and 3) the Classical Piano MIDI (CPM) database (Krueger, 2018). The reason we collected data from multiple sources is to cover more music performances which are available in the ASAP dataset, and data with accurate annotations for model evaluation which can be obtained from the MAPS and CPM sources. While all the pieces in the ACPAS dataset consist of expressive performances, the ones from the ASAP dataset comprise real performances, and the ones from the other two sources are manually tuned performances by musicians, aimed at simulating human-like performance.

Here we present a summary of the three data sources utilized to compile the ACPAS dataset:

- **ASAP dataset.** The ASAP dataset (Foscarin et al., 2020) is a dataset of aligned music scores and performances with downbeat, beat, time signature, and key signature annotations. The dataset offers 1067 music performances in both audio and MIDI formats, along with 236 distinct music scores provided in MIDI and MusicXML formats. The performances and scores are aligned at beat level. Music performances are collected from the MAESTRO dataset v2.0.0 (Hawthorne et al., 2019) which are performed by pianists in the International Piano-e-Competition. Music scores are written by amateur musicians. Beat and downbeat annotations are created from an automatic annotation workflow based on alignment and mapping algorithms with a final manual correction step.
- **MAPS dataset.** The MAPS dataset (Emiya et al., 2010) is a dataset with 270 MIDI-annotated piano recordings from 158 distinct music scores. The MIDI files are obtained from the CPM database, which will be described in the next paragraph. The recordings are partly synthesised

with different soundfonts and partly recorded from a Yamaha Disklavier. The original MAPS dataset contains several subsets, including isolated notes and monophonic excerpts, random chords, common chords, and pieces of music. In our work, we merely take the data from a subset of pieces of music. Nonetheless, the MIDI annotations available in the MAPS dataset are restricted to physical time only, which does not meet the needs of our audio-to-score transcription task. Consequently, we opt to utilise the MIDI scores from the Augmented-MAPS dataset (Ycart et al., 2018), which provide rhythm, time signature, and key signature annotations for the music pieces in the MAPS dataset.

- **Classical Piano-Midi Page.** The classical piano Midi (CPM) database (CPMDatabase, 2021) is a set of MIDI performances of 337 classical piano music pieces manually tuned on music scores to simulate human-like performances. As a result, the MIDI files can work as a score-level annotation of the piano performances. A part of the musical pieces from the CPM database was incorporated into the MAPS dataset. However, the CPM database has been continuously updated by its creator over the years, so some of the pieces are now different from the versions used in the creation of MAPS. We thus include all up-to-date pieces provided in the CPM database in our ACPAS dataset.⁶ We further synthesise these pieces using different piano fonts to introduce greater variability in the audio samples.

To address the absence of audio recordings in the CPM database and expand the sampling space, we generate audio recordings by synthesising each MIDI performance gathered from the aforementioned sources. We use the same automatic synthesis pipeline used in the creation of the MuseSyn dataset. During the synthesis process, we introduce randomness by selecting one of the four piano fonts available in the Native Instrument Kontakt Player. We apply random adjustments to the piano fonts to create variations in hardness or softness across performances. Additionally, we incorporate a level of reverberation to simulate real recording conditions. This combination of random

⁶Accessed in August 2021. A copy of the accessed version is available at <https://github.com/cheriell/ClassicalPianoMIDI-dataset>

choices and adjustments enhances the diversity and realism of the generated audio recordings. The synthetic audio recordings are rendered into monaural .wav files with a sample rate of 44.1kHz and a bit depth of 16 bits.

Combining data from all the aforementioned resources, we draw a dataset with 497 distinct music pieces and 179.8 hours of 2189 audio performances. Owing to the diverse data sources, the music performances can be categorised as either genuine human performances (sourced from the ASAP dataset) or synthesised performances with carefully crafted tempos and dynamics to emulate human-like renditions (derived from the MAPS dataset and the CPM database). The music audio recordings in the dataset are a combination of real audio recordings sourced from the ASAP dataset and the MAPS dataset, as well as synthetic recordings generated both from the MAPS dataset and through our own synthesis process. The expressive timing deviations present across the ACPAS dataset provide a more challenging scenario for transcription systems to handle. They allow models to learn to handle expressive performances and enable a more realistic examination of transcription system capabilities in real-world scenarios.

The ACPAS dataset comprises several components for each music performance. These components include: 1) audio recording of the performance, 2) MIDI performance recording, 3) MIDI score, and 4) a TSV annotation file containing beat, downbeat, time signature, and key signature annotations. For performances from the MAPS dataset or the CPM database, the MIDI performance and MIDI score are combined into a single MIDI file, annotated with temporal and dynamic changes. For performances from the ASAP dataset, the MIDI scores are adjusted to match the annotated beats, key, and time signatures by updating the tick and tempo in the MIDI files. This allows direct use of the MIDI scores as ground truth for certain audio-to-score transcription evaluation metrics (e.g. the MV2H metric (McLeod & Steedman, 2018a; McLeod, 2019)).

The alignment between performances and symbolic scores operates at different levels depending on the data source. For performances from the MAPS dataset and the CPM database, the alignment is exact: these performances are MIDI files manually tuned from musical scores to simulate human-like

performances rather than genuine live recordings, so performance and score correspond perfectly. For performances from the ASAP dataset, the alignment may contain errors arising from two sources: (1) these are real human piano performances which do not always strictly follow the written score, owing to expressive interpretation choices and performance errors; (2) the alignment between performance and score is produced by automatic alignment algorithms, which are imperfect and can introduce additional mismatches.

The TSV annotation file provides synchronised beat and downbeat annotations for both the music performance (audio and MIDI) and the score (MIDI). The format of the TSV file is similar to that in the ASAP dataset, containing a list of beat times and their corresponding labels, including beat annotation (e.g., “b” for beat, “db” for downbeat, and “bR” for an estimated beat with rubato), time signature annotation in string format (e.g., “4/4”), and key signature annotation as the number of sharps (positive) or flats (negative). These metadata annotations are carried over directly from the source datasets; no independent re-validation was performed during dataset construction, so their accuracy relies on the source datasets. Unlike the MuseSyn dataset, the ACPAS dataset encompasses a wide range of notes and playing techniques commonly found in classical music.

We divide the ACPAS dataset into a Real recording subset and a Synthetic subset. The Real recording subset covers 578 real recording performances from two sources: 1) the two real recording subsets, namely “ENSTDkCl” and “ENSTDkAm”, sourced from the MAPS dataset and the corresponding MIDI scores from the A-MAPS dataset, and 2) the performances from the ASAP dataset with audio recordings from the MAESTRO dataset. The Synthetic subset consists of 1611 performances with synthetic audio recordings derived from three distinct sources: 1) Performances from the synthetic subsets within the MAPS dataset, accompanied by MIDI scores from the A-MAPS dataset; 2) MIDI performances and scores sourced from the ASAP dataset. The audio files are synthesised from performance MIDIs using Native Instrument Kontakt Player; and 3) MIDI performances and scores sourced from the CPM database. The audio files are synthesised from performance MIDI using Native Instrument Kontakt Player.

We established a pre-defined training/validation/testing split for the ACPAS dataset, ensuring that there is no overlap of music pieces among the splits throughout the entire dataset. To achieve this, we consider existing train/test splits from the MAPS and MAESTRO datasets. The provided split guarantees that no test piece from the MAPS and MAESTRO datasets is included in the training or validation split of the ACPAS dataset. This strategic approach allows models trained on the ACPAS dataset to be effectively evaluated using the test sets of MAPS and MAESTRO. In particular, we designate the test set of the MAPS dataset to consist of the two real recording subsets, “ENSTDkC” and “ENSTDkAm,” which are commonly used for testing. Similarly, for the MAESTRO dataset, we consider the test split provided in v2.0.0. Once we have reserved all the test pieces that appear in the MAPS and MAESTRO datasets, the remaining pieces are randomly divided into the training/validation sets. During the synthesis process, we reserve one piano font exclusively for synthesising test pieces. The other three piano fonts are utilised across all three splits (training, validation, and testing). This allocation ensures an effective and unbiased evaluation by including new piano soundfonts in the testing set. Individual splits for each music piece and music performance are provided in the dataset metadata. We provide two sets of metadata: a list of all the distinct music pieces and metadata for each performance. The latter covers information including distinct piece ID, composer, title, performance duration, and path to the audio and MIDI files.

Despite its scale, the ACPAS dataset has certain limitations in terms of stylistic diversity. The audio recordings are not diverse enough to fully represent real-world recording conditions: they consist of synthesised recordings and real recordings from only two sources, namely the MAPS dataset (recorded on a Yamaha Disklavier) and the ASAP dataset (sourced from a Yamaha E-Piano Competition, recorded on the same piano over the years). Furthermore, the musical styles in the dataset are limited to classical piano music, which does not cover the full breadth of piano music encountered in real-world scenarios, such as popular, jazz, or blues styles. These limitations in diversity constrain the ability of models trained on ACPAS to generalise to unseen expressive performances. In addition, the dataset size, while sufficient

Table 3.1: Dataset statistics across splits. Performances from the MAPS dataset are not counted towards distinct MIDI performance since they are derived from the CPM database. Pf: piano font.

| Split | Dist. Score | Audio Performance | | Dist. MIDI Performance | |
|------------------------|-------------|-------------------|--------------|------------------------|--------------|
| | | No. | Duration (h) | No. | Duration (h) |
| <i>MuseSyn dataset</i> | | | | | |
| training | 168 | 168×3 Pf | 23.0 | 168 | 7.7 |
| validation | 21 | 21×3 Pf | 3.0 | 21 | 1.0 |
| testing | 21 | 21×4 Pf | 4.1 | 21 | 1.0 |
| Total | 210 | 651 | 30.0 | 210 | 9.7 |
| <i>ACPAS dataset</i> | | | | | |
| training | 359 | 1523 | 127.7 | 1137 | 94.4 |
| validation | 49 | 184 | 11.2 | 110 | 7.8 |
| testing | 89 | 482 | 40.9 | 157 | 15.2 |
| Total | 497 | 2189 | 179.8 | 1404 | 117.4 |

for developing an initial framework for audio-to-score transcription, may not provide enough training data for models to generalise broadly to real-world scenarios.

3.3 Dataset statistics

3.3.1 Dataset splits

Table 3.1 presents an overview of the two datasets’ statistics categorised by training, validation, and testing splits. Notably, the MuseSyn dataset contains less than 10 hours of distinct MIDI performances, whereas the ACPAS dataset is significantly larger, comprising over 100 hours of distinct MIDI performances. The audio recordings in both datasets exhibit longer durations compared to the MIDI performances, owing to differences in recording conditions. In the ACPAS dataset, the number of distinct scores is fewer than the MIDI performances due to the inclusion of multiple interpretations of the same music piece by different pianists.

Table 3.2: Comparison on different AMT datasets.

| Dataset | Dist. Score | Audio Perf. | | Dist. MIDI Perf. | |
|----------------|-------------|-------------|--------------|------------------|--------------|
| | | No. | Duration (h) | No. | Duration (h) |
| MAPS | X | 270 | 18.1 | 158 | 10.1 |
| MAESTRO v2.0.0 | X | 1282 | 201.2 | 1282 | 201.2 |
| ASAP | 222 | 519 | 44.7 | 1067 | 94.2 |
| MuseSyn | 210 | 651 | 30.0 | 210 | 9.7 |
| ACPAS | 497 | 2189 | 179.8 | 1404 | 117.4 |

Additionally, [Table 3.2](#) presents a comparison of the statistics of different datasets used in Automatic Music Transcription (AMT) tasks. The MAPS dataset ([Emiya et al., 2010](#)), MAESTRO dataset ([Hawthorne et al., 2019](#)), and the ASAP dataset ([Foscarin et al., 2020](#)) are commonly utilised in this context. The MAPS and MAESTRO datasets do not include music scores and are often employed for multi-pitch estimation and note tracking tasks. On the other hand, the ASAP dataset provides beat-aligned scores along with the performances, but it lacks a sufficient number of audio performances required for audio-to-score transcription. To address this limitation, we have integrated data from various sources, including the ASAP dataset, into the ACPAS dataset. As a result, the ACPAS dataset has the advantage of having both music scores and an ample quantity of audio recordings, making it suitable for audio-to-score transcription tasks. In contrast, the MuseSyn dataset, displayed as the smallest dataset in the table, focuses on popular music rather than classical pieces. It serves as a resource for exploring transcription performance in music with limited note/melody complexity.

3.3.2 Other statistics

Here, we present a statistical analysis of the score annotations in the ACPAS and MuseSyn datasets. For the ACPAS dataset, we separately report statistics for data sourced from the ASAP dataset and those sourced from the MAPS dataset/CPM database. This distinction is necessary because they involve different performance features, and the score annotations are provided

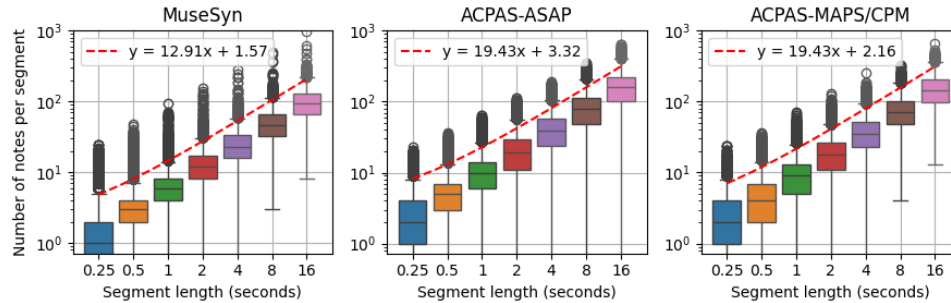


Figure 3.1: Notes per segment in different datasets. Red dashed line: Estimated 95% cutoff points of the number of notes per segment over different segment lengths, computed using linear regression.

in different formats. In the following, we refer to the three datasets/subsets as MuseSyn, ACPAS-ASAP and ACPAS-MAPS/CPM, respectively.

We calculate the statistics across the entire dataset, including the training, validation and test splits. The box plots for the statistics follow the conventional setting, with the middle line indicating the median, the upper and the lower boundaries indicating the upper and lower quartile, and dots indicating outliers. Below are the statistics we analysed to inform the configuration choices in our experiments.⁷

Number of notes across different recording lengths

For implementation purposes, we check the distribution of the number of notes per audio segment in different segment lengths. Very small segment lengths are also included to have a better understanding of the data distribution. Figure 3.1 shows the distributions for different datasets.

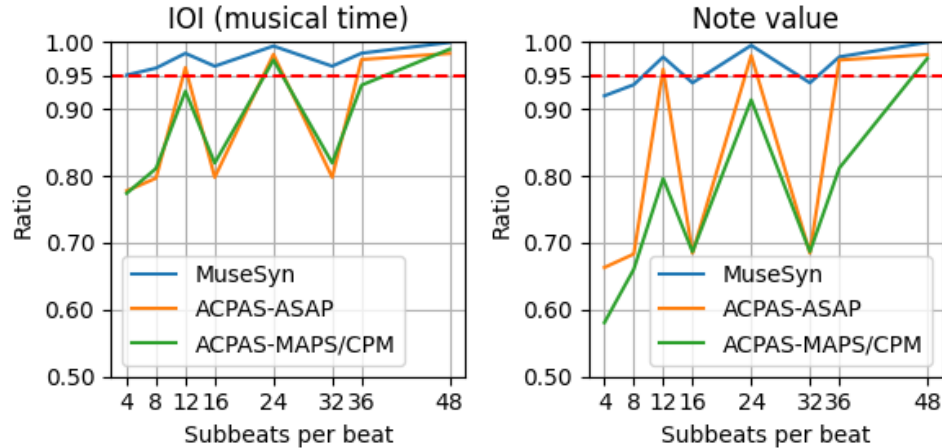


Figure 3.2: Representable scores in terms of inter-onset intervals in musical time and note values by different resolutions (in subbeats per beat).

Subdivision

We consider the subdivision process as dividing beats into multiple subbeats with a certain resolution. Here, we check the representable scores in terms of (musical) inter-onset intervals and note values with different resolutions (in subbeats per beat) to assist with choosing an appropriate resolution for our experiments.

For pieces in the ACPAS-MAPS/CPM, the MIDI ticks are annotated quite noisily (frequently ± 1 , ± 2 , ± 3 , ± 4 ticks apart from where they are supposed to be).⁸ Thus, we round the ticks into a lower resolution of 48 ticks per quarter note by down-sampling from the default resolution of 480 ticks per quarter note. This resolution should be sufficient for our experiments at the

⁷For the score-related statistics, such as note values, hand parts, we exclude those non-aligned notes.

⁸We checked the tick annotations both using `pretty_midi` and `mido`, with both packages giving the same result. This indicates that the error is not introduced by `tick_to_time()` and `time_to_tick()` functions provided in the `pretty_midi` package, which involves a back-and-forth computation to retrieve ticks. So, we use `pretty_midi` in our implementation for easiness.

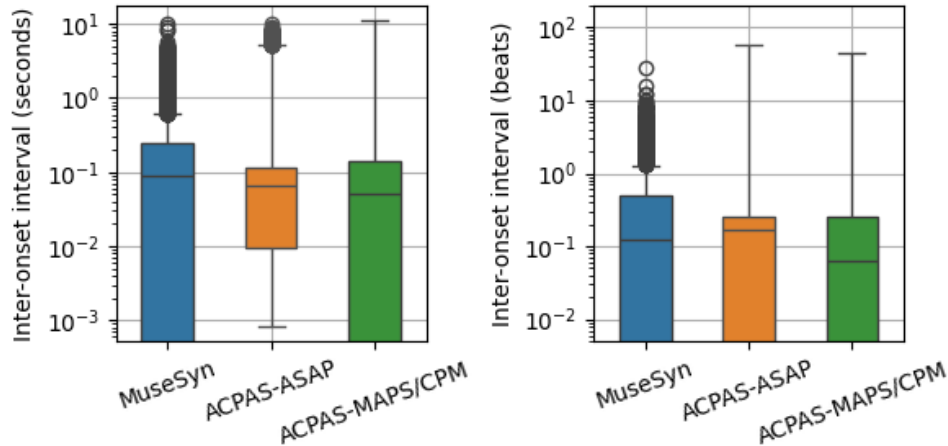


Figure 3.3: Inter-onset intervals in different datasets.

current stage. Note that when down-sampling, the resolution of 48 is in ticks per quarter note, which has a different unit from our definition of subdivision (i.e., dividing beats into subbeats per beat).

Figure 3.2 shows how different resolution for the subdivision process influences the ratio of representable scores in terms of inter-onset interval in musical time, and note value. From what is suggested in the figure, a resolution of 24 subbeats per beat tends to be a value that can represent most cases. Specifically, this resolution can represent over 95% of cases across the datasets, making it a good choice for our experiments. It is also well-suited for most classical piano music, as it covers common note values including 32nd notes, triplets, and sextuplets when quarter notes are used as the beat unit. While alternative resolutions were considered, higher resolutions would increase model complexity without significant gains in coverage, and lower resolutions may fail to represent a non-negligible portion of the data.

Inter-onset interval

To have a better estimate of how we encode note onsets in both physical time and musical time, we check the distribution of inter-onset intervals. Figure 3.3

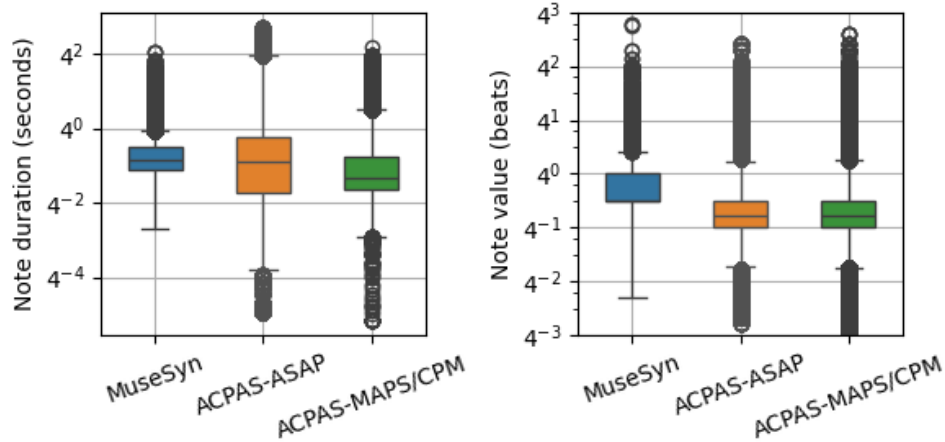


Figure 3.4: Note duration (in seconds) and note value (in beats) in different datasets.

shows their distributions in different datasets. Roughly, most inter-onset intervals are below 10 seconds in physical time, and 50 beats. 75% of the inter-onset intervals are well below these. This can be something worth taking into account when defining the encoding of our score representation.

Note duration, note value

Statistics on note duration (in seconds) and note value (in beats) can be found in [Figure 3.4](#). We can see that most note durations are below 4 seconds. As for note values, most of them fall below 4 beats.

frequent IOIs and note values

We analyse frequent inter-onset intervals and note values using a chosen resolution of 24 subbeats per beat. [Figure 3.5](#) shows how they distribute across different datasets. In our experiments, we might consider combining and encoding the rare ones in one token, which can simplify the vocabulary. Relevant missing information can be reconstructed given the beat predictions.

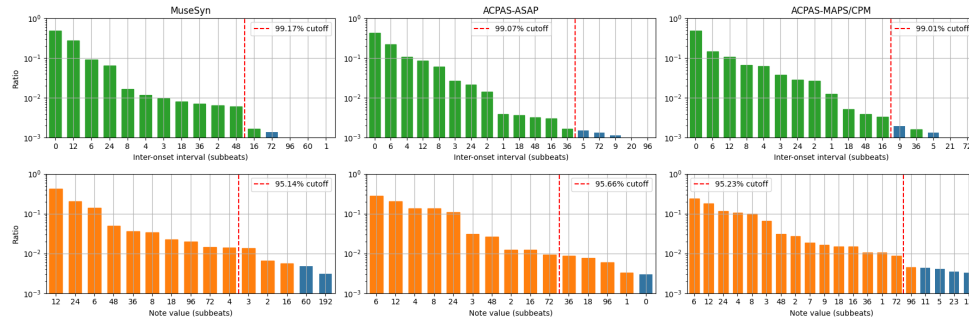


Figure 3.5: Proportion of frequent inter-onset intervals and note values in subbeats (with a resolution of 24 subbeats per beat) across different datasets. Green blocks: IOIs that fall within the cutoff ratio among all IOIs. Orange blocks: Note values that fall within the cutoff ratio among all note values. We use a lower cutoff ratio for note value than for inter-onset intervals, since we need lower resolution for note values.

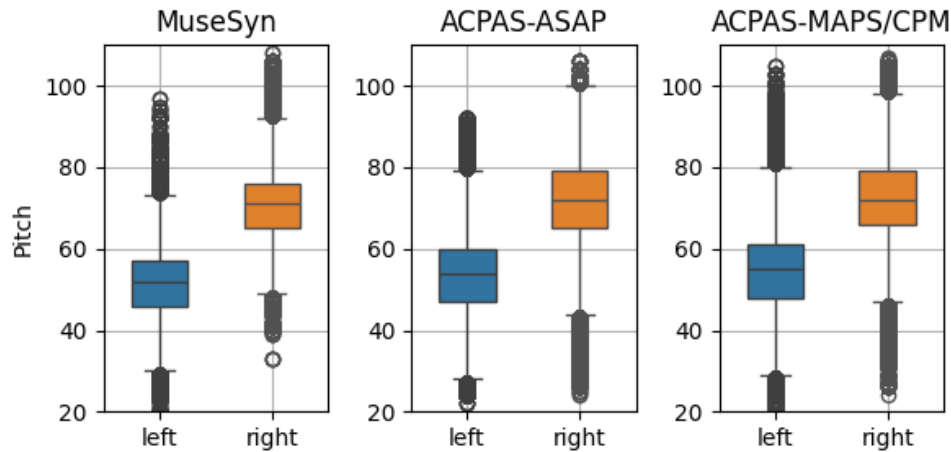


Figure 3.6: Pitch distributions per hand part in different datasets.

Hand part

As an additional statistical check, we plot the pitch distributions for each hand part in [Figure 3.6](#). This plot already shows that separating the hand

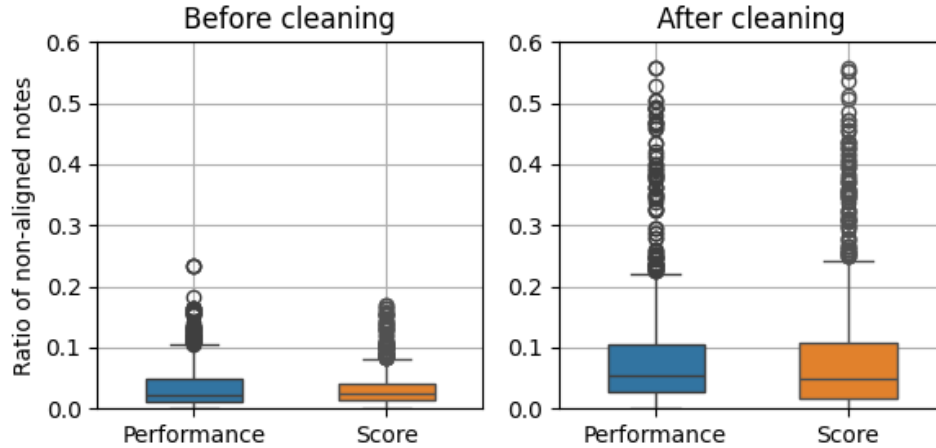


Figure 3.7: Ratio of non-aligned notes per piece in ACPAS-ASAP.

part using a simple cutoff pitch value can achieve over 75% accuracy.

Non-aligned notes

Thanks to the (n)ASAP dataset (Peter et al., 2023), which extends the ASAP dataset with note-level alignment (released in the middle of this PhD project), we are able to extend the ASAP music pieces in the ACPAS dataset with note-level alignments. Note alignments are provided in the (n)ASAP dataset with matches between performance notes and notes in the music score. Due to alignment errors and expressive performance, there are occasionally non-aligned notes, i.e., extra notes in the performance which do not have a matched score note, or missing notes in the performance where a score note is committed in the performance. We consider these as the “formally” non-aligned notes. However, after a closer check of the alignment annotations provided by the official (n)ASAP dataset, we found some extra annotation errors. There are occasional instances where notes have reversed physical and musical timing and where performance notes are missed out in the alignment annotation file. We consider these wrongly-annotated notes as “additional” non-aligned notes which we filter out by our examination. To account for these errors, we

also classify these “additional” non-aligned notes as non-aligned, even though they are not labelled as such in the official (n)ASAP dataset.

Figure 3.7 shows the proportion of the non-aligned notes before and after our examination process. We can see that the ratio of non-aligned notes for most pieces before our examination is roughly below 10%. After the examination by Peter et al. (2023), the proportion of non-aligned notes goes higher to roughly below 25%.

3.4 Discussion and conclusion

In this chapter, we have presented two datasets specifically curated for audio-to-score transcription research: the MuseSyn dataset and the ACPAS dataset. Each dataset emphasises a distinct music style and encompasses varying levels of expressive performance richness. Both datasets include music performance recordings in audio files, aligned with corresponding music scores. Due to the demand for professional music knowledge to create and record A2S transcription datasets, we collected the data from existing resources and synthesised the music recordings whenever they were unavailable.

There are certain limitations that need to be specifically clarified. For instance, the MuseSyn dataset suffers from a lack of expressive performances and has a limited size. Similarly, the ACPAS dataset faces a limited number of real audio performances and a lack of note-level alignment. Creating accurate note-level alignment between the music performances and scores represents another significant challenge when developing audio-to-score transcription datasets. Human performances often involve missing notes and extra notes, requiring careful consideration to handle such variations effectively. Moreover, dealing with diverse playing techniques poses an additional challenge. For example, situations may arise where multiple notes are present in a music performance, but the corresponding score only contains one or two notes (e.g., in vibrato or glissando). Despite these limitations, the datasets are sufficient for developing a first framework for audio-to-score transcription and offer a basis for further exploration in this field.

In this PhD project, we leverage the two datasets for exploring different approaches to audio-to-score transcription. The MuseSyn dataset is utilised

as a resource for investigating holistic methods in this context. On the other hand, the ACPAS dataset is employed to explore both pipeline-based methods and holistic methods. The decision not to use the MuseSyn dataset in pipeline-based methods stems from several reasons. First, our focus in the pipeline-based method is on the performance MIDI-to-score conversion step, which requires data with real performance features. The MuseSyn dataset has all its MIDIs converted directly from musical scores and thus presents no performance features: these MIDIs can be converted back to musical scores simply by estimating a global tempo, making the pipeline-based exploration on this dataset trivial. Second, the MuseSyn dataset has a relatively small size compared to other datasets primarily designed for note-tracking tasks. Third, the MuseSyn dataset lacks expressive performances, making it unsuitable for rhythm quantization tasks. In [Chapter 4](#) and [Chapter 5](#), we provide detailed descriptions of how we incorporate these two datasets in the training and evaluation of audio-to-score transcription systems.

Chapter 4

Pipeline-based Methods for Audio-to-Score Piano Transcription

4.1 Introduction

In this chapter, we introduce our work on pipeline-based methods for audio-to-score piano transcription. We follow the traditional AMT pipeline as presented in [Figure 2.4](#) in [Chapter 2](#), which divides the audio-to-score transcription into two major steps: *audio-to-performance MIDI transcription (A2PM)* and *performance MIDI-to-score conversion (PM2S)*.

Thanks to the rapid development of deep learning, recent years have seen remarkable advances in audio-to-performance MIDI transcription, especially for piano music ([Hawthorne et al., 2018](#); [Kong et al., 2021](#); [Toyama et al., 2023](#)). State-of-the-art piano transcription systems can achieve F-scores of over 95% on the notewise onset-only F-measure when evaluated on the MAESTRO dataset, one of the benchmark datasets for piano transcription ([Hawthorne et al., 2019](#); [Kong et al., 2021](#); [Toyama et al., 2023](#)). Here, we consider the onset-only F-measure as a reference evaluation metric since it is the benchmark metric that correlates best with human judgement according to what we discovered in ([Ycart et al., 2020b](#)). Given existing advances in audio-to-performance MIDI transcription (note-level transcription) in piano music, one of the possible methods for *audio-to-score transcription (A2S)* is

to follow the *traditional AMT pipeline* (as described in [Chapter 2](#)) by using an existing audio-to-performance MIDI transcription (A2PM) system. Thus, in this chapter, we focus on the latter (less explored) part of the AMT pipeline, i. e., performance MIDI-to-score conversion.

Although research on various sub-tasks of performance MIDI-to-score conversion (PM2S) dates back to earlier decades ([Desain & Honing, 1989](#); [Temperley & Sleator, 1999](#); [Raphael, 2001](#); [Takeda et al., 2002](#)), the first paper that fully brings the PM2S problem into literature was in 2016 ([Cogliati et al., 2016](#)), in which the authors worked on converting a performance MIDI recording into a LilyPond ([Nienhuys & Nieuwenhuizen, 2003](#)) score using hidden Markov models. The method is further improved by using a convolutional sparse coding-based multi-pitch estimation and a Melisma Analyser-based rhythm quantisation ([Cogliati, 2018](#)). Among various subtasks in performance MIDI-to-score conversion, rhythm quantisation is one of the most essential tasks that has been widely discussed in the literature. Over the years, people have applied various algorithms to the task, using hidden Markov models ([Nakamura et al., 2017b, 2018](#); [McLeod & Steedman, 2018b](#)) and Markov random fields ([Nakamura et al., 2017a](#); [Shibata et al., 2021](#)). In addition to rhythm quantisation, there is research on other subtasks of PM2S, such as note value and voice detection using recurrent neural networks ([Hiramatsu et al., 2021](#)) and score formatting from quantised MIDI using the Transformer architecture ([Suzuki, 2021](#)). For a more detailed literature study on the PM2S task and relevant subtasks, we refer to the background chapter (see [Chapter 2](#)).

In this chapter, we propose a new deep learning-based PM2S method, which we use together with an existing performance MIDI transcription system ([Kong et al., 2021](#)) to form an audio-to-score transcription system. We develop a convolutional-recurrent neural network that directly converts a performance MIDI into a score MIDI. Among the various sub-tasks in performance MIDI-to-score conversion, we pay special attention to the rhythm quantisation step. We propose to do rhythm quantisation via neural beat tracking, i. e., tracking beats from a MIDI note sequence. To our knowledge, this is the first method that solves rhythm quantisation using deep learning. We investigate different input features and compare our proposed beat-tracking method

with a baseline beat-tracking model using a temporal convolutional network. We further expand the convolutional-recurrent neural network to predict the time signatures, key signatures and hand parts in piano music. To better capture the model’s capability in transcribing expressive piano performances, we train and evaluate our proposed method on the ACPAS dataset (described in Chapter 3). We compare different input encodings for note sequences, and run ablation studies on different input features and data augmentation methods. We compare our method with two commercial software products, MuseScore (MuseScore, 2022) and Finale (Finale, 2022), and an HMM-based statistical method (Shibata et al., 2021). We show that our proposed method achieved better performance compared to the two commercial software products and the two statistical methods based on the MV2H metric (McLeod & Steedman, 2018a) (see description of the MV2H metric in subsection 2.4.2 of Chapter 2).

Following that, we introduce our proposed audio-to-score transcription pipeline. We combine our proposed PM2S method with an existing audio-to-performance MIDI transcription system (HighRes) (Kong et al., 2021), ending up in a pipeline-based system for audio-to-score transcription. To account for the domain shift from recorded performance MIDI to transcribed MIDIs, we compare the use of different training data. Lastly, we present the audio-to-score transcription results on the ACPAS dataset using our proposed pipeline using the MV2H metric, and compare it with three other pipeline-based methods (i. e., HighRes+MuseScore (Kong et al., 2021; MuseScore, 2022), POVNet+Melisma Analyzer (Cogliati et al., 2016; Shibata et al., 2021), and POVNet+HMMs (Shibata et al., 2021)). We show that our proposed method achieves better overall results, although underperforming POVNet+HMMs on two MV2H sub-metrics.

The rest of this chapter is organised as follows. In Section 4.2, we explain our proposed performance MIDI-to-score conversion method. After that, we introduce how we use the proposed PM2S method in an audio-to-score transcription pipeline in Section 4.3. Conclusions can be found in Section 4.4. We released our code for PM2S at <https://github.com/cheriell/PM2S> and pre-trained models at <https://zenodo.org/records/10520196>. Related publi-

cation for this chapter is (Liu et al., 2022).

4.2 Performance MIDI-to-score conversion

4.2.1 Introduction

In this section, we introduce our proposed method for performance MIDI-to-score conversion (PM2S), with a special focus on rhythm quantisation from performance MIDI. Considering rhythm quantisation as a fine-grained tracking of beats and beat subdivisions, we propose to do rhythm quantisation by combining neural beat tracking from performance MIDIs. Over the years, research on neural beat tracking has been mainly focused on tracking beats and downbeats from music audio recordings (audio domain) (Müller, 2021; Böck & Davies, 2020; Zhao et al., 2022). Instead, we propose to track beats and downbeats from performance MIDIs (symbolic domain). On the one hand, this corresponds better to our rhythm quantisation task on the input data; on the other hand, it allows tracking beats from a more sparse input encoding than an audio spectrogram, which is usually the input data for an audio-domain beat tracking system. After tracking the beats, we extend the neural network to do rhythm quantisation by predicting the quantised note onsets and note values within beats. We use similar network structures to predict hand parts, time signatures, and key signatures from a performance MIDI. After that, we use the predicted information to convert the performance MIDI into a score MIDI.

In the following sections, we first introduce our beat tracking method in subsection 4.2.2, followed by how we extend the beat tracking model for rhythm quantisation in subsection 4.2.3. We describe the prediction of hand parts, time signatures and key signatures in subsection 4.2.4. After that, subsection 4.2.5 describes our beat tracking and rhythm quantisation experiments.

June
(Barcarolle)

Tchaikovsky
The Seasons

Andante cantabile

x x x x x x x x x x x x x x x x x x x

x: onset beats

x: non-onset beats

Figure 4.1: Illustration of onset beats and non-onset beats. Music example: the first five bars of “The Seasons - June (Barcarolle), by P. I. Tchaikovsky”.

4.2.2 Beat tracking from performance MIDI

Onset beats and non-onset beats

To match the input data in performance MIDI format, we consider dividing beats into two categories. *Onset beats* which refers to the beats that are at note onsets, and *non-onset beats* which refers to the remaining beats (that are not at note onsets). Figure 4.1 provides an example of how we define them in a musical piece. It can be observed from the figure that each onset beat can be aligned with at least one note onset. This allows us to model onset beats using a sequential model as a binary classification task. For non-onset beats, we propose a dynamic programming algorithm to infer non-onset beats from onset beats.

Onset beat prediction

We consider the prediction of onset beats as a binary classification task. Given a performance MIDI in the form of a note sequence:

$$\mathbf{X} := \{(p_n, o_n, d_n, v_n)\}_{n=1}^{N^{\text{notes}}}, \quad (4.1)$$

where p_n , o_n , d_n and v_n are the pitch, onset, duration, and velocity of the n -th note in the note sequence, and N^{notes} is the number of notes, the notes in the note sequence are ordered by their onset times in increasing order, we define a model for onset beat prediction

$$f^{\text{onset.beat}} : \mathbf{X} \rightarrow \mathbf{y}^{\text{onset.beat}} \quad (4.2)$$

where $\mathbf{y}^{\text{onset.beat}}$ are the probabilities of each note onset being a beat:

$$\mathbf{y}^{\text{onset.beat}} := \{y_n^{\text{onset.beat}}\}_{n=1}^{N^{\text{notes}}}, y_n^{\text{onset.beat}} \in \{0, 1\}. \quad (4.3)$$

Figure 4.2 shows the model architecture of $f^{\text{onset.beat}}$. We use a convolutional recurrent model with three convolutional layers (`ConvBlock`) and two bi-directional recurrent layers (`GRUBlock`, using a gated recurrent unit). For the convolutional layers, we use one-dimensional (along the note index n in Equation 4.1) convolutional layers with a kernel size of 9, which enables the convolutional layers to learn short-term dependencies among musical notes. Bi-directional recurrent layers are used so that the model can learn longer-term dependencies from both previous and future notes. The predicted onset beat probabilities are then converted to binary labels by dynamic thresholding, where the dynamic threshold is calculated based on the maximum predicted probability over a fixed segment length in seconds.

Additionally, we propose to jointly track beats, downbeats, and local tempo (more precisely, we use inter-beat intervals in our implementation) in a multitask learning model. Similar to how we define onset beats, we track *onset downbeats* (downbeats which are at note onsets) by predicting whether each note onset is a downbeat in a binary classification task. For local tempo, we calculate the inter-beat intervals between the neighbouring beats of each note onset. We quantise the inter-beat intervals linearly into 200 bins and consider the prediction of them as a multi-class classification task. Figure 4.3 shows the model architecture of the multitask model we use for joint beat, downbeat, and tempo estimation from performance MIDIs.

Non-onset beats prediction

We now explain how we predict the non-onset beats from the onset beats. We propose a dynamic programming algorithm (Cormen et al., 2022) based on

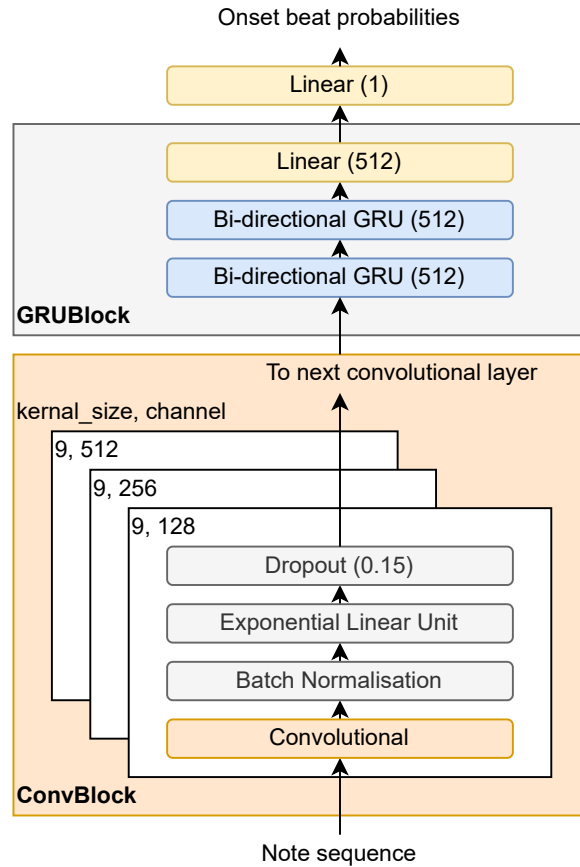


Figure 4.2: Model architecture for onset beat prediction.

two assumptions:

- **Assumption 1:** Non-onset beats are at subdivisions of their neighbouring onset beats. (We use this to find candidate non-onset beats.)
- **Assumption 2:** The tempo change after combining onset beats and non-onset beats should be minimised. (We use this to optimise our choice of non-onset beats).

These two assumptions, however, are only a rough estimation of the beat patterns in music and do not always hold in real music performances, especially when a higher level of tempo change happens in rich expressive performances.

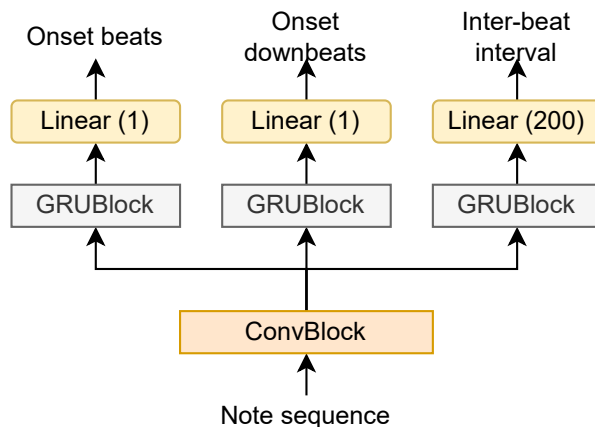


Figure 4.3: Model architecture for joint (onset) beat, (onset) downbeat and tempo estimation from performance MIDI. The ConvBlock and GRUBlock have the same architecture as in Figure 4.2.

We thus consider this method as a preliminary exploration of how non-onset beats can be predicted.

Assuming there are $N^{\text{onset.beats}}$ onset beats predicted, we define the onset beats sequence as:

$$\mathbf{b}^{\text{onset.beats}} := \{b_n^{\text{onset.beats}}\}_{n=1}^{N^{\text{onset.beats}}}, \quad (4.4)$$

where $b_n^{\text{onset.beats}}$ is the n -th onset beat (in seconds). Considering possible non-onset beats in the subdivisions of two neighbouring onset beats and assuming there are K non-onset beats between the n -th and $(n+1)$ -th onset beats, we compute non-onset beat candidates at the following positions (time in seconds):

$$\left\{ b_n^{\text{onset.beats}} + \frac{k}{K+1} (b_{n+1}^{\text{onset.beats}} - b_n^{\text{onset.beats}}) \right\}_{k=1}^K, \quad (4.5)$$

In our experiments, we use $K \in \{0, 1, 2, 3\}$. This is to say, we insert non-onset beats by linear interpolation:

- if $K = 0$, we do not insert non-onset beats between the two onset beats;
- if $K = 1$, insert one non-onset beat at interpolated position 1/2;

- if $K = 2$, insert two non-onset beats at interpolated position $1/3$ and $2/3$;
- if $K = 3$, insert three non-onset beats at interpolated positions $1/4$, $2/4$, $3/4$.

For each beat interval $(b_n^{\text{onset_beats}}, b_{n+1}^{\text{onset_beats}})$, we search for the best K value according to assumption 2 using dynamic programming. To minimise the tempo change after combining onset beats and non-onset beats, we define an objective function that is minimised during our search:

$$\mathbf{b} := \{b_n\}_{n=1}^{N^{\text{beats}}} := \mathbf{b}^{\text{onset_beats}} \cup \mathbf{b}^{\text{non-onset_beats}}, \quad (4.6)$$

$$\mathcal{O}^{\text{tempo_change}} = \sum_{n=1}^{N-2} \left| \log \left(\frac{b_{n+2} - b_{n+1}}{b_{n+1} - b_n} \right) \right|, \quad (4.7)$$

where \mathbf{b} is the beat sequence after combining onset beats and non-onset beats and N^{beats} is the number of beats in the final beat sequence \mathbf{b} . b_n is the n -th beat and $\mathcal{O}^{\text{tempo_change}}$ is the objective function that describes the level of tempo change over the whole piece. However, this objective function does not take into account adding too many non-onset beats to the beat sequences. We thus add a penalty associated with the number of non-onset beats to the objective function, resulting in:

$$\mathcal{O} = \mathcal{O}^{\text{tempo_change}} + \lambda \times N^{\text{non-onset_beats}}. \quad (4.8)$$

The λ is the penalty coefficient applied to avoid adding too many non-onset beats, which is set to 1 in our experiments, and $N^{\text{non-onset_beats}}$ is the number of non-onset beats. In this way, we obtain an objective function \mathcal{O} to be minimised in the non-onset beats prediction process that encourages both a low level of tempo change and adds fewer non-onset beats. The pseudo-code of the dynamic programming algorithm we use to find the optimal non-onset beats is explained in Algorithm 1.

4.2.3 From beat tracking to rhythm quantisation

Rhythm quantisation becomes easier once we obtain the beat positions. We use a convolutional recurrent neural network to predict the musical positions

Algorithm 1 Non-onset beats prediction

Input: List of onset beats $\mathbf{b}^{\text{onset_beats}}$ **Output:** List of all beats

```

1:  $n \leftarrow 1$ 
2: for  $K = 0, 1, 2, 3$  do
3:   Initialise objective function  $\mathcal{O}_K \leftarrow 0$ 
4:   Initialise beat sequence  $\mathbf{b}_K \leftarrow \{\mathbf{b}_1^{\text{onset\_beats}}\}$ 
5: end for
6: for  $n = 1, 2, \dots, N^{\text{onset\_beats}} - 2$  do
7:   for  $K_{\text{cur}} = 0, 1, 2, 3$  do
8:     Get candidate non-onset beats for current step by Equation 4.5
9:     if Tempo is beyond tempo range limits then
10:      Go to next  $K_{\text{cur}}$ 
11:     end if
12:     for  $K_{\text{prev}} = 0, 1, 2, 3$  do
13:      Update objective function by Equation 4.8
14:     end for
15:     Select the minimum objective among all  $K_{\text{prev}}$  values
16:     Add non-onset beats for the current step to the beat sequence
      mapped to the selected  $K_{\text{prev}}$ 
17:     end for
18:     for  $K_{\text{cur}} = 0, 1, 2, 3$  do
19:      Update  $\mathcal{O}_K$ ,  $\mathbf{b}_K$  mapped with  $K_{\text{cur}}$ 
20:     end for
21: end for
22: Return the beat sequence in  $\mathbf{b}_K$  with the minimum objective function  $\mathcal{O}_K$ 

```

of each note. We take both the performance MIDI and the onset beat predictions from our beat tracking model as input data for the quantisation model, as in [Figure 4.4](#). The rhythm quantisation model learns to jointly predict quantised note onsets and note values. In our experiments, we round the musical positions into 24 bins per quarter note, which allows for common note lengths such as 16th note, 32nd note and triplets. The same quantisation res-

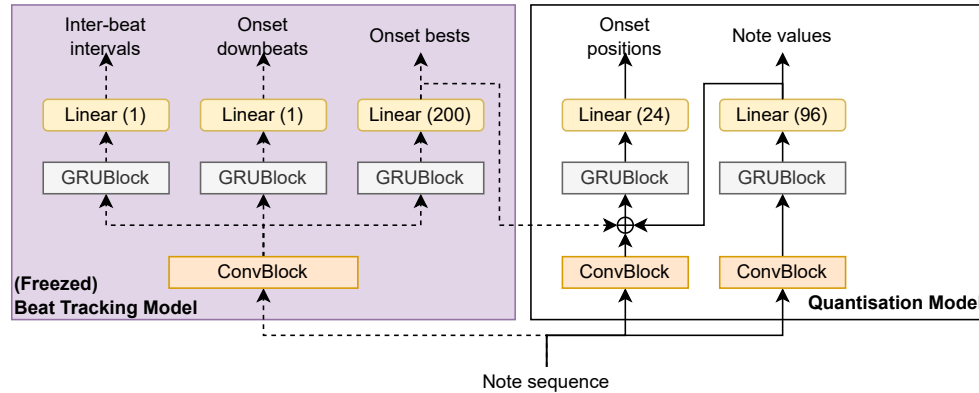


Figure 4.4: Training the rhythm quantisation model with the help of beat tracking. The ConvBlock and GRUBlock have the same model architecture and size as those in the beat tracking model.

olution is used for the prediction of note values. For each note, both outputs (quantised note onset and note value) are one-hot encoded and modelled as multi-class classification tasks.

During post-processing, we use linear interpolation as a reference method to identify predictions that are highly probable to be wrong. We consider linear interpolation as one of the simplest ways to do rhythm quantisation, in which we linearly interpolate the inter-beat intervals to get the quantised note onsets and offsets. This can be a reference because, compared to the level of tempo change in a whole piece of music performance, there is usually less tempo change within one beat. In practice, we take the output from the quantisation model if it is not too far (less than a 16th note duration) from what we obtain by linear interpolation. Otherwise, we take the output from the linear interpolation.

4.2.4 Prediction of hand parts, time signatures and key signatures

We use similar convolutional recurrent neural networks to model hand parts, time signatures and key signatures. We define hand parts \mathbf{Y}_h , time signature

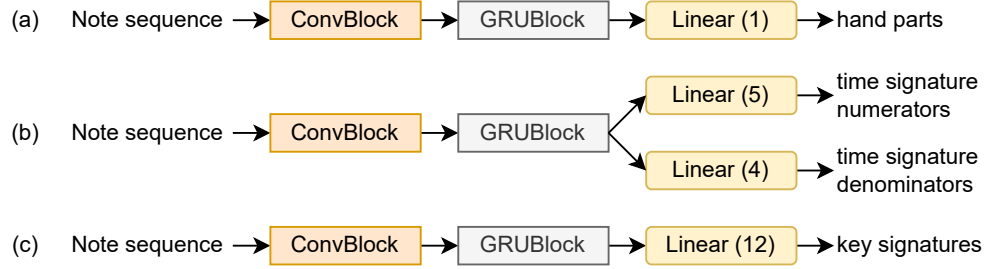


Figure 4.5: Model architectures for the prediction of hand parts (a), time signature changes (b) and key signature changes (c). The ConvBlock and GRUBlock have the same model architecture as in the beat tracking model.

changes \mathbf{Y}_t and key signature changes \mathbf{Y}_k to be mapped to each note (or note onset time), in the following form:

$$\mathbf{Y}_h = \{h_n\}_{n=1}^{N^{\text{notes}}}, h_n \in \{0, 1\}, \quad (4.9)$$

$$\mathbf{Y}_t = \{(t_n^n, t_n^d)\}_{n=1}^{N^{\text{notes}}}, t_n^n \in \{0, 2, 3, 4, 6\}, t_n^d \in \{0, 2, 4, 8\}, \quad (4.10)$$

$$\mathbf{Y}_k = \{k_n\}_{n=1}^{N^{\text{notes}}}, k_n \in \{0, 1, \dots, 11\}, \quad (4.11)$$

where h_n , t_n^n , t_n^d , and k_n are the hand part, time signature nominator, time signature denominator, and the key signature at the n -th note, respectively. For hand parts, values 0 and 1 refer to the left and right hands. For the time signature, 0s for both t_n^n and t_n^d are used as a placeholder to indicate other values, similar to how unknown words are modelled in natural language processing tasks. For the key signature changes, values 0 to 11 correspond to key signatures C, Db, D, Eb, E, F, Gb, G, Ab, A, Bb, and B, respectively.

The model structure is shown in Figure 4.5. The model is trained using binary cross-entropy loss for hand part prediction, and negative log-likelihood loss for the prediction of time signature changes and key signature changes. After the rhythm quantisation and the prediction of hand parts, time signature changes and key signature changes, a musical score can then be generated by combining the predictions with the performance MIDI.

Table 4.1: Dataset Statistics. Performance from the MAPS dataset (Emiya et al., 2010) and the CPM database (CPMDatabase, 2021) for the same piece are not counted as different performances, since MAPS pieces are originally obtained from the CPM database.

| Split | Distinct Pieces | Performances | Duration (h) | Notes (10^3) |
|------------|-----------------|--------------|--------------|------------------|
| Training | 426 | 1324 | 108.3 | 3984.0 |
| Validation | 49 | 157 | 12.7 | 517.6 |
| Testing | 29 | 29 | 2.2 | 73.2 |
| Total | 504 | 1510 | 123.2 | 4574.7 |

4.2.5 Experiments

Data

To explore the ability of our proposed method for rhythm quantisation in expressive piano performance, we use the ACPAS dataset in our experiments (see Chapter 3). We use a real recording subset in MAPS (i. e., the ENSTDkCl subset) as the test set, as in (Nakamura et al., 2018; Hiramatsu et al., 2021). This test set is only a part of the ACPAS test set. We use only a part of the ACPAS test set since there is no note-level alignment between the performance MIDIs and the annotated musical scores for the piece collected from the ASAP dataset. Other music pieces not included in the test set are randomly split into training and validation based on their distinct music piece labels. In this way, we get a train/validation/test setup with no overlapping music pieces among splits. Table 4.1 shows the dataset statistics.

The dataset annotations are provided in different formats. The A-MAPS dataset and the CPM database provide fully annotated score MIDIs with tempo and metrical information. Thus, we extract the annotations we need from the score MIDIs directly. Performances from the ASAP dataset come with two sets of annotations: score MIDIs and annotations in .tsv files. However, the score MIDIs were written by non-professionals, and the dataset creators do not suggest using them as ground truth annotations. Thus, we follow the creators’ suggestion to use the provided annotations in the .tsv files, in

which beat, downbeat, time signature and key signature annotations are provided. Because the .tsv annotations do not cover precise fine-grained metrical and hand part annotations, we mask the ASAP data on musical onset time, note value, and hand part prediction during training. Moreover, when matching note onset times to beats or non-beats in our proposed model, we set a tolerance of ± 50 ms to account for short-time variances (e. g., micro-timing) introduced by human performance.

For data augmentation, we consider the following techniques:

- **Pitch shift:** Shift MIDI pitch values up or down for the whole music performance. The shifted pitch is defined as: $p_s = p + p_{\text{shift}}$ where p is the original pitch value, $p_{\text{shift}} \in \{0, \pm 1, \pm 2, \dots, \pm 12\}$.
- **Tempo change:** Change the tempo to a ratio of the original tempo. The new tempo $v_c = \alpha \times v$ where v is the original tempo and α is the tempo change ratio, which is randomly sampled in the range $[0.8, 1.2]$.
- **Note removal:** For polyphonic music, we usually expect little change to the metrical structure of a music piece when removing some concurrent notes. Thus, for each group of concurrent notes, we randomly remove a part of them from the MIDI performance.
- **Extra note:** Contrary to note removal, we randomly select 0-100% of notes from the MIDI performance and add new notes that are concurrent with the selected ones. We keep the velocity and duration the same to preserve the original music structure as much as possible. The new note pitches are ± 12 semitones apart from the original note pitches.

Evaluation metrics

We define a *note-level F-measure* to evaluate the tracking of onset beats, and a *beat-level F-measure* which follows the benchmark F-measure for beat and downbeat tracking (Davies et al., 2009) with a time tolerance of ± 70 ms.¹ In both cases, a true positive means a predicted beat is in the ground truth; a

¹We use ± 70 ms because this is the conventional tolerance for the beat tracking task (Müller, 2021), although for AMT evaluation, a tolerance of ± 50 ms is commonly used.

Table 4.2: Description of different input data encodings.

| Feature | Encoding | Description |
|----------|----------|--|
| Pitch | M | 128-dimensional one-hot vectors in MIDI pitch numbers |
| | C | 12-dimensional one-hot vectors in octave values |
| Onset | AR | The raw value in seconds |
| | AO | One-hot vectors quantised by 10ms resolution |
| | SR | Onset time shift in seconds compared to the previous note onset |
| | SO | One-hot encoded onset time shift quantised by 10ms resolution (with a maximum onset time shift of 4s, larger values are trimmed to 4s) |
| Duration | R | The raw values in seconds |
| | O | One-hot vectors quantised by 10ms resolution (similar to onset shift, large values are trimmed to 4s) |
| Velocity | – | Velocities are always normalised to 0-1 |

false positive means a predicted beat is not in the ground truth; and a false negative means a ground truth beat is missing in the prediction. Please refer to [subsection 2.4.2](#) in [Chapter 2](#) for more detailed definitions of these metrics. For both F-measures, we report their precision, recall, and F-score (P_{note} , R_{note} , and F_{note} for note-level F-measure and P_{beat} , R_{beat} , F_{beat} for beat-level F-measure). Similar definitions are used for downbeat.

Input data comparison

We start our experiments with the onset beat tracking model (without multi-tasking, see [Figure 4.2](#)) by exploring different encodings of the input data. We compare different ways of encoding a MIDI sequence as explained in [Table 4.2](#).

For all possible input encoding combinations, we train and evaluate the onset beat prediction. The model learning rate is 0.001. Since different input

Table 4.3: Note-level F-measure results for onset-beat tracking using different input data encoding combinations.

| Input Encodings | | | Note-Level F-measure | | |
|-----------------|-------|----------|----------------------|-------------------|-------------------|
| Pitch | Onset | Duration | P_{note} | R_{note} | F_{note} |
| M | AR | R | 86.7 | 77.7 | 79.9 |
| M | AR | O | 89.9 | 57.2 | 65.2 |
| M | AO | R | 82.1 | 78.3 | 79.3 |
| M | AO | O | 83.3 | 72.2 | 76.0 |
| M | SR | R | 89.2 | 89.4 | 87.8 |
| M | SR | O | 88.8 | 91.9 | 89.5 |
| M | SO | R | 91.2 | 94.3 | 91.3 |
| M | SO | O | 91.1 | 90.3 | 89.1 |
| C | AR | R | 86.7 | 68.7 | 73.6 |
| C | AR | O | 80.7 | 64.4 | 67.2 |
| C | AO | R | 82.8 | 76.2 | 78.4 |
| C | AO | O | 82.7 | 71.9 | 76.0 |
| C | SR | R | 90.4 | 88.0 | 87.3 |
| C | SR | O | 90.2 | 88.9 | 87.5 |
| C | SO | R | 89.9 | 91.3 | 89.1 |
| C | SO | O | 88.8 | 90.6 | 88.0 |

encodings can cause changes in the model convergence speed, we do not add learning rate decay in this comparison. The model is trained using a batch size of 32 over 4 GPUs with a dropout rate of 0.15. For each combination, we take the best model checkpoint on the validation set during training and evaluate it over the test set. Model performance on all different input data encoding combinations is reported in [Table 4.3](#).

From the results, we can see that for pitch encoding, using the MIDI pitches (M) tends to outperform using the chroma groups (C) most of the time. For onsets, onset shift (SR, SO) leads to better results than absolute onset (AR, AO) across all encoding combinations. Using one-hot encoding (AO, SO) for onset is better than using the raw values for most cases. Six out

Table 4.4: Note-level F-measure results on ablation study for input features.

| Input feature omitted | P_{note} | R_{note} | F_{note} |
|-----------------------|-------------------|-------------------|-------------------|
| Pitch | 90.4 | 93.7 | 90.6 |
| Onset | 84.6 | 72.8 | 76.4 |
| Duration | 89.5 | 93.1 | 90.1 |
| Velocity | 89.5 | 93.9 | 90.6 |
| Use all features | 91.2 | 94.3 | 91.3 |

of eight encoding combinations result in better model performance with onsets encoded in one-hot format. However, an opposite preference is discovered for duration encoding, where duration in raw values (R) leads to better results for more cases. Among all the sixteen input data encoding combinations tested, the one with MIDI pitch (M), one-hot onset shift (SO), and duration in raw value (R) shows the best model performance. We use this input data encoding combination in our subsequent experiments.

Using the best input encoding combination observed in the previous comparison, we do an ablation study on the input features. Results are reported in Table 4.4. From the table, we see that all four input features are helpful in beat tracking, among which the onset feature is the most beneficial one. This is consistent with the consensus that onsets are the feature to carry most metrical information in music performances. People can usually do beat tracking from drum beats without knowing other information, such as pitch or duration. Pitch and velocity are less important but still have some positive impact on the overall performance. That may be because pitch carries some harmony information that helps beat prediction. Velocity provides useful information as well since beats are more likely to be aligned with heavy notes. Finally, duration is of certain importance, suggesting it carries more metrical information than pitch and velocity.

Ablation study on data augmentation methods

To better understand the behaviour of the data augmentation methods, we do an ablation study on the data augmentation methods mentioned earlier in this

Table 4.5: Note-level F-measure results on the ablation study for data augmentation methods.

| Augmentation method omitted | P_{note} | R_{note} | F_{note} |
|-----------------------------|-------------------|-------------------|-------------------|
| Pitch shift | 92.0 | 92.0 | 90.6 |
| Tempo change | 91.7 | 89.5 | 89.7 |
| Note removal | 91.5 | 90.9 | 90.4 |
| Extra note | 91.2 | 94.3 | 91.3 |
| Use all methods | 92.9 | 93.7 | 92.2 |
| No data augmentation | 89.7 | 95.2 | 90.9 |

section. We omit one data augmentation method at a time and compare the model performance on tracking onset beats. Results of the study are shown in [Table 4.5](#).

Results on different data augmentation methods suggest that all four data augmentation methods we used improve model performance, among which tempo change is the most beneficial one. Not performing data augmentation does not result in the lowest note-level F-score. However, its low precision rate indicates a limitation on predicting more false positives, which is discouraged since we will be adding out-of-note beats based on the predicted in-note beats. It is possible that we can add the false negatives back when predicting out-of-note beats, but we cannot remove the false positives.

Comparison with a baseline beat tracking model

Using the best configurations based on the previous experiment results, we include the non-onset beats prediction and evaluate our proposed beat tracking model on beat-level F-measure. We compare our method with a baseline model that is similar to an existing audio beat tracking model introduced in ([Böck & Davies, 2020](#)), using the beat and downbeat tracking parts. Since we work with performance MIDIs, we train the baseline model using a piano-roll input (calculated from the note sequence), replacing the original audio spectrogram input. In [Figure 4.6](#), we show the baseline model architecture we use in our experiment. The predicted beat and downbeat probabilities are then

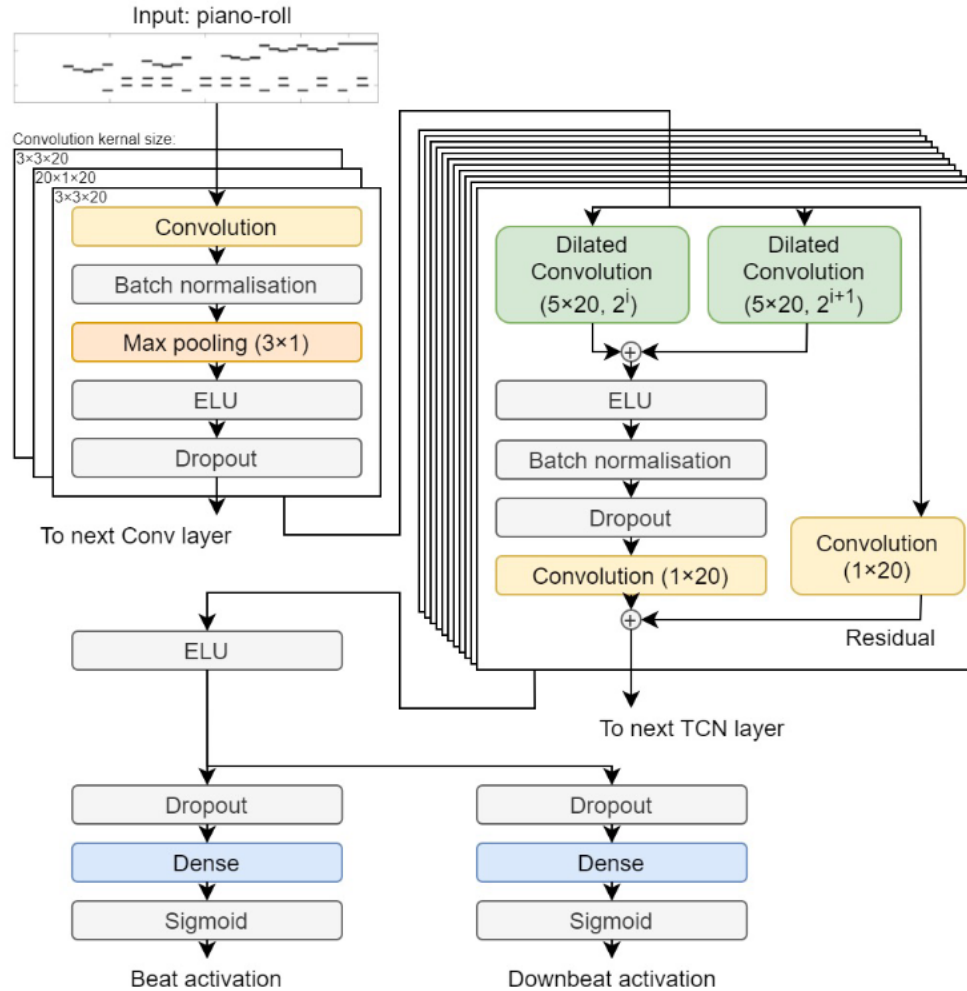


Figure 4.6: Baseline beat tracking model architecture using a temporal convolutional network.

passed to a dynamic Bayesian network (Böck et al., 2014; Krebs et al., 2015) to get beats and downbeats in seconds.²

Table 4.6 shows the comparative results between the baseline and our

²In practice, we use the madmom Python library, which provides a function for the dynamic Bayesian post-processing method.

Table 4.6: Beat-level F-measure results on the baseline and proposed models.

| Models (output data) | F_{beat} | F_{downbeat} |
|------------------------------|-------------------|-----------------------|
| Baseline (beat, downbeat) | 66.9 | 57.6 |
| Ours (beat, downbeat) | 85.7 | 63.3 |
| Ours (beat, downbeat, tempo) | 86.2 | 69.8 |

Table 4.7: MV2H evaluation on performance MIDI-to-score conversion.

| Methods | F_{pitch} | F_{voice} | F_{meter} | F_{value} | F_{harmony} | F_{MV2H} |
|-----------|--------------------|--------------------|--------------------|--------------------|----------------------|-------------------|
| Finale | 82.2 | 54.6 | 9.9 | 92.2 | 86.2 | 65.0 |
| MuseScore | 10.0 | 65.0 | 15.3 | 95.0 | 84.5 | 54.0 |
| HMMs | 91.2 | 71.1 | 51.7 | 91.3 | 77.0 | 76.5 |
| Ours | 99.8 | 87.0 | 61.7 | 99.9 | 91.1 | 87.9 |

proposed model. Results suggest our proposed model outperforms the baseline when jointly trained with beats and downbeats. This may be because the baseline model is a general-purpose system designed to operate on richer content than piano music alone, and our proposed model can better handle tempo changes. By adding tempo to the output data, the performance of our proposed model can be further improved, which suggests the benefit of using multitask learning.

Score-level evaluation

Using the best configurations found in the comparative experiments, we train and evaluate our proposed PM2S model as a whole. We report the model performance on the MV2H metric. During post-processing, we convert the performance MIDIs into quantised MIDIs by merging the rhythm quantisation predictions into the performance MIDI files. To avoid potential alignment problems in the MV2H evaluation algorithm, including extremely long waiting times and possible alignment errors, we choose to use the non-aligned version of the evaluation tool. To do this, we force the quantised MIDIs to have the same physical time as the ground truth score MIDIs. So, we create

the quantised MIDIs by first calculating tempo change annotations and then adding the notes (obtained from the performance MIDIs) and tempo change annotations to a new MIDI file. Tempo changes are calculated at the note onset times of each group of concurrent notes, where we consider notes as concurrent if their onset times are within 50ms apart. We use a default setting of a fourth note per beat.

We compare our proposed method with two commercial software products, Finale v27 (Finale, 2022) and MuseScore v3 (MuseScore, 2022), by importing the performance MIDIs, and with an HMM-based statistical method whose result on the same test set is reported in (Shibata et al., 2021). Results are in Table 4.7. Results suggest that our proposed model outperforms MuseScore, Finale, and the HMM-based method across all MV2H sub-metrics. Significantly better performance can be observed in the voice separation and metrical alignment sub-metrics (F_{voice} and F_{meter}). Better hand part separation and key signature estimation (F_{harmony}) further contribute to more readable output scores and a better representation of musical structure and grammar. By checking outputs generated from MuseScore, we found that its low performance on F_{pitch} is caused by time shifts introduced when quantising notes according to a constant tempo estimated over the whole music piece. Constant tempo estimation also caused its low performance reported on F_{meter} . A similar limitation can be found in output scores from Finale. On the contrary, our proposed method tracked tempo changes during rhythm quantisation and preserved the expressiveness of music performance as much as possible. This not only benefits metrical alignment, but also results in high accuracy on F_{pitch} and F_{value} . Still, the rhythm quantisation performance (F_{meter}) is far from satisfactory. Some typical errors include double/half tempo errors and errors introduced by missing/extra beat predictions.

To provide a better understanding of the performance of our proposed method, we provide some example outputs from our model together with their performance MIDI recordings at <https://cheriell.github.io/research/PM2S>.

4.3 The audio-to-score transcription pipeline

4.3.1 Introduction

In this section, we introduce how we combine the proposed PM2S model (described in [Section 4.2](#)) with an audio-to-performance MIDI piano transcription system ([Kong et al., 2021](#)) to achieve the goal of audio-to-score piano transcription. We first describe the audio-to-score transcription pipeline in [subsection 4.3.2](#), and then present our experiments in [subsection 4.3.3](#).

4.3.2 Method

We follow the traditional audio-to-score transcription pipeline as described in [subsection 2.2.2](#), which covers the following two steps: 1) audio-to-performance MIDI transcription (A2PM) and 2) performance MIDI-to-score conversion (PM2S).

Audio-to-performance MIDI transcription

The A2PM step works to transcribe music audio recordings into performance MIDIs in the form of a note sequence describing each note’s onset, pitch, duration, and velocity. We use the high-resolution piano transcription system proposed by [Kong et al. \(2021\)](#). This model achieves state-of-the-art results at the time of our experiment and can transcribe piano recordings into performance MIDIs in the format we need for the next step. We use the pre-trained model checkpoint released along the paper to get the transcribed performance MIDIs from music recordings.

Performance MIDI-to-score conversion

The PM2S step works to convert performance MIDIs into a machine-readable music score format. For this step, we use the model we proposed in [Section 4.2](#) which converts performance MIDIs into score MIDIs by doing rhythm quantisation, hand part prediction, time signature prediction, and key signature prediction using a convolutional recurrent neural network.

4.3.3 Experiments

Data

In our experiments, we use the same dataset and dataset split as in [subsection 4.2.5](#). The public pre-trained piano transcription model ([Kong et al., 2021](#)) we use for the audio-to-performance MIDI transcription step was trained on the MAESTRO dataset ([Hawthorne et al., 2019](#)), which does not have an overlap with the test set we use. For the performance MIDI-to-score conversion step, in order to account for the domain shift from recorded performance MIDIs (e.g., MIDI recordings from piano keyboard, which are clean performance data) to transcribed MIDIs (which are usually noisy data with various types of transcription errors), we consider retraining our proposed PM2S model using transcribed MIDIs.

Training data comparison

Since the model we described in [Section 4.2](#) is trained on clean performance MIDIs (recorded from MIDI keyboards, with accurate note onsets, offsets, note values, and velocities), which may not generalise well to transcribed performance MIDI inputs, we compare different training input data for the beat tracking part of our proposed PM2S model:

1. **MIDI-GT**: Using the pre-trained PM2S model developed in [Section 4.2](#), trained on ground truth performance MIDIs.
2. **MIDI-HighRes**: Training a PM2S model using transcribed performance MIDIs obtained from the audio-to-performance MIDI transcription step.
3. **MIDI-Mixed**: Training a PM2S model using both ground truth performance MIDIs and transcribed performance MIDIs.

Whichever MIDI version we use during model training, we evaluate the models using the note-level beat tracking F-measure. We further explore whether including note value in the input data can be helpful or not, to account for more errors in note offset estimation introduced by the audio-to-performance MIDI transcription step.

Table 4.8: Note-level F-measure results for onset beat tracking with different MIDIs for model training, evaluated on the audio-to-score transcription pipeline.

| | MIDI for Training | Note Duration | P_{note} | R_{note} | F_{note} |
|----|-------------------|---------------|-------------------|-------------------|-------------------|
| 1. | MIDI-GT | ✓ | 78.5 | 76.9 | 76.4 |
| 2. | MIDI-HighRes | ✓ | 77.1 | 78.3 | 76.8 |
| 3. | MIDI-Mixed | ✓ | 80.1 | 75.9 | 77.5 |
| 4. | MIDI-GT | ✗ | 75.5 | 68.7 | 71.1 |
| 5. | MIDI-HighRes | ✗ | 76.8 | 72.0 | 73.3 |
| 6. | MIDI-Mixed | ✗ | 76.9 | 72.1 | 73.5 |

In the experiments, we use the same model hyperparameters and training configurations (learning rate, batch size, etc.) for the PM2S model as in [subsection 4.2.5](#). During evaluation, we follow the audio-to-score pipeline to get score MIDIs and compare them with the ground truth score MIDIs.

Results can be seen in [Table 4.8](#). Results in rows 1-3 show that training the PM2S model using transcribed MIDIs does improve the model’s performance. Having both the ground truth performance MIDIs and transcribed performance MIDIs in the training data can further improve the model performance (row 3). Comparing rows 1-3 with rows 4-6, we can see that beat tracking results get worse when removing note duration (more precisely, note duration d_n in [Equation 4.1](#)) from the input note sequence. This suggests note offset information is still helpful to beat tracking, despite possible transcription errors. We use the best configuration (row 3 in [Table 4.8](#)) in this comparative study for the beat tracking part of our proposed PM2S model, while keeping the other parts of the model unchanged, when evaluating the audio-to-score transcription pipeline.

Score-level evaluation

In [Table 4.9](#), we present the results of the pipeline-based methods for audio-to-score transcription, using the MV2H metric described in [subsection 2.4.2](#). We compare two methods for audio-to-performance MIDI transcription (A2PM):

Table 4.9: MV2H evaluation on the ACPAS dataset using the audio-to-score conversion pipeline. The results of HighRes+Finale are labelled as “–” since Finale is no longer supported at the time of our experiments.

| Method | | MV2H Results | | | | | |
|---------|------------------|--------------------|--------------------|--------------------|--------------------|----------------------|-------------------|
| A2PM | PM2S | F_{pitch} | F_{voice} | F_{meter} | F_{value} | F_{harmony} | F_{MV2H} |
| GT | MuseScore | 10.0 | 65.0 | 15.3 | 95.0 | 84.5 | 54.0 |
| GT | Finale | 82.2 | 54.6 | 9.9 | 92.2 | 86.2 | 65.0 |
| GT | HMMs | 91.2 | 71.1 | 51.7 | 91.3 | 77.0 | 76.5 |
| GT | Ours | 99.8 | 87.0 | 61.7 | 99.9 | 91.1 | 87.9 |
| HighRes | MuseScore | 8.5 | 46.3 | 15.2 | 80.5 | 79.3 | 46.0 |
| HighRes | Finale | – | – | – | – | – | – |
| POVNet | Melisma Analyzer | 81.0 | 53.3 | 42.4 | 85.2 | 72.7 | 66.9 |
| POVNet | HMMs | 85.0 | 67.5 | 41.4 | 87.3 | 71.7 | 70.6 |
| HighRes | Ours | 83.5 | 48.5 | 48.2 | 91.6 | 91.4 | 72.6 |

the high-resolution piano transcription model (Kong et al., 2021) (HighRes) and the multipitch detection model proposed in (Shibata et al., 2021) (POVNet). For reference, we also include the results using ground truth (GT) performance MIDIs as input for the next step. We use the public HighRes model, which is pre-trained on the MAESTRO dataset. The POVNet is a CNN-based model trained on the MAPS dataset. For the performance MIDI-to-score conversion (PM2S) step, we compare our method with four other methods: 1) Using the MuseScore software by importing the performance MIDI and exporting the MIDI scores, 2) similarly, using the Finale software, 3) using the Melisma Analyzer approach proposed in (Cogliati et al., 2016), and 4) using an HMM-based statistical method described in (Shibata et al., 2021). In Table 4.9, results on the Melisma Analyzer and the HMMs are those reported in (Shibata et al., 2021), which share the same test set as the one we used in our experiments (the MAPS-ENSTDkCl subset).

Results show that after switching to use transcribed MIDIs, the MV2H results generally get lower in almost every MV2H sub-metric. The decrease in the multi-pitch detection metric (F_{pitch}) may be mainly introduced by errors in the A2PM step. Comparing GT+Ours with HighRes+Ours, a significant

drop can be observed in hand part prediction (F_{voice}). This might be because of the domain shift from recorded performance MIDIs (on which the hand part prediction model is trained) to the transcribed MIDIs (on which the hand part model is evaluated). Rhythm quantisation (F_{meter} and F_{value}) also showed a similar decreased accuracy when using transcribed MIDIs. While we adapted our model to be trained on transcribed MIDIs, the task (i. e., rhythm quantisation) itself may be more difficult on noisy data (transcribed MIDIs are usually noisier than recorded performance MIDIs). On the other hand, there can also be accumulated errors introduced by the audio-to-performance MIDI transcription (A2PM) step.

Comparing the five PM2S methods, we observe a trend that statistical method (Melisma Analyzer and HMMs) tends to be better than commercial software (MuseScore and Finale), while our deep learning method outperforms statistical methods in average MV2H F-score (F_{MV2H}). Still, the HMMs achieved the best results on the multi-pitch estimation and hand separation sub-metrics (F_{pitch} and F_{voice}).

Since this chapter focuses primarily on performance MIDI-to-score conversion, so we do not include results on the MuseSyn dataset here. Instead, these results are presented in [Chapter 5](#), as the MuseSyn dataset consists of score MIDIs rather than performance MIDIs. In [Chapter 5](#), we compare the pipeline-based methods with holistic methods on both the MuseSyn dataset and the ACPAS dataset.

4.4 Conclusion

In this chapter, we introduced our proposed method for audio-to-score piano transcription using a traditional automatic music transcription pipeline, i. e., audio-to-performance MIDI transcription (A2PM) followed by performance MIDI-to-score conversion (PM2S).

We focused on the performance MIDI-to-score conversion (latter) step of the pipeline, especially the more difficult part of rhythm quantisation. We propose to solve rhythm quantisation by neural beat tracking using a convolutional recurrent neural network, which predicts whether notes are at a beat or not. We explored different input data encoding and data augmentation

methods on the beat tracking model and found that note onset is the most important input feature in beat prediction, and it is best to encode it into a one-hot onset-shift vector. Among the data augmentation methods explored, tempo change benefits the most. We validate our model’s beat tracking ability in comparison with a baseline beat tracking model, which uses a pianoroll input and a temporal convolutional neural network structure. After combining the prediction of other musical score elements (i. e., hand parts, time signatures, key signatures), we evaluate our proposed performance MIDI-to-score conversion method on the MV2H metric in comparison with two commercial software packages (i. e., MuseScore and Finale). Results suggest the benefit of using the proposed method, highlighting the benefit of modelling expressive tempo changes.

We further combine our proposed performance MIDI-to-score conversion method with an existing audio-to-performance MIDI transcription model (Kong et al., 2021), resulting in a pipeline-based audio-to-score piano transcription system. We explored the use of different training data for the beat tracking part of the PM2S model, including using ground truth performance MIDI and transcribed MIDI. Results showed a benefit of using a mixture of both MIDI versions. Evaluated on the MV2H metric, score-level results on the audio-to-score pipeline showed a decrease in the system performance when shifting from the use of ground truth performance MIDI to transcribed performance MIDI, which suggests possible accumulated errors from the first step of the pipeline (i. e., audio-to-performance MIDI transcription), and potential difficulties introduced by the domain shift from clean data (ground truth performance MIDI) to noisy data (transcribed MIDI) in the latter step of the pipeline (i. e., performance MIDI-to-score conversion). Still, the overall evaluation result of the audio-to-score transcription pipeline is better than using existing music notation software (MuseScore) or statistical methods (Melisma Analyzer or HMMs).

There are still a lot of limitations in the proposed system. One of the biggest limitations is in rhythm quantisation, which shows the worst result among all the MV2H sub-metrics. Possible ways for improvement include using more powerful model architectures, such as Transformers (Vaswani et al.,

2017) and improving non-onset beat predictions and post-processing using methods like dynamic Bayesian networks (Böck & Davies, 2020) or local periodicity (Chiu et al., 2023). Further directions include expanding the output data to generate a machine-readable score in the form of MusicXML (Suzuki, 2021) and probing our system’s ability to deal more with genre and instrumentation (Shibata et al., 2021; Tanaka et al., 2020; Wu et al., 2020).

Chapter 5

Holistic Methods for Audio-to-Score Piano Transcription

5.1 Introduction

Following an exploration of the traditional pipeline-based methods for audio-to-score piano transcription, we switch our focus to holistic methods in this chapter. As we discussed in [Chapter 2](#), recent literature has mainly focused on two approaches for audio-to-score transcription: 1) traditional *pipeline-based* methods transcribe music audio step by step in the order of subtasks ([Nakamura et al., 2018](#); [Shibata et al., 2021](#)), and 2) *holistic* (or end-to-end) methods design algorithms that directly convert an audio input to a score format in the form of Western staff notation ([Carvalho & Smaragdis, 2017](#); [Román et al., 2018](#); [Román et al., 2019](#); [Román et al., 2020](#)). Compared to traditional methods, holistic methods may have advantages in minimising the risk of accumulated errors in different steps.

In this chapter, we investigate holistic methods for audio-to-score transcription. As we discussed in [Chapter 4](#), the first attempt towards audio-to-score transcription with holistic methods adopts sequence-to-sequence models as presented in ([Carvalho & Smaragdis, 2017](#)), which proves audio-to-score can be performed in an holistic manner. However, the work is limited to monophonic or two-melody polyphonic music, constrained note durations and 4/4

time signatures. Another holistic audio-to-score system developed in (Román et al., 2018; Román et al., 2019; Román et al., 2020) makes use of a convolutional recurrent neural network (CRNN) with connectionist temporal classification (CTC) loss. It uses a CTC-friendly score data representation to achieve higher performance. However, the system is still limited to monophonic transcription or a simple case of polyphonic transcription (string quartets or Bach chorales). Another limitation among those above-mentioned holistic audio-to-score methods is that they only output a beat-quantised musical notation, lacking important descriptive information in music analysis, such as absolute time for note onsets and offsets.

Unlike previous works which target solely a musical “score” representation (without any performance information), in this chapter, we try to predict both a *performance-level transcription*, in the form of a piano-roll representation or performance note sequence, and a *score-level transcription*, in the form of symbolic music notation. This can be beneficial in real-world applications in the sense of not only providing the transcribed music score notation, but also enabling various applications that require a performance-level transcription, such as expressive performance analysis and performance-to-score alignment. In our work, we also explore how the learning of both performance and score can affect the ability of the transcription system.

The remaining of this chapter covers mainly two parts: 1) Section 5.2 describes our first step towards a joint multi-pitch detection and score transcription system using a multitask model composed of a CRNN and RNN-based sequence-to-sequence models with attention mechanisms; 2) Section 5.3 introduces further exploration towards better score transcription using a long short-term decoding strategy with Transformers. At the end of this chapter, we provide a brief conclusion in Section 5.4.

5.2 Joint multi-pitch detection and score transcription

5.2.1 Introduction

In this section, we introduce our first holistic audio-to-score system for joint multi-pitch detection and score transcription. We observe that previous literature on holistic audio-to-score transcription (Carvalho & Smaragdis, 2017; Román et al., 2018; Román et al., 2019; Román et al., 2020) is limited to monophonic transcription or a fixed polyphony level transcription. In this work, we intend to extend the use of holistic methods for audio-to-score to a more general application scenario of polyphonic piano music with varying polyphony levels, also, to support the estimation of music performance characteristics in a piano-roll format. We propose a multitask holistic model composed of convolutional layers, recurrent layers and sequence-to-sequence models with an attention mechanism for audio-to-score transcription, which is, to our knowledge, the first holistic model that transcribes polyphonic piano music into both a piano-roll format (corresponding to a descriptive notation of the music audio) and a score in Western staff notation (corresponding to a prescriptive notation of the musical audio). Additionally, we propose a modified score representation based on the LilyPond format for modelling polyphonic music, which learns and predicts 7 times faster and performs better than the LilyPond format (Nienhuys & Nieuwenhuizen, 2003) score representation on this model and tests the effect of using different input time-frequency representations, including Short Time Fourier Transform, Mel Spectrogram, Constant-Q Transform, Variable-Q Transform (Schörkhuber et al., 2014) and Harmonic Constant-Q Transform (Bittner et al., 2017).

In the following sections, we introduce our proposed score data representation (subsection 5.2.2), the multitask learning model architecture (subsection 5.2.3), our comparative experiments on different input and output representations as well as single- or multitask learning (subsection 5.2.4), and a short conclusion in subsection 5.2.5.

5.2.2 Data representations

Audio data representations

Like in most AMT systems, we use as input a time-frequency representation of the audio signal, and we compare commonly used time-frequency representations. All representations are log-valued and the signals are resampled to ensure the hop size of every spectrogram being equal to 10ms, which means equal length in the model input. The time-frequency representations we compare are:

- *STFT* - Magnitude spectrogram from the Short Time Fourier Transform with a Hanning window and FFT window length in $\{1024, 2048\}$. Signal sampling rate is 44.1kHz.
- *Mel Spectrogram* - Mel Spectrogram with different FFT window length in $\{1024, 2048\}$ and different number of Mel bands in $\{128, 192, 256\}$. Signal sampling rate is 44.1kHz.
- *Constant-Q Transform (CQT)* - Spectrogram obtained from the Constant-Q Transform (Brown, 1991), with bins per octave in $\{12, 24, 36, 48, 60\}$, number of octaves in $\{7, 8\}$ and lowest frequency equal to pitch $A_0=27.5$ Hz, which is the lowest pitch in piano. Signals are resampled at 25.6 kHz to fit a hop length of 256.
- *Harmonic Constant-Q Transform (HCQT)* - Spectrogram from the Harmonic Constant-Q Transform proposed in (Bittner et al., 2017), which is a 3-dimensional spectrogram with CQTs based on shifted harmonics. The parameters we select from are bins per octave in $\{36, 60\}$, number of octaves in $\{5, 6\}$ and number of harmonics in $\{4, 5, 6\}$. Signals are resampled at 25.6kHz.
- *Variable-Q Transform (VQT)* - Spectrogram calculated from Variable-Q Transform proposed in (Schörkhuber et al., 2014). We select γ values in $\{10, 20, 30\}$, number of bins per octave in $\{36, 60\}$ and number of octaves in $\{7, 8\}$. Signals are resampled at 25.6kHz.

Score data representations

One of the major challenges in developing an holistic audio-to-score system is selecting an output representation that can support polyphonic music and includes various cues present in music scores. Compared to sentence outputs in ASR tasks, music notation is much more structured and complex. Some of the most commonly used symbolic music score encoding formats are MusicXML, **Kern, LilyPond, ABC and PAE (Müller, 2015). However, MusicXML is a verbose music encoding, and the **Kern format only supports monophonic music per voice. All the other three formats support polyphonic music and encode music scores into strings. Here, we use the LilyPond (Nienhuys & Nieuwenhuizen, 2003) format as a base representation for our score data representation.

Although much more concise than the MusicXML format, LilyPond encoding is still complex with hierarchical structures such as parts and voices. To make the transcription task simpler, we assume that there are only two hand parts in piano music and only one voice per hand, where each voice can have multiple concurrent notes. We consider the left hand part and right hand part scores as two outputs predicted jointly. In this way, we discard the hierarchical structure of the LilyPond format and keep the most essential information in two strings. We assume our model to predict one bar at a time, this means that we only take into account the notes and rests in our score data representation, no barline or key/time signature symbols are included. At this stage, we do not consider adding playing techniques such as arpeggios, trills, vibratos, or rhythm structures such as triplets and quintuplets. The symbols we use in LilyPond format are:

- *Pitch* - Combined with pitch chroma (e.g. ‘c’ for C, ‘cis’ for C♯ and ‘ces’ for C♭) and pitch height (e.g. ‘ ’ for higher octave and ‘ , ’ for lower octave, duplicate e.g. ‘ ” ’ for double octaves).
- *Duration* - We use numbers to represent durations, e.g. ‘8’ for an 8th note duration (duration symbol can be omitted for a 4th note duration). The same duration representation is used for chords and rests. ‘ . ’ is added for dotted notes - resulting in e.g. ‘4.’.

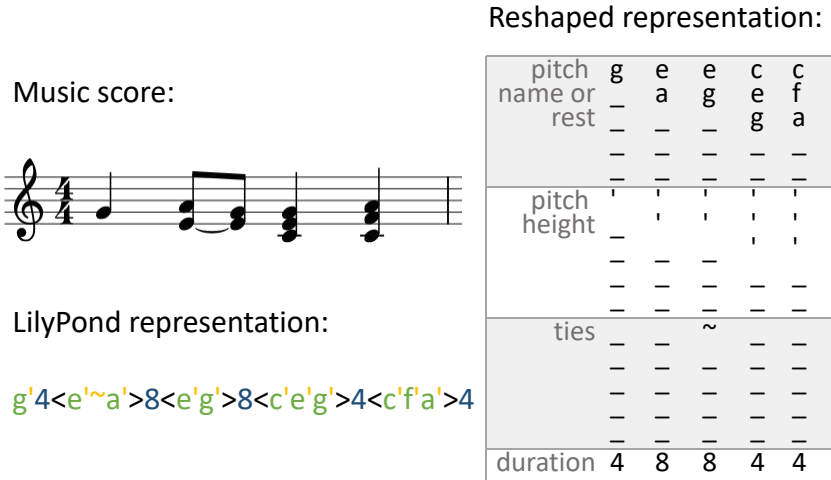


Figure 5.1: Example music score and corresponding LilyPond and Reshaped representation.

- *Rest/Note/Chord* - A rest is represented as ‘*r*’ followed by its duration symbol. A note/chord is represented by its pitch(es) and duration. Chord pitches are grouped by brackets (e.g. pitches for a C major starting with a middle C is ‘*c' e' g'* ’)
- *Tie* - Ties are represented using ‘~’, added to its start note, such as ‘*c4 ~ c8*’ for a tied note *c*, or ‘*c ~ e g)4(c f a)2*’ for a tied *c* in chord.

Based on the above defined musical symbols, we compare the following two score data representations; an example for the two score representations can be seen in [Figure 5.1](#).

- *LilyPond representation* - A representation based on LilyPond encoding by removing extra symbols and keeping only the described necessary symbols to reconstruct a musical score.
- *Reshaped representation* - Considering the length of a LilyPond score representation and the difficulty in learning structural information, we propose a Reshaped data representation based on the LilyPond representation that describes a score in a 2D matrix of symbols. We assume

a maximum of five concurrent notes per hand, one for each finger, and the 2D matrix of symbols is indexed by symbol index and time, where each time step consists of $(5+5+5+1=)16$ symbols corresponding to five symbols for each one of pitch names or rest, pitch heights, ties and one symbol for duration. Each column of the matrix can reconstruct a rest, note or a chord in a music score.

5.2.3 Model architecture

Based on the data representations described above, we build a model¹ that predicts a piano-roll Y_p and a music score representation $Y_s = \{S_r, S_l\}$ from a music audio spectrogram input X , where S_r and S_l are the sequences of score symbols for the right and left hand parts. That is to find:

$$Y_p, \{S_r, S_l\} = \underset{Y_p \in \mathcal{U}_p, S_r \in \mathcal{U}_s, S_l \in \mathcal{U}_s}{\operatorname{argmax}} P(Y_p, \{S_r, S_l\} | X) \quad (5.1)$$

where \mathcal{U}_p is the universal set of possible piano-rolls and \mathcal{U}_s is the universal set of score representation sequences for one hand part. Although Y_p and X are strictly time-aligned, S_r, S_l are not aligned with X and have different lengths.

We design the model as a multitask learning model (Ruder, 2017) with a shared convolutional stack and three separate sequential models, corresponding to Y_p, S_r , and S_l . The convolutional stack of the model has four convolutional layers, with a pooling layer in the middle of the stack following the second convolutional layer, and a linear layer following the last convolutional layer. We use two layers of bi-directional Gated Recurring Units (GRU) followed by a linear output layer with Exponential Linear Unit (ELU) activation for the piano-roll prediction, resulting in a Convolutional Recurrent Neural Network (CRNN) together with the shared convolutional stack. For scores, since the LilyPond score representation is not always shorter than the input audio spectrogram in term of time steps (which is a pre-requisite of using a CTC loss) we adopt the more universal sequence-to-sequence model originally proposed in neural machine translation (Sutskever et al., 2014), and use an attention mechanism (Bahdanau et al., 2015) to make the model capable of

¹Code is publicly available online at:

<https://github.com/cheriell/ICASSP2021-audio-to-score>

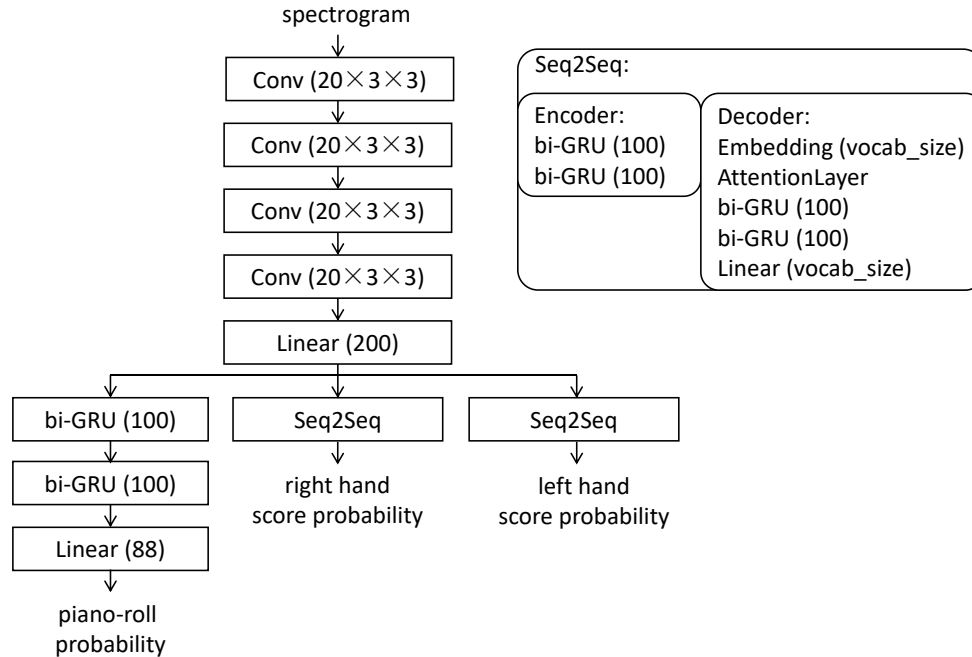


Figure 5.2: Model structure with layer sizes.

learning long sequences. Two sequence-to-sequence models are added following the shared convolutional stack, for the right hand score S_r and the left hand score S_l respectively. The encoder of a sequence-to-sequence model is composed of two bi-directional GRU layers, and the decoder consists of one (for LilyPond score representation, or 16 for Reshaped score representation) embedding layer(s), one attention layer that attends to the encoder output, two one-directional GRU layers and one (for LilyPond representation, or 16 for Reshaped representation) linear output layer(s) with log-valued softmax activation. A diagram explanation of our proposed model architecture and the dimensions of each layer can be found in [Figure 5.2](#).

Table 5.1: Statistics of the MuseSyn dataset. For polyphony levels, numbers out of brackets are calculated without adding piano pedals; numbers in brackets are calculated with piano pedals.

| Statistic | Details |
|-------------------------|-------------------------------|
| Number of music pieces | 210 |
| Total hours | 9.62×4 piano models |
| Total notes | 222,219 |
| Use of piano pedal | 29% (61 pieces) |
| Maximum polyphony level | 13 (26) |
| Average polyphony level | 2.87 (3.21) |
| Time signatures | 4/4, 3/4, 5/4, 6/8, 9/8, etc. |
| Key signatures | all 12 key signatures |

5.2.4 Experiments

Data

We use the MuseSyn dataset mainly covering popular piano music described in Chapter 3, we refer to the readers to Chapter 3 for details of how this dataset is created. Table 5.1 gives a summary of the dataset statistics. We do this as a starting point and because there is a lack of AMT datasets that provide score ground truth on both physical and musical time. A dataset that best fits this task is the ASAP dataset (Foscarin et al., 2020), which was only published shortly before we summarised this work into a paper, thus, we considered it to be investigated in our future work.

We use a train/validation/test ratio of 8:1:1 in our experiments, which corresponds to 168 pieces for training, 21 pieces for validation and 21 pieces for testing. Among the four piano models in the dataset, we use three piano models in training and validation, and all four piano models in testing. We keep the same train/validation/test split throughout all experiments.

Training and inference

In our experiments, all the input audio spectrograms are calculated using librosa (McFee et al., 2020). The Python package pretty_midi (Raffel & Ellis, 2014) is used to extract information from MIDI files. We use the default velocity-valued piano-roll in pretty_midi as our piano-roll reference, and the ground truth downbeat times to cut audio recordings into bars. Zero padding is added to the input audio spectrograms and piano-rolls. For score representations, we split them into lists of symbols and consider the symbols as tokens in a sentence like tokens in natural language processing problems (Goldberg, 2017). SOS, EOS and PAD symbols are added to the sentences. Symbols are encoded by one-hot encoding. For the Reshaped representation, the symbols are separately one-hot encoded, that is, we use separate index dictionaries for pitch name, pitch height, tie and duration symbols.

For the piano-roll part, a minimum squared error loss is used for the output during training. To obtain a binary piano-roll during inference, we threshold the model’s output with a velocity value of 30, since low velocities are not audible. No post-processing steps like smoothing, minimum duration pruning or gap filtering are applied. The obtained note sequences form the binary piano rolls for note-level evaluations.

For the score part, a negative log-likelihood loss is used for the score representation. A 50% teacher forcing ratio (Goldberg, 2017) is used in the training process. This means the models use the ground truth in the decoding process with a 50% probability at each time step, otherwise it uses the most probable predicted output symbol. During inference, no teacher forcing is used, and we simply adopt a greedy decoding to obtain the predicted output sequence. To further obtain a full score transcription from the models, we combine the predicted scores for all bars, and post-process the scores by obtaining the most probable time signature, adding missing rests and removing extra rests. The final score representation can be directly decoded to MusicXML format.

Experimental setup

Comparison of time-frequency representations. In this experiment, we compare the performance on the use of the STFT, Mel Spectrogram, CQT,

HCQT and VQT as input representations and their different parameters described in [subsection 5.2.2](#) for multi-pitch detection. We assume the best input representation for multi-pitch detection will perform well on score transcription, since the two tasks are highly related. Hence, we use only the piano-roll part of our proposed model, removing the pooling layer in the convolutional stack to achieve higher resolution. Results for this comparison are evaluated only on the three piano models used in model training.

Comparison of score representation. We use only the shared and score component to compare the two score representations described in [subsection 5.2.2](#), keeping the convolutional stack and the two sequence-to-sequence models for score prediction. We call the two models score-only model with LilyPond or Reshaped representation. The best-performing spectrogram from the previous comparison is used as our input representation.

Combination of piano-roll and symbolic score. Using the best input and score representation from the above, and the full multi-task model, we train a Joint model that simultaneously predicts a piano-roll transcription and a symbolic score transcription. We compare this new model with piano-roll only and score-only models to see how the two tasks influence each other.

Evaluation metrics

For the input representation experiment, we evaluate the performance with the AMT benchmark frame-level and note-level onset only and onset-offset evaluation metrics ([Bay et al., 2009](#)). We use an onset tolerance of 50ms and keep the offset tolerance to be 20% of the note duration, or 50ms, which ever is larger, as is used in the MIREX public evaluations ([Bay et al., 2009](#)).

For score evaluation, since there is no existing standard audio-to-score evaluation metric, we use the following two metrics as an indication of the system’s performance:

- *Word Error Rate (WER)* of the LilyPond representation, adopted from neural machine translation tasks ([Goldberg, 2017](#)). For the Reshaped representation, we first reconstruct the output to the LilyPond format

Table 5.2: F-measure of piano-roll prediction on different input representations and models. F_f : frame-level, F_{on} : note-level onset only, F_{onoff} : note-level onset and offset.

| Input representations | F_f | F_{on} | F_{onoff} |
|-----------------------|-------------|-------------|-------------|
| STFT | 89.5 | 81.0 | 61.7 |
| Mel Spectrogram | 89.0 | 82.1 | 63.0 |
| CQT | 91.9 | 85.4 | 67.4 |
| HCQT | 91.0 | 84.1 | 65.3 |
| VQT | 91.9 | 85.7 | 68.5 |

and then calculate the word error rate to the ground truth LilyPond representation.

- *MV2H metric* proposed in (McLeod & Steedman, 2018a) for complete AMT evaluation, composed of five sub-metrics: multi-pitch detection accuracy (F_{pitch}), voice separation accuracy (F_{voice}), metrical alignment accuracy (F_{meter}), note value detection accuracy (F_{value}). The harmonic analysis submetric is not included, since we do not include key detection and chord estimation in our experiments. We redefine the overall accuracy of this metric (F_{MV2H}) as the average over the four sub-metrics. In this work, we use v1.0 of this metric, assuming our transcription and the input audio are time-aligned.

Results

Multi-pitch estimation results. Table 5.2 shows the F-scores of benchmark frame-level and note-level evaluation metrics on the model predictions on different input audio spectrograms with their best performing parameters, and on the Piano-roll only model and Joint model under the best input representation we find. Among the five spectrogram types, VQT shows the best performance, with a γ value of 20, and 8 octaves \times 60 bins per octave in the frequency axis. No large performance difference on this metric is found between the Piano-roll only model and the Joint model. For more detailed results on other parameters of each input time–frequency representation, we

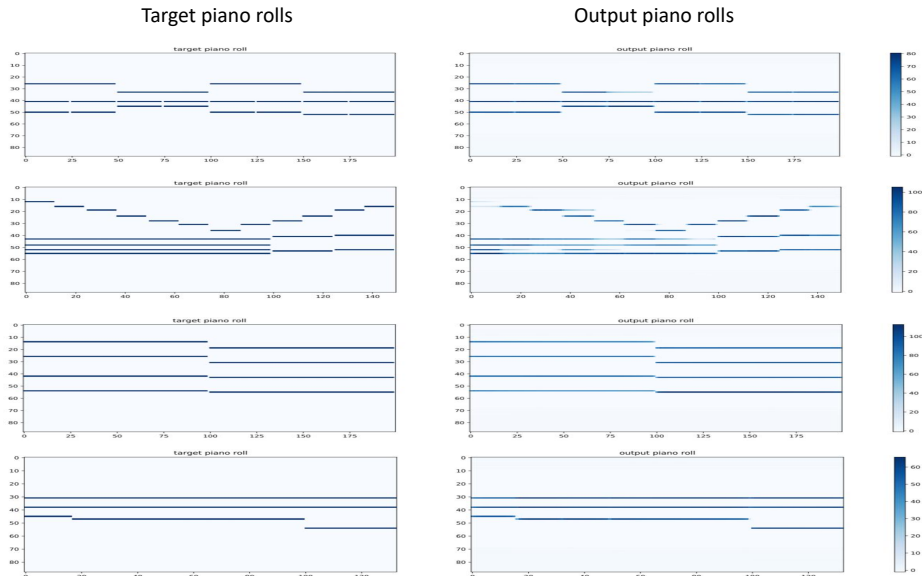


Figure 5.3: Example velocity-valued piano roll targets and outputs from model with the best input representation. Left column: ground truth piano rolls, Right column: transcribed piano rolls (non-binary, as predicted probabilities).

refer readers to [Section A.1](#) in the Appendix.

[Figure 5.3](#) includes example ground truth and transcribed piano rolls using the best input representation. Generally, the pitches are correctly predicted. However, the model is not very good at predicting short gaps between repeated notes, which results in merged note errors.

Score transcription results. [Table 5.3](#) shows the WER evaluation and MV2H evaluation results on score prediction for the two Score-only models and the Joint model we described in [Table 5.2.4](#). All inputs are VQT spectrograms with 8×60 frequency bins and a γ value equal to 20. Results show the Reshaped representation is slightly better than the LilyPond representation in both metrics. The Reshaped representation also outperforms the LilyPond representation in terms of the time and memory resources required (around 7 times faster and half the memory). The Joint model performs better than the

Table 5.3: Word error rates and MV2H results in percentage for different score representations. LilyPond: Score-only model with LilyPond representation; Reshaped: Score-only model with Reshaped representation; Joint: Joint model with Reshaped representation. ϵ_{right} and ϵ_{left} refer to the word error rates for the transcriptions for the right-hand part and left-hand part.

| WER | ϵ_{right} | ϵ_{left} | ϵ | | |
|----------|---------------------------|--------------------------|--------------------|--------------------|-------------------|
| LilyPond | 38.0 | 39.0 | 38.5 | | |
| Reshaped | 37.8 | 34.5 | 36.2 | | |
| MV2H | F_{pitch} | F_{voice} | F_{meter} | F_{value} | F_{MV2H} |
| LilyPond | 66.7 | 90.3 | 94.8 | 93.2 | 86.3 |
| Reshaped | 69.6 | 89.7 | 94.8 | 93.7 | 86.9 |

Table 5.4: Model performances on single-task and multi-task models.

| F-measure | F_f | F_{on} | F_{onoff} | | |
|-----------------|---------------------------|--------------------------|--------------------|--------------------|-------------------|
| Piano-roll Only | 86.4 | 67.6 | 52.0 | | |
| Joint | 88.0 | 66.7 | 53.6 | | |
| WER | ϵ_{right} | ϵ_{left} | ϵ | | |
| Score-only | 37.8 | 34.5 | 36.2 | | |
| Joint | 37.6 | 35.3 | 36.5 | | |
| MV2H | F_{pitch} | F_{voice} | F_{meter} | F_{value} | F_{MV2H} |
| Score-only | 69.6 | 89.7 | 94.8 | 93.7 | 86.9 |
| Joint | 71.1 | 90.8 | 94.9 | 94.4 | 87.8 |

Score-only model with Reshaped representation in terms of MV2H evaluation metric, which suggests an advantage in adding the piano-roll prediction task.

In Table 5.5, we show the time and space resources required by the two score representations. Training time and memory are measured as average values per epoch with a batch size of 8, and inference time and memory are measured for predicting scores of the whole test set. Score post-processing time is

Table 5.5: Time and memory used in training and inference for the Score-only model with LilyPond and Reshaped representations.

| | Score rep. | Time (hh:mm) | Memory (MB) |
|------------------|------------|--------------|-------------|
| <i>Training</i> | LilyPond | 19:55 | 10,817 |
| | Reshaped | 2:36 | 5,043 |
| <i>Inference</i> | LilyPond | 11:22 | 2,555 |
| | Reshaped | 2:03 | 1,677 |

not included in the measurement. We notice time and memory consumption of the Reshaped representation is much less than that of the LilyPond representation. Moreover, the LilyPond representation inference time is greater than the duration of the music recording, and the Reshaped representation inference time is shorter than the duration of the music recording.

Two example ground truth scores and their transcriptions from the Joint model are in Figure 5.4. In the examples, the model generally captures the notes and meters and can transcribe the melodies to some extent. Main errors found in the two examples include octave errors, note duration errors, extra/missing note errors and voice separation errors. Especially, in the last measure of the second example (Figure 5.4-(b)), a lot of mistakes can be found in the left-hand score.

5.2.5 Conclusion

Our results suggest the benefit of multitask learning in audio-to-score transcription, as well as the feasibility of using a sequence-to-sequence model for the task. This benefit can be attributed to the many common musical features shared between the piano roll and the musical score, such as pitch, onset, and duration; predicting both jointly encourages the model to learn these shared representations more effectively through a common latent space. Results also show that among the five audio input representations in our comparison, VQT achieved the best results. This outcome is consistent not only with the quantitative evaluation but also with the musical properties of VQT: while CQT benefits from a logarithmic frequency scale that is well-suited for represent-

Example 1: Far Horizons (Piano Cover)

Calmly

(a)

Example 2: His Theme - Piano

(b)

Figure 5.4: Example transcriptions. GT: ground truth music score. TR: transcribed music score. Ground truth key signatures are used for visualisation purposes. Transcription outputs for all pieces in the test set are available in our online repository at <https://github.com/cheriell/ICASSP2021-A2S>

ing musical pitches, its time resolution decreases towards lower frequencies, which can compromise accurate note timing information. VQT addresses this

by introducing a variable-Q factor that balances time resolution across different frequency ranges, making it a more suitable input representation for transcription tasks where precise note timing is essential. For the output score representation, we find that a reshaped score representation achieves competitive results with much lower computational complexity.

However, our system remains constrained to transcribing pre-segmented bars using ground-truth downbeats, which limits its applicability in real-world scenarios. Moreover, the relatively low performance in multi-pitch estimation indicates a substantial gap between our system and state-of-the-art frame- and note-level transcription approaches. In our subsequent study, we address these limitations by introducing a fully automatic data processing pipeline and incorporating the Transformer architecture.

5.3 Joint note- and score-level transcription via long short-term decoding

5.3.1 Introduction

Following our exploration in [Section 5.2](#), we further extend our study on holistic methods for audio-to-score transcription using the Transformers architecture ([Vaswani et al., 2017](#)), which has been proven to be effective in various machine learning tasks. In this part, we particularly pay attention to the temporal dependencies of different musical elements. We categorise musical elements into two types: those that are more short-term and rely primarily on local information (e.g., pitch and onsets), and those that are more long-term and dependent on broader temporal structures (e.g., meter and measure). In the rest of this chapter, we refer to them as *short-term musical elements* and *long-term musical elements*.

In this study, we investigate how such categorisation influences the performance of our audio-to-score transcription system. To enable the joint learning of both short-term and long-term musical elements, we introduce an event-based music representation that unifies their encoding. Furthermore, we propose a long short-term decoding strategy, in which two separate decoders operate in successive stages, each focusing on one type of musical element.

We compare this two-stage decoding strategy with a conventional single-stage approach and evaluate its impact on transcription performance based on sub-metrics tailored to different musical aspects. Additionally, we also examine the incorporation of beat information within the transcription task and analyse its effect on model performance.

The succeeding sections are organised as follows. We first introduce our proposed event-based music representation in [subsection 5.3.2](#), followed by our proposed long short-term decoding strategy in [subsection 5.3.3](#). Our experiments and their results are described in [subsection 5.3.4](#) and [subsection 5.3.5](#). Finally, we briefly summarise our findings in [subsection 5.3.6](#).

5.3.2 Event-based music representation

In this section, we describe our proposed event-based music representation, which works as the output data representation for our transcription system. For the input data representation, we refer to [subsection 5.3.4](#) for details.

Overview

In [Figure 5.5](#), we provide an example of our proposed event-based music representation. We define a music excerpt as a list of note events

$$\mathbf{Y} = \{\mathbf{n}_i\}_{i=1}^{N^{\text{notes}}}, \quad (5.2)$$

where, similar to what we did in [Chapter 4](#), notes are ordered by firstly onset time and secondly pitch. Here, each note is defined as a group of eight musical elements:

$$\mathbf{n}_i = (p_i, o_i, d_i, h_i, l_i, b_i, m_i, v_i), \quad (5.3)$$

where the first four are *short-term musical elements*, representing note pitch (p_i), onset (o_i), duration (d_i) and hand part (h_i); the latter four are *long-term musical elements*, representing measure length (l_i), beat division (b_i), musical onset (m_i) and note value (v_i). We adopt this categorisation because elements such as pitch, onset, offset, and hand part primarily depend on local information. In contrast, the latter four elements are closely tied to musical meter and require modelling longer-term temporal dependencies.

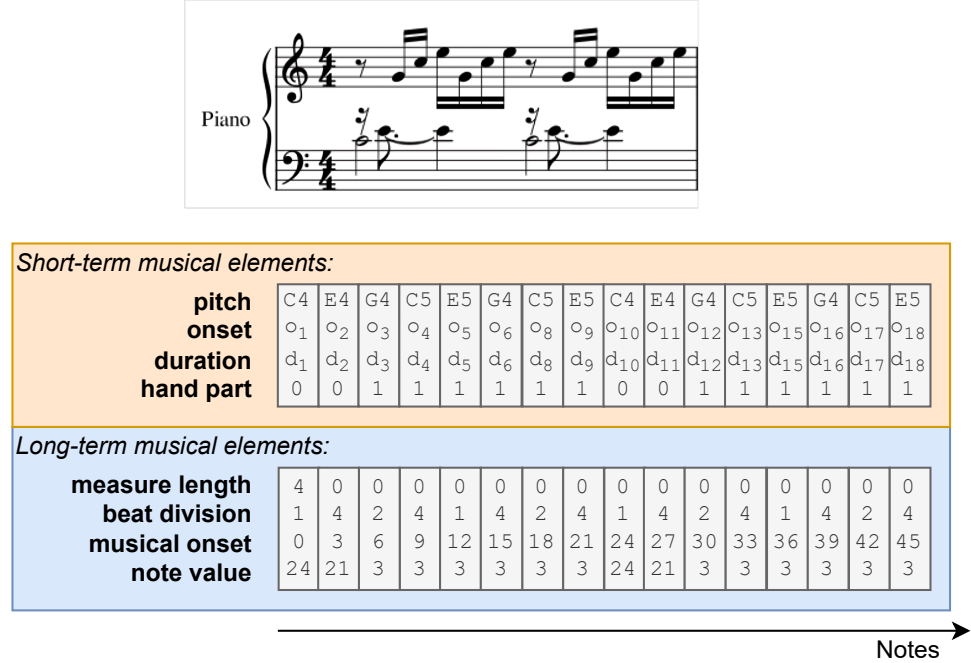


Figure 5.5: An example overview of the proposed event-based music representation.

Following the above, we can obtain a short-term part of the note sequence \mathbf{Y}^s and a long-term part of the note sequence \mathbf{Y}^l as below.

$$\mathbf{Y}^s = \{\mathbf{n}_i^s\}_{i=1}^{N^{\text{notes}}}, \quad \text{where } \mathbf{n}_i^s = (p_i, o_i, d_i, h_i), \quad (5.4)$$

$$\mathbf{Y}^l = \{\mathbf{n}_i^l\}_{i=1}^{N^{\text{notes}}}, \quad \text{where } \mathbf{n}_i^l = (l_i, b_i, m_i, v_i), \quad (5.5)$$

This representation allows us to capture note sequences in a musical performance (through pitch, onset, and offset) as well as in a musical score (through hand part and meter-related long-term elements). It thereby aligns with our goal of modelling both performance-level and score-level transcription, encompassing both short-term and long-term musical elements. We'll further explain how we define each musical element, respectively.

Short-term musical elements

Pitch. We define pitch as the MIDI pitch numbers ranging from 0 to 127, covering all 128 MIDI pitch numbers. In [Figure 5.5](#), we used pitch names instead for better readability.

Onset. Onset is defined as the quantised musical onset time from the beginning of the music recording, with a time resolution of $\tau = 0.01$ s. Although our findings in [Chapter 4](#) suggested that it can be beneficial to use a relative time encoding, in this study, we opt to use the absolute time of the musical onset to avoid accumulating errors during the decoding process.

The maximum onset value corresponds to the maximum length of the music recording. Additionally, we define a special value `<tie_left>` for note onset, for those notes that started before the beginning of the music recording segment.

Duration. Instead of using note offsets (such as in ([Hawthorne et al., 2021](#))), we choose to use note duration instead to simulate a better local dependency between the output music representation and the embeddings from the input audio recording, assuming that the encoder can learn a latent space that aligns well with note-level information in the output representation.

Similar to onset, we define the duration as quantised time, with the same time resolution $\tau = 0.01$ s. We limit the maximum note duration value to the maximum length of the music recording. For those notes that continue after the end of the music segment, we define their duration as `<tie_right>`. And for those notes that started before the beginning of the music segment, we account for only the duration within the current music segment.

Hand part. We define the hand part using only two values: 0 for the left hand (or lower staff) and 1 for the right hand (or upper staff). Unlike in [Section 5.2](#), where the left and right hand parts were separated into two output sequences, we represent the hand part as a single element rather than using separate sequences. This design choice may facilitate future extensions, for example, when incorporating more voices.

Long-term musical elements

Measure length. We define the measure length to capture the downbeats or barlines in the musical score. Specifically, we assign a measure length value only to the first note of each measure, while setting the value to 0 for all other notes. This definition enables us to infer barline positions and time signatures from the score. In practice, the measure length is defined as the duration in ticks in the MIDI score. In [Figure 5.5](#), we used the length in beats instead for better readability.

Beat division. We incorporate beat division to account for beat-level information during the transcription process. Unlike traditional beat-tracking methods that estimate beat times directly, we encode beat information in terms of beat-level subdivisions. Specifically, we define a beat division value b_i for each note, which indicates the metrical subdivision of the note within a beat. In this formulation, a note that happens exactly on beat has $b_i = 1$, while a note occurring at half beat has $b_i = 2$, and so on (i.e., the note is placed at a k/b_i fraction of the beat, with the minimum possible b_i).

Musical onset. We define musical onset as the absolute musical position of the note, with a resolution of 12 subdivisions per quarter note, which is fine enough to describe most notes (see statistical analysis in [Chapter 3](#)). Similar to onsets, we use `<tie_left>` to represent those notes that started before the music segment.

Note value. We round the note values using the same resolution of 12 subdivisions per quarter note as for musical onsets. Similar to how we defined note duration, we use a `<tie_right>` to represent the notes that last beyond the music segment. For those notes that started before the segment, we account for only the note value within the current music segment.

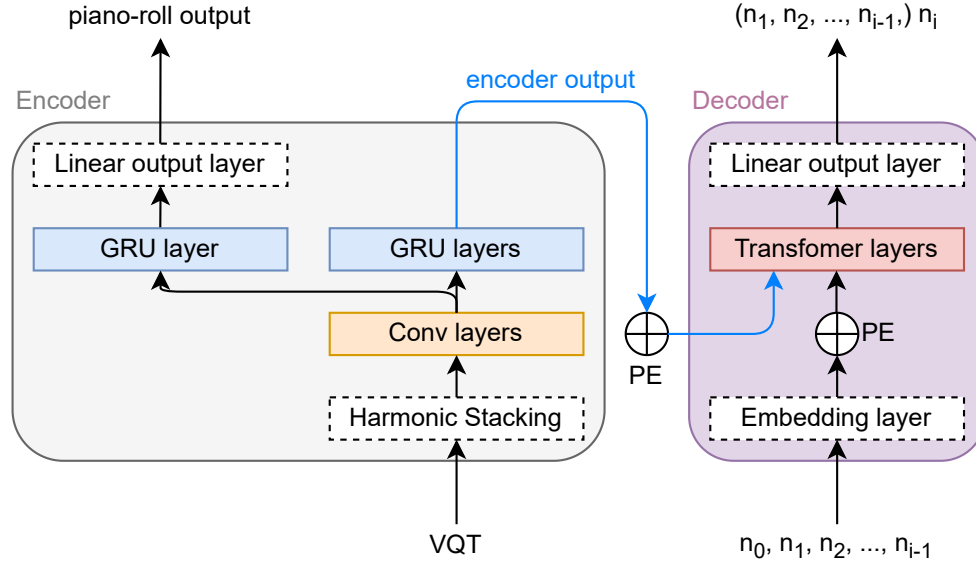


Figure 5.6: Baseline model architecture. PE is short for positional encoding. We start the output sequence with n_0 to account for the `< sos >` token in practice.

5.3.3 Long short-term decoding strategy

Baseline model

Before introducing our proposed long short-term decoding strategy, we provide a description of the baseline model in this study, which is adapted from our previous study described in Section 5.2. Figure 5.6 shows an overview of the model architecture.

For the encoder, we adopt the VQT input representation and extend it with a harmonic stacking operation, enabling the convolutional layers to better capture harmonic information (Bittner et al., 2022). The encoder is implemented as a CRNN. To support joint learning with a piano-roll output, we use separate GRU layers: one to predict the piano-roll output and another to produce the encoder output that is forwarded to the decoder.

The decoder follows a Transformer-based architecture. We add positional

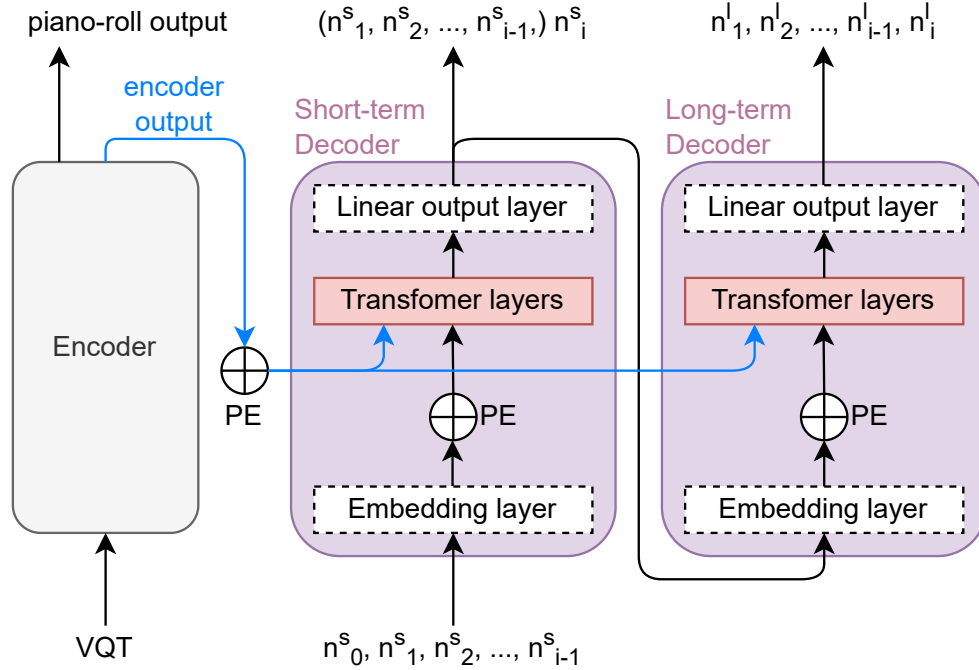


Figure 5.7: Long short-term decoding strategy with a short-term decoder and a long-term decoder. Similar to Figure 5.6, n^s_0 is used to account for the `<sos>` token in practice.

embeddings to the encoder outputs, allowing the decoder to capture positional information. To support our proposed event-based music representation, we modify the embedding and linear output layers in the decoder to handle multiple tokens per time step. As illustrated in Figure 5.6, each time step corresponds to a note in the sequence, and the musical elements associated with that note are decoded in parallel through the decoder’s linear output layer.

Long short-term decoding

We propose a long short-term decoding strategy that decodes the event-based music representation in two stages, processed by firstly the short-term decoder and secondly the long-term decoder. Given \mathbf{h}_{enc} being the encoder output, the

long-short decoding strategy models the following probability distributions:

$$\mathbf{h}'_{\text{enc}} = \text{positional_encoding}(\mathbf{h}_{\text{enc}}) , \quad (5.6)$$

$$D^{\text{short}} := P(\mathbf{n}_i^{\text{s}} | \mathbf{h}'_{\text{enc}}, \mathbf{n}_1^{\text{s}}, \mathbf{n}_2^{\text{s}}, \dots, \mathbf{n}_{i-1}^{\text{s}}) , \quad (5.7)$$

$$D^{\text{long}} := P(\mathbf{Y}^{\text{l}} | \mathbf{h}'_{\text{enc}}, \mathbf{Y}^{\text{s}}) , \quad (5.8)$$

where D^{short} and D^{long} are the model weights for the short-term decoder and the long-term decoder.

Figure 5.7 provides an overview of the process. This strategy employs the same encoder as the baseline model but separates the decoder into a short-term decoder and a long-term decoder. In the first stage, the short-term decoder autoregressively predicts the short-term musical elements of the note sequence \mathbf{Y}^{s} . In the second stage, the long-term decoder predicts the long-term musical elements of the note sequence \mathbf{Y}^{l} in a single step, as no causal mask is required. We place the short-term decoder first, as it captures fundamental musical elements directly derived from the audio recording, while simultaneously providing information that supports the subsequent long-term decoding.

5.3.4 Experimental setup

In this section, we present the experimental setup, including the dataset, evaluation metrics, comparative studies, and the detailed training and inference configurations. For reproducibility, we offer the source code and pretrained model checkpoints of our experiments at:

<https://github.com/cheriell/a2s-ls-decoding>

Data

In this study, we use the MuseSyn dataset with the same split as in Section 5.2, and refer to Table 5.1 for detailed statistics.

Unlike the previous experiment in this chapter, we adopt a different data processing pipeline. Instead of segmenting music recordings into bars using ground-truth downbeat annotations, we divide them into fixed 10-second segments. This design choice serves two purposes: first, it enables us to evaluate

our method in a more realistic setting where downbeat annotations are unavailable; and second, it allows us to test the algorithm on longer excerpts, since the average bar length in the MuseSyn dataset is roughly 2 seconds, which is considerably shorter.

During training, we randomly sample music segments from the training set and apply data augmentations, including pitch shifting (-4 to $+4$ semitones) and time stretching (with a random ratio between 0.8 and 1.2). For evaluation, music segments are sampled consecutively without augmentation.

Evaluation metrics

We evaluate our transcription system using the MV2H metric (McLeod & Steedman, 2018a) for score-level transcription (see subsection 2.4.2 in the background chapter, where we described it in detail). Following the setup in Section 5.2, we omit the harmony sub-metric since our system does not perform key detection. Accordingly, we redefine the MV2H F-score (F_{MV2H}) as the average of the remaining four sub-metrics: multi-pitch (F_{pitch}), voice separation (F_{voice}), metrical alignment (F_{meter}), and note value prediction (F_{value}). For evaluation, we adopt version 2.0 of the MV2H toolbox, which includes automatic alignment between reference and prediction.

Additionally, we use the conventional F-measure metric for frame-level and note-level transcription, since our system also predicts note onsets and duration. We report the framewise and notewise (with and without offset) F-scores (F_f , F_{on} and F_{onoff}) as we did in our previous experiment in this chapter (Section 5.2).

Comparative studies

To gain a clearer understanding of model performance, we compare several variations of the baseline model (described in subsection 5.3.3) against our proposed model with the long short-term decoding strategy:

Baseline model with an RNN decoder (Baseline-RNN). We replace the Transformer-based decoder of the baseline model with an RNN-based decoder with an attention mechanism (as in Section 5.2). This enables a

comparison with our previous architecture under the new experimental setup (i.e., similar model architecture with revised data processing pipeline and representation). Furthermore, since our proposed output representation encodes more information than a score-only representation (e.g., the LilyPond-based format), we omit onset, duration, and beat division to obtain a strictly score-only output representation (i.e., $\mathbf{n}_i = (p_i, h_i, l_i, m_i, v_i)$).

Baseline model with score-only musical elements (Baseline-Score-Only). We then retain the same score-only output representation as in *Baseline-RNN*, but switch back to the Transformer-based decoder used in the baseline model. This enables us to analyse the effect of decoder architecture (RNN vs. Transformer) on model performance.

Baseline model with beat division in the score (Baseline-ScoreWith-Beat). We further extend the output representation of the *Baseline-ScoreOnly* model by adding beat division, in order to investigate how beat information influences model performance. The resulting note sequence is represented as $\mathbf{Y} = \{(p_i, h_i, l_i, b_i, m_i, v_i)\}_{i=1}^{N^{\text{notes}}}$.

Baseline model with all musical elements (Baseline-AllElms). In this variant, we adopt the baseline architecture while incorporating all musical elements defined in our proposed event-based music representation. Rather than distinguishing between short-term and long-term elements, we treat them as a unified set in the baseline model, yielding the output representation $\mathbf{Y} = \{(p_i, o_i, d_i, h_i, l_i, b_i, m_i, v_i)\}_{i=1}^{N^{\text{notes}}}$.

Proposed model with the long short-term decoding strategy (LS-Decoding). In this model, we adopt the long short-term decoding strategy described in [subsection 5.3.3](#), where the note sequence is generated through a two-stage decoding process: first producing \mathbf{Y}^s , followed by \mathbf{Y}^l , as defined in [Equation 5.4](#) and [Equation 5.5](#).

Model training and inference

In our experiments, the encoder is composed of four convolutional layers and two GRU layers for the encoder output, while a single GRU layer is used for the piano-roll output. In addition to the multi-pitch piano-roll output described in [Section 5.2](#), we also include onset prediction in the piano-roll output; both are trained using a mean squared error loss. The decoder consists of four Transformer layers. For all model variants, we employ a cross-entropy loss to learn the output note sequence. To align with our proposed event-based music representation, we add `<sos>` and `<eos>` tokens to each musical element at the start and end of the note sequence, allowing multiple start and end tokens to indicate the boundaries of the note sequence.

During training, we employ a scheduled sampling strategy to enhance the model’s robustness to prediction errors. Specifically, the probability of feeding the model’s own predictions into the decoder is gradually increased from 0% at the start of training to 50% by epoch 50. During inference, we employ greedy decoding, selecting the most likely token at each step. If any musical element encounters a `<eos>` token, decoding is terminated for the note sequence.

Models are trained on four H100 GPUs with a batch size of 30, a learning rate of 1e-3, and a weight decay of 0.01. A `ReduceLRonPlateau` scheduler reduces the learning rate by a factor of 0.2 after 25 epochs without improvement, and early stopping is applied with a patience of 50 epochs.

5.3.5 Results

We now present and discuss the results of our experiments, summarised in [Table 5.6](#) for score transcription results (MV2H metric) and in [Table 5.7](#) for frame-level and note-level transcription results (F-measures).

For the Transformer-based model variants described in the comparative studies, we report the mean and standard deviation over three runs to account for the effect of randomness in model initialisation, since this is an important factor when performance differences between models are relatively small. Although three runs are insufficient for a precise estimate of the standard deviation, we include it to indicate variability in model performance and to support a more comprehensive analysis. For more detailed results of each

Table 5.6: Score transcription results using MV2H metric on the MuseSyn test set. HRes: The high-resolution piano transcription system proposed by (Kong et al., 2021). RNN*: The method we developed in Section 5.2; RNN-Hierarchical*: The hierarchical method introduced by (Zeng et al., 2024); we use * to indicate music segments are pre-processed into bars using ground truth downbeats.

| Method | F_{pitch} | F_{voice} | F_{meter} | F_{value} | F_{MV2H} |
|-------------------------------|--------------------|--------------------|--------------------|--------------------|-------------------|
| <i>Pipeline-based Methods</i> | | | | | |
| HRes+PM2S (pretrained) | 92.6 | 47.5 | 26.2 | 79.9 | 61.6 |
| HRes+PM2S (retrained) | 92.6 | 54.3 | 42.8 | 79.2 | 67.2 |
| HRes+MuseScore | 79.3 | 52.5 | 59.3 | 81.5 | 68.2 |
| <i>Holistic Methods</i> | | | | | |
| RNN* | 71.1 | 90.8 | 94.9 | 94.4 | 87.8 |
| RNN-Hierarchical* | 81.2 | 90.4 | – | 95.3 | – |
| Baseline-RNN | 62.1 | 68.5 | 60.0 | 79.0 | 67.4 |
| Baseline-ScoreOnly | 90.3±2.2 | 84.5±1.6 | 70.5±2.7 | 87.9±0.7 | 83.3±1.8 |
| Baseline-ScoreWithBeat | 91.7±0.6 | 84.2±1.0 | 73.8±0.7 | 88.3±0.2 | 84.5±0.6 |
| Baseline-AllElms | 93.1±0.6 | 85.6±0.1 | 74.8±1.7 | 89.5±0.4 | 85.8±0.5 |
| LS-Decoding | 90.9±0.6 | 84.3±1.0 | 83.1±2.6 | 88.1±0.1 | 86.6±0.5 |

run, we refer the readers to Section A.2 in the Appendix. In addition to the comparative study results, we also report results from several previous pipeline-based methods and holistic approaches for comparison and discussion.

In the following, we first compare the RNN-based models, then examine the Transformer-based baseline models, and subsequently analyse the impact of the long short-term decoding strategy. Finally, we compare our holistic model against prior pipeline-based methods.

Comparison between RNN-based models

We first compare the *Baseline-RNN* with two previous holistic audio-to-score models: the joint multi-pitch detection and score transcription model intro-

Table 5.7: Frame-level and note-level transcription results on the MuseSyn dataset. RNN*: The method we developed in Section 5.2, where the results are evaluated based on the piano-roll output. On the contrary, in the latter two rows (new results in this experiment), results are evaluated based on the note sequence output from the decoder.

| Method | F_f | F_{on} | F_{onoff} |
|------------------|-----------------|-----------------|-----------------|
| HRes | 78.8 | 92.7 | 58.3 |
| RNN* | 88.0 | 66.7 | 53.6 |
| Baseline-AllElms | 84.0±0.4 | 94.4±0.5 | 76.0±0.8 |
| LS-Decoding | 85.6±0.3 | 95.0±0.5 | 78.3±0.1 |

duced in Section 5.2 (*RNN**), and the hierarchical audio-to-score transcription system proposed in (Zeng et al., 2024) (*RNN-Hierarchical**). Models marked with * are trained and evaluated on music segments pre-processed into bars using ground-truth downbeat annotations. As discussed in subsection 5.3.4, the *Baseline-RNN* shares a similar architecture with *RNN** (a sequence-to-sequence model with a CRNN encoder and an RNN decoder with attention), but differs in its data processing pipeline and representation. Compared to *RNN**, the *Baseline-RNN* shows a clear performance drop, and the gap widens further when compared with *RNN-Hierarchical**. The largest degradation occurs in the metrical alignment sub-metric (F_{meter}), underscoring the greater difficulty of transcribing “unprocessed” segments versus pre-segmented bars. Additionally, RNNs struggle with longer sequences: the *Baseline-RNN* processes 10s segments, whereas the other two models decode bar-level outputs averaging only about 2s in length.²

Baseline models with a Transformer-based decoder

When comparing models with a Transformer-based decoder to the *Baseline-RNN*, we observe that the Transformer-based models achieve substantially better results across all four MV2H sub-metrics, suggesting clear benefits of

²Although *RNN-Hierarchical* takes 5 bars of input audio, its hierarchical decoder is structured such that the note-level RNN decoder processes only a single bar at a time.

using a Transformer for the decoder. We attribute this advantage primarily to the multi-head self-attention mechanism, which allows the model to attend to different positions in the sequence simultaneously, regardless of their distance. This is particularly beneficial for score-level transcription, where musical grammar imposes long-range structural constraints. RNN-based decoders, by contrast, compress past context into a fixed-size hidden state that diminishes progressively over longer sequences, limiting their ability to capture such dependencies. Compared to *RNN** and *RNN-Hierarchical**, the Transformer-based models also achieve significantly better performance in multi-pitch estimation (F_{pitch}) and note-level F-scores (F_{on} and F_{onoff}), underscoring the Transformer’s advantage in capturing note-level transcription. Slightly lower results are observed for frame-level transcription (F_{f}), possibly indicating that frame-level prediction is better suited to a piano-roll representation than to a note-sequence format. For the remaining three sub-metrics (F_{voice} , F_{meter} , and F_{value}), the Transformer-based models still underperform compared to *RNN** and *RNN-Hierarchical**, highlighting the continued difficulty of capturing score-level information without pre-segmenting music into bars.

Among the Transformer-based models, incorporating beat division into the output representation appears to be beneficial. Specifically, the *Baseline-ScoreWithBeat* model achieves higher average scores than the *Baseline-ScoreOnly* model in multi-pitch detection (F_{pitch}), metrical alignment (F_{meter}), and note value detection (F_{value}), while attaining competitive performance in voice separation (F_{voice}). We also note that the results of the *Baseline-ScoreOnly* model exhibit greater variability compared to those of the *Baseline-ScoreWithBeat* model. This suggests that, with appropriate engineering effort (e.g., tuning the random initialisation seed), the two models may have comparable potential.

The best-performing baseline model is *Baseline-AllElms*, which outputs all eight musical elements defined in the proposed music representation. This result suggests that joint learning of performance and score transcription is beneficial not only when incorporating a piano-roll output (as we found in [Section 5.2](#)), but also helpful when enriching the transcribed note sequence with performance information such as note onsets and durations.

Effect of long short-term decoding strategy

Comparing the score transcription results of *LS-Decoding* with those of *Baseline-AllElms* (MV2H results in [Table 5.6](#)), we observe a substantial improvement in metrical alignment, with an increase of more than nine percentage points in F_{meter} . However, multi-pitch detection decreases by around two percentage points, and the other two sub-metrics also drop slightly (by approximately one percentage point each). Overall, the averaged MV2H metric (F_{MV2H}) improves by about one percentage point. These findings suggest that the long short-term decoding strategy yields a more balanced performance across sub-metrics by substantially improving the most challenging one, metrical alignment, which is typically the lowest among existing methods, while sacrificing some performance in the others.

The frame- and note-level results, however, paint a different picture (see [Table 5.7](#)). The F-scores (F_f , F_{on} , and F_{onoff}) of the *LS-Decoding* model consistently surpass those of the *Baseline-AllElms* model. This provides a contrasting perspective on the multi-pitch detection results, since the multi-pitch sub-metric in MV2H (F_{pitch}) should, in principle, be consistent with the onset-only F-measure (F_{on}). The discrepancy is likely attributable to differences in the decoding processes: for score transcription outputs, we used score-related musical elements (i.e., those available in *Baseline-RNN* and *Baseline-ScoreOnly*) to ensure comparability with other baselines, whereas for performance outputs, we decoded the note sequence using transcribed pitch, onset, and duration. Another possible factor is the alignment procedure in the MV2H metric, which aligns reference and transcribed scores. Alignment errors introduced by this process may disproportionately affect certain sub-metrics, especially multi-pitch detection. For these reasons, we regard the frame- and note-level F-measures as a more reliable indicator of multi-pitch detection performance.

The improvement in metrical alignment is closely linked to the long-term musical elements predicted in the second decoding stage, suggesting that the long-term decoder effectively captures meter information, which is an essential aspect of audio-to-score transcription. This also indicates that allowing the long-term decoder to “attend to the future” may further enhance transcription accuracy. The observed boost in multi-pitch detection further suggests that

the short-term decoder learns a more precise temporal transcription. On the other hand, the slightly lower performance in voice separation (F_{voice}) and note value detection (F_{value}) indicates that employing the long short-term decoding strategy can also introduce challenges for certain musical aspects. Nevertheless, the overall improvements in both score transcription and frame- and note-level transcription highlight the potential benefits of adopting the long short-term decoding strategy.

Comparison with pipeline-based methods

We also compare our results with several pipeline-based methods. Specifically, we combine the high-resolution piano transcription model (Kong et al., 2021) with either the PM2S model developed in Chapter 4 (denoted as *HRes+PM2S*) or with MuseScore 3 (MuseScore, 2022) (denoted as *HRes+MuseScore*). For the high-resolution piano transcription model (*HRes*), we use the publicly released version pretrained on the MAESTRO dataset. For PM2S, we evaluate both the pretrained model on the ACPAS dataset and the retrained model on the MuseSyn dataset.³

We first compare the multi-pitch estimation results. For this, we refer to the frame- and note-level F-measures in Table 5.7, as they provide a more reliable estimate than the multi-pitch sub-metric in MV2H. Both *Baseline-AllElms* and *LS-Decoding* outperform the high-resolution piano transcription model. Although the *HRes* model is not trained on the MuseSyn dataset, these results nevertheless suggest that our models achieve competitive performance.

For score-level transcription, we observe a different trend from the results on the ACPAS dataset reported in Chapter 4. The *HRes+PM2S* method does not outperform *HRes+MuseScore*, suggesting that in the context of popular music with relatively constant tempo, PM2S offers no clear advantage over algorithmic approaches by MuseScore. The gap in F_{voice} and F_{meter} between

³During post-processing, we smooth the beat estimations by encouraging a constant tempo derived from the estimated beats. This yields better results for F_{meter} than applying the same post-processing pipeline described in Chapter 4. Using that pipeline without tempo smoothing, the *HRes+PM2S* method introduces numerous tempo fluctuations (a common characteristic of expressive performance) and performs very poorly in metrical alignment, with F_{meter} scores of 2.9 and 4.4 for the pretrained and retrained models, respectively.

the pretrained and retrained PM2S models further indicates that the method is sensitive to the training corpus. Despite this, when comparing our new holistic methods (i.e., *LS-Decoding* and *Baseline-AllElms*) with the pipeline-based approaches, both achieve superior results in voice separation (F_{voice}), metrical alignment (F_{meter}), and note value detection (F_{value}). Nevertheless, it remains uncertain whether these advantages will hold in the context of expressive piano performance, which we hope to investigate in future work.

Transcription output examples

In [Figure 5.8](#) and [Figure 5.9](#), we show two example transcription outputs (score-level and note-level) from the *LS-Decoding* model in comparison with the ground truth annotations. We present the same pieces as in [Figure 5.4](#) so that we can compare our new model with the RNN-based model we described in [Section 5.2](#). Note that we used different musical elements to decode the score-level and note-level transcriptions (long-term musical elements for decoding the score, and short-term onsets and duration for decoding the note-level transcription), as discussed earlier. For more output examples, we invite the readers to generate using our released code and trained model checkpoints.

From the transcription examples, we notice that the *LS-Decoding* model tends to successfully capture the meter information, and presents fewer extra/missing note errors than our previous RNN-based model (especially in the left hand of example 2, “His Theme - Piano”). However, note value detection (for score-level transcription) and note offset detection (for note-level transcription) of the *LS-Decoding* doesn’t look good enough. We notice that the model tends to shorten the note values in both examples, especially for longer notes in the left hand. More broadly, beyond the error types presented in the examples, what remains challenging for the holistic model are the expressive performance techniques. For example, rubato is particularly difficult: our current music representation does not encode expressive timing deviations, and the many-to-one alignment between performance and score under rubato is problematic for automatic alignment algorithms. Syncopation and triplets present similar difficulties, as both require accurate estimation of rhythmic patterns that the model tends to confuse with more regular alternatives. Ped-

Example 1: Far Horizons (Piano Cover)

Calmly

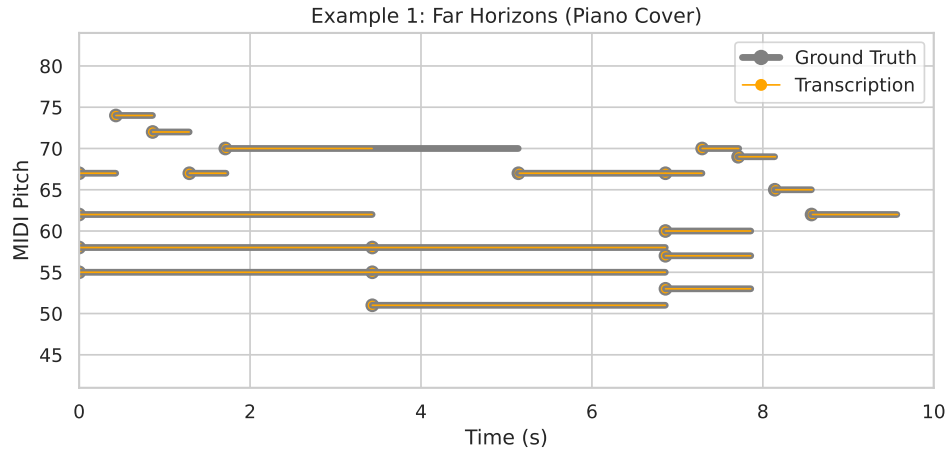
(a)

Example 2: His Theme - Piano

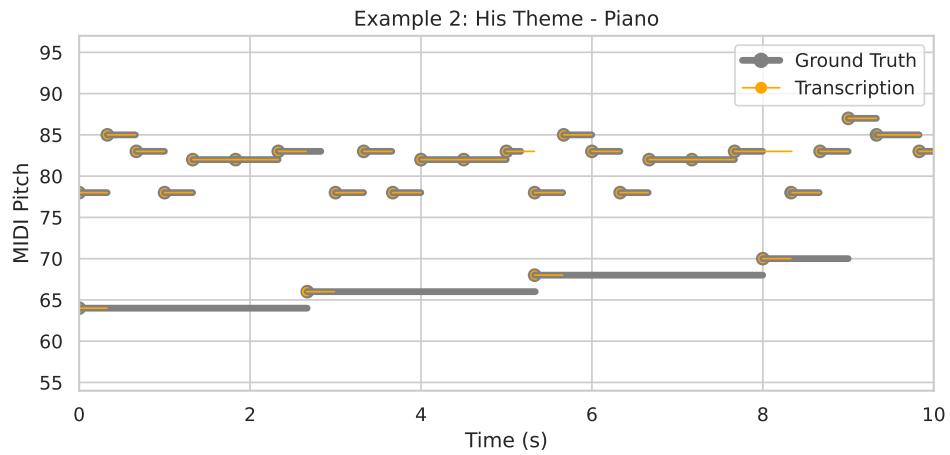
(b)

Figure 5.8: Example score transcriptions by the *LS-Decoding* model. GT: ground truth music score. TR: transcribed music score. We use MuseScore 4 to format the transcribed MIDI scores, where the ground truth key signatures are used for visualisation purposes.

alling introduces a different kind of challenge: it requires inferring note offsets from audio in which the sustain properties of the piano make note boundaries



(a)



(b)

Figure 5.9: Example note-level transcriptions by the *LS-Decoding* model presented in a piano-roll format.

acoustically ambiguous.

5.3.6 Summary of findings

Based on the above discussions, we draw several key conclusions. Transcribing unprocessed music segments is considerably more challenging than working with pre-segmented bars, underscoring the difficulty of “real” transcription when ground-truth downbeats are unavailable. On a modelling level, replacing the RNN decoder with a Transformer yields substantial performance gains, and jointly learning note-level and score-level transcription provides further benefits. In addition, adopting a long short-term decoding strategy not only improves multi-pitch estimation but also enhances metrical alignment. In terms of modelling paradigms, the Transformer-based holistic methods outperform the two pipeline-based approaches considered here. However, this advantage is demonstrated in a controlled setting of synthetic popular music with nearly constant tempo (the MuseSyn dataset). A related concern is that training predominantly on synthetic data may lead the model to produce notational outputs that reflect characteristics specific to synthetic music, which may not transfer to real human performances. Addressing this will require introducing more diverse training datasets, as well as evaluation strategies such as hold-out tests on real performance recordings, to better assess and improve generalisation. Whether such improvements can generalise to more realistic contexts, such as expressive human performances, remains an open question and a promising direction for future research.

5.4 Conclusion

In this chapter, we explored holistic methods for audio-to-score transcription. Our focus was on joint approaches that integrate multi-pitch detection (frame- and note-level transcription) with score-level transcription. Such approaches not only improve overall model performance but also open up opportunities for downstream applications that require access to both note-level and score-level representations.

In the first part of the chapter (Section 5.2), we proposed a model as an initial step toward joint multi-pitch detection and score transcription, based on a CRNN encoder and a sequence-to-sequence decoder with an attention mech-

anism. We introduced a matrix-like score data representation that is more efficient than a LilyPond-based representation, and we adopted the VQT spectrogram as the input feature due to its superior performance on multi-pitch detection compared to four other commonly used time–frequency representations. Training and evaluation on the MuseSyn dataset demonstrated the benefit of multitask learning, as the model jointly produced both piano-roll and score representations. However, this model relied on ground-truth downbeat times to segment recordings into bars, which limits its applicability in real-world scenarios where such information is not available. To address this, the second part of the chapter investigated a more realistic setting by removing the pre-segmentation step and working with longer, unprocessed recordings.

In [Section 5.3](#), we proposed a long short-term decoding strategy for audio-to-score transcription. We designed an output music data representation that encodes both note-level and score-level information, enabling joint learning of event-based note sequences. Our experiments showed that employing a Transformer-based decoder substantially outperforms an RNN-based decoder, and further demonstrated the advantages of multitask learning with the proposed representation. Importantly, the long short-term decoding strategy not only improves note-level transcription but also yields significant gains in metrical alignment, a key aspect of score-level transcription. Overall, this strategy enhanced transcription performance, and the final model outperformed two previous pipeline-based methods.

It should be noted, however, that these results are obtained under the constrained conditions of the MuseSyn dataset, which features synthetic popular music with minimal tempo variation. In future work, we aim to investigate how the proposed holistic methods generalise to more realistic scenarios, such as expressive human performances and classical music recordings.

Chapter 6

Conclusion and Future Work

Having completed the presentation of our experimental work, we now summarise the contributions of this thesis and reflect on key challenges and future opportunities. This chapter begins with a summary of our main contributions (see [Section 6.1](#)), followed by a discussion of future directions closely aligned with this work and personal research interests (see [Section 6.2](#)). We then outline broader challenges and opportunities in the field of AMT (see [Section 6.3](#)), and conclude the chapter with some final reflections (see [Section 6.4](#)).

6.1 Summary of contributions

In this thesis, we worked on the automatic transcription of piano music recordings into a symbolic score format, i.e., audio-to-score piano transcription. This goes a step further than the widely-explored note-level transcription and aims to predict score-level information such as meter and key. We started by collecting two piano datasets for the task, and then explored two types of methods: (1) pipeline-based methods, which first convert a music recording into a note-level transcription and then use another system to transcribe the note-level transcription further into a musical score; and (2) holistic methods, where an end-to-end system is trained to directly transcribe a music recording into a musical score. In the following, we summarise our contributions and findings from each part of this thesis.

We first collected two datasets for audio-to-score piano transcription: the

MuseSyn dataset, which focuses on popular music, and the ACPAS dataset, which focuses on Western classical music. The two datasets also feature different levels of expressive performance. The MuseSyn dataset is a synthesised dataset created by synthesizing music recordings from musical scores, which presents no expressiveness. On the contrary, the ACPAS dataset is a collection of expressive classical piano music. The music recordings are either manually tuned to simulate human performance or recorded from e-piano competitions. We use the two datasets to support our study in this thesis. We also made the two datasets publicly available to support audio-to-score transcription research in the community. For details about the two datasets, we refer to [Chapter 3](#).

We explored pipeline-based methods for audio-to-score piano transcription by adopting an existing transcription system to transcribe music recordings into note sequences of performance MIDIs. We then focused on the latter step of converting note sequences in performed MIDIs into a score format. For this step, we propose to use beat tracking to support rhythm quantisation. We categorise beats into those that are at note onsets, and others that are not at note onsets. We use a CRNN network to track notes that are at beats in a performance MIDI. After that, a dynamic programming algorithm is used to estimate the beats that are not at note onsets. Based on the predicted beats, we can extract meter information from performance MIDIs. We further expand the CRNN network to also extract hand parts, key signatures, and time signatures. And lastly, we combine our proposed method for performance MIDI-to-score conversion with an existing audio-to-performance MIDI piano transcription system into an audio-to-score transcription pipeline. We found this approach effective when evaluated on the ACPAS dataset, featured by rich expressiveness. We presented our exploration with pipeline-based methods in [Chapter 4](#).

For the holistic methods, we first focused on sequence-to-sequence networks based on convolutional networks and recurrent networks with attention mechanisms. We explored the use of multitask learning for audio-to-score transcription by jointly doing multi-pitch estimation and score transcription. We compared different input audio representations for the piano transcription

task and found that among the five commonly used audio representations (i.e., STFT, Mel Spectrogram, CQT, HCQT, and VQT), VQT tends to be the most effective one. We compared a LilyPond score representation with a reshaped score representation based on the LilyPond encoding, where the reshaped score representation has multiple symbols per time step to represent multiple concurrent notes. We found that the reshaped score representation can achieve competitive results to the LilyPond score representation, but at a lower computation cost in both time and memory. In our comparison between single-task learning (i.e., score transcription only) and multitask learning (i.e., joint learning of multi-pitch estimation and score transcription), we found that using multitask learning is beneficial for audio-to-score transcription.

As a further step, we investigated the use of Transformers for holistic audio-to-score piano transcription. To this end, we introduced a long short-term decoding strategy for joint note- and score-level transcription, leveraging an event-based symbolic music representation. Our findings demonstrate that jointly learning note-level and score-level information within a unified note-sequence output provides benefits in the form of multitask learning. Moreover, the proposed long short-term decoding strategy enhances the modelling of both short-term musical features, as reflected in frame- and note-level transcription, and long-term musical features, as reflected in score-level transcription. Finally, we compared our holistic approaches with two existing pipeline-based methods and showed the potential of holistic methods for advancing audio-to-score transcription. For detailed explanations of our studies on holistic methods, we refer to [Chapter 5](#).

6.2 Continuing the trajectory

Building on the studies described in this thesis, we see several promising directions for our future research. In this section, we outline potential avenues for further exploration that are not only closely aligned with our current work but also reflect areas of personal interest. These directions offer both challenges and opportunities as natural extensions of the problems we explored in the thesis, and point out directions for continued research.

6.2.1 Towards instrument agnostic score-level transcription

While this thesis has focused on piano-only transcription, an important next step is to extend this work toward more general, real-world scenarios in which music is performed with a diverse range of instruments. In such cases, developing an instrument-agnostic transcription system becomes highly valuable, as it enables handling situations where the instrument type is unknown or mixed. The trade-offs between instrument-specific and instrument-agnostic transcription systems have been widely discussed in the literature ([Bittner et al., 2022](#); [Weiß & Peeters, 2022](#)). From my perspective, a promising approach is to first build general-purpose systems capable of capturing broad musical patterns, and subsequently fine-tune or adapt them for specific instruments. Such systems offer the advantage of learning more universal musical structures from both audio and symbolic domains by leveraging a wide variety of training corpora.

Encouragingly, recent research has begun to place greater emphasis on transcription systems that go beyond instrument-specific scenarios. These approaches involve training models on a diverse mixture of datasets containing various instruments and musical styles, with the goal of developing systems capable of transcribing music recordings regardless of instrument type. Examples include ([Bittner et al., 2022](#); [Weiß & Peeters, 2022](#); [Cwitkowitz et al., 2024](#)), where models have been proposed to transcribe musical pitch, and in some cases also note onsets, using a piano-roll representation. However, such efforts remain largely focused on frame- and note-level transcription. Limited exploration can be found in score-level transcription in these more general settings. This research direction could also be further explored by applying the methods developed in this thesis to additional datasets beyond piano to facilitate score-level transcription in more diverse, instrument-agnostic scenarios. That said, the most pressing barrier to instrument-agnostic transcription remains the lack of large-scale annotated datasets for instruments beyond piano and for non-Western music. Audio-language models applied to speech recognition illustrate what becomes possible when sufficient data is available; similar architectural advances in music transcription are currently constrained more by this data gap than by model capacity.

Instrument-agnostic transcription models can be trained not only in a fully supervised manner, but also through strategies such as pre-training and fine-tuning in either the audio or symbolic domain. One possibility of moving beyond frame- and note-level transcription to the score level in this context is to leverage a wide range of musical corpora during pre-training in the symbolic domain. This offers benefits since it doesn't require well-aligned audio-score data pairs and enables the use of abundant symbolic music data that would otherwise be difficult to incorporate. In the following section, we delve deeper into this topic by discussing pre-training and domain adaptation approaches for music transcription.

6.2.2 Domain adaptation and pre-training strategies

Following the previous discussion, this subsection points out promising directions of improving transcription through domain adaptation and pre-training strategies.

Domain adaptation approaches have gained increasing attention in recent years, particularly for scenarios where annotated data is scarce, such as for guitar, choral music, or recordings with mismatched acoustic conditions (Riley et al., 2024; McLeod, 2025). They have also been applied in more complex scenarios involving orchestral music and expressive singing, as demonstrated in my subsequent work (Liu & Weiß, 2024, 2025), where we use the “piano” data as the source domain, thanks to large amounts of high-quality annotated data made possible by piano key sensors.

A variety of domain adaptation methods can be explored for music transcription, including, but not limited to, fine-tuning (Riley et al., 2024), adversarial feature alignment (Gharib et al., 2018), and self-supervised pre-training (Li et al., 2024). More broadly, several important questions remain under-explored in this context: (1) Can we go beyond piano and incorporate other instruments, or synthetic audio recordings as source-domain data? (2) Can we leverage large-scale, unlabeled audio and symbolic data to pre-train transcription models in a more generalizable way, for example, through self-supervised learning? (3) Can domain adaptation techniques be extended beyond note-level transcription to support score-level transcription, thereby enabling more

structured and musically meaningful outputs?

While many current research remains focused on supervised and instrument-specific transcription tasks (Gardner et al., 2022; Tamer et al., 2023; Yan & Duan, 2024; Riley et al., 2024), we anticipate a growing shift toward domain adaptation and self-supervised pre-training approaches. This trend is motivated not only by the substantial cost and expertise required to produce high-quality annotations, but also by the rapid advancements in large-scale self-supervised pre-training. For instance, foundation models such as MERT (Li et al., 2024) and MusicFM (Won et al., 2024), as well as pre-trained music large language models in the symbolic domain (Guo & Dixon, 2025), offer promising new avenues. These models can serve not only individual downstream tasks but also provide robust learned representations that enable more complex objectives, such as automatic music transcription. Collectively, these developments lay a strong foundation for the MIR community to pursue more flexible, generalizable, and scalable transcription systems.

As concrete future directions, I am currently investigating the use of pre-trained large language models for audio-to-score transcription. Such models already demonstrate an ability to learn aspects of musical structure and grammar, and I believe this can greatly enhance the generation of musically meaningful scores. Moreover, while domain adaptation holds clear potential for advancing transcription in classical Western music, it is also promising for non-Western musical scenarios, where annotated corpora are often even scarcer and underrepresented. We will elaborate on these aspects later in this chapter.

6.2.3 Linking with other fundamental MIR tasks

Automatic music transcription, as one of the core tasks in Music Information Retrieval (MIR), is closely connected to quite a few other fundamental MIR problems. These include, for example, music synchronization, in the form of transcription-assisted audio-to-score alignment (Simonetta et al., 2021); chord recognition, which can be considered as transcription of chord symbols within musical scores (Müller, 2021); local and global key estimation, as predicting key signature changes as part of audio-to-score transcription (this is slightly

touched in [Chapter 4](#)); beat and downbeat tracking, which highly relates to the transcription of meter in musical performances (we found it helpful for score-level transcription in [Chapter 4](#)); and music generation, where transcription shares underlying representations with music language models or supports tasks such as live automatic music accompaniment with real-time transcription systems ([Kwon et al., 2024](#)).

As a follow-up example, in ([Chiu et al., 2025](#)), my collaborators and I explored cross-modal beat tracking using one of the models developed in [Chapter 4](#). By comparing and combining beat tracking models operating in the audio and symbolic domains, we observed complementarity between the two modalities in both beat and downbeat tracking tasks. Our study also suggests that downbeat tracking from the symbolic domain can be particularly promising when supported by a reliable transcription model, as it enables the system to focus more on musical structure rather than low-level acoustic properties.

6.2.4 Cross-modal and cross-disciplinary directions

Transcription does not need to exist in isolation, it can be extended through synergy with other modalities and disciplines. The integration of symbolic scores and language models in this thesis already establishes a natural bridge between the audio and symbolic domains. Building on this foundation, several cross-modal and cross-disciplinary directions emerge. For example, audio-symbolic-image connections can be explored through the joint use of optical music recognition (OMR) and audio transcription ([Alfaro-Contreras et al., 2023](#)), or through music transcription from video recordings that combine visual and acoustic cues, which is similar to audio-visual speech recognition ([Serdyuk et al., 2022](#); [Cappellazzo et al., 2025](#)). Similarly, audio-text-symbolic integration opens up opportunities for tasks such as simultaneous lyrics and score transcription ([Martinez-Sevilla et al., 2023](#)), or content-based music captioning in the sense of generating textual descriptions from music audio, which closely relates to traditional music tagging tasks ([Manco et al., 2021](#)).

Beyond MIR-specific applications, transcription systems can also contribute to broader fields such as computational humanities. For instance, automatic music transcription systems can support the analysis of cultur-

ally significant music corpora, aiding research in musicology, cultural heritage preservation, and historical music performance studies (Weiß & Müller, 2024). These interdisciplinary directions highlight the potential of transcription models not only as standalone tools, but also as bridges across diverse areas, including humanities and technology.

6.3 Further challenges and opportunities

While automatic music transcription (AMT) remains a highly active area within the field of music information retrieval, the performance of current systems still falls short of expectations, particularly in score-level transcription. Significant challenges persist, but so do opportunities for meaningful advancement. In this section, we highlight several broader limitations and open problems in AMT, aiming to shed light on promising directions for future research beyond the scope of this thesis. Particular attention is given to addressing the foundational barriers that currently constrain progress in the field, such as the lack of diverse datasets, limitations in benchmark evaluation metrics, and related systemic issues.

6.3.1 Datasets

The lack of annotated datasets remains a significant barrier to the advancement of AMT systems. Due to the difficulty of collecting and accurately annotating music recordings, available data for most transcription tasks remains limited, especially for non-Western music, less common instruments, and score-level transcription.

Beyond their size, existing AMT datasets also suffer from several limitations. For instance, the temporal precision of annotations in datasets based on real recordings is often suboptimal. This is one of the reasons why many AMT systems adopt relatively large onset and offset tolerances in note tracking evaluations. Moreover, annotations are typically restricted to basic music features such as pitch, onset and offset times, and occasionally velocity. A more complete transcription would require additional information, including rhythm, key, articulation, and expressive performance annotations.

One of the central challenges in constructing large-scale datasets with score-level annotations lies in aligning audio performances with their corresponding scores, and potentially with expressive performance annotations. Recently, semi-automatic approaches that combines automatic alignment with human corrections have become common (Foscarin et al., 2020; Peter et al., 2023; Weiß et al., 2023). However, these methods still require substantial manual effort, limiting the scalability and diversity of usable datasets.

Exploring ways to leverage existing technologies to generate more data is still one of the steps toward developing better transcription systems. For example, synthesising audio from symbolic data using high-quality audio synthesisers can offer efficient solutions to generate large datasets. While such synthetic data does not fully reflect real-world performance characteristics, it can still be valuable for learning general musical representations that transfer across domains. Nonetheless, researchers need to remain aware of the limitations and potential biases introduced by synthetic datasets, especially when training models intended for real-world applications.

6.3.2 Evaluation metrics

Current evaluation metrics for automatic music transcription (AMT) primarily focus on frame-wise and note-wise comparisons, often using piano-roll representations or note event sequences. These benchmarks typically rely on precision, recall, and F1 scores, offering limited insight into the types or perceptual characteristics of transcription errors. However, not all errors carry equal perceptual or musical weight. For instance, extra notes in a dense polyphonic texture may be more disruptive than missing ones; extra off-key notes may be more noticeable than on-key errors; and errors in a melody or predominant voice are generally more salient than those in inner voices.

Moreover, less work has addressed the evaluation of complete transcription systems, particularly in score-level AMT. Although several score-level evaluation metrics have been proposed in recent years (Nakamura et al., 2018; McLeod & Steedman, 2018a; Hiramatsu et al., 2021), researchers often choose metrics based on individual preference. This lack of standardisation hinders the comparability of results across different studies. Establishing a shared

benchmark metric could significantly improve consistency in evaluation and facilitate more meaningful comparisons. In this direction, the MUSTER metric ([Hiramatsu et al., 2021](#)) may represent a promising candidate.

Another critical gap lies in the limited integration of perceptual considerations into standard AMT evaluation. While some earlier works have attempted to categorise error types ([Poliner & Ellis, 2007](#); [Nakamura et al., 2018](#)), they still fall short of modelling human listening perception. An early contribution ([Daniel et al., 2008](#)) introduced a perceptually motivated evaluation for multi-pitch detection, but this approach has not been adopted widely. Moreover, perceptual evaluation metrics tailored for score-level AMT remain largely underexplored. Developing such metrics would offer a more musically meaningful and user-relevant assessment of system performance. A practical approach is to first define evaluation criteria in collaboration with musicians and music theorists, who can clarify what matters most when reading a transcription. The overall approach will be similar to how we perceptually evaluated note-level piano transcription in ([Ycart et al., 2020a](#)). This step establishes a foundation for any subsequent user study, which becomes considerably more informative once the evaluation criteria are clearly specified.

6.3.3 Non-Western music

Most existing AMT methods are designed for Western tonal music, while considerably less attention has been given to music beyond this paradigm, such as folk, traditional and various world music cultures. As a result, current systems often struggle to transcribe non-Western music accurately.

Key differences between Western and non-Western music that influence AMT design include distinct pitch space organisations (e.g., microtonality), the prevalence of heterophony (as opposed to homophony or polyphony typical in Western music), complex rhythmic and metrical structures, alternative tuning and temperament systems, variations in instrument timbre and playing techniques, and differences in expressive performance practices and notation formats.

These differences are even more pronounced in the context of score-level transcription. The assumptions underlying many AMT systems are often

rooted in Western classical music, particularly regarding melody, harmony, and structural organisation. This introduces significant bias when audio-to-score transcription systems are applied to non-Western music. This issue is further amplified by the increasing reliance on sequence modelling techniques, such as music language models or Transformer-based decoders, which tend to reinforce the structural norms present in the data they are trained on.

Finally, the lack of large-scale annotated datasets for non-Western music remains a major barrier. Without such resources, it is hard to develop and evaluate systems that are sensitive to the unique characteristics of diverse musical traditions.

6.3.4 Score-level transcription

Although recent research in AMT has increasingly turned toward complete or score-level transcription, current methods are still not suitable for general-purpose audio-to-score transcription in real-world scenarios. This is especially evident in multi-instrument or instrument-agnostic settings, where even note-level transcription remains highly challenging. As a result, most score-level transcription studies are still confined to single-instrument contexts.

In pipeline-based approaches, a major challenge is the accumulation of errors across the various stages. For example, the systems responsible for converting performance MIDI data into musical scores are typically trained on clean, high-quality performance MIDI, rather than noisy and error-prone MIDI output from real note-level transcription systems. This mismatch can lead to significant degradation in performance when such systems are applied in end-to-end pipelines. Although fine-tuning these models to handle noisy inputs is possible, doing so requires additional effort every time a new or improved note-level transcription model is introduced into the pipeline.

On the other hand, holistic approaches, which aim to produce a score directly from audio, can avoid error accumulation but introduce their own challenges. These methods require models capable of not only identifying performed notes from audio but also understanding musical structure and grammar to generate structured musical scores. Training such models requires large-scale annotated datasets, which are challenging to obtain. Unlike the

note-level case, where datasets like MAESTRO (Hawthorne et al., 2019) exist, comparable resources for score-level transcription are scarce. As discussed in subsection 6.3.1, the creation of such datasets is particularly difficult due to the complexity of aligning audio with rich symbolic representations.

Another challenge in holistic transcription is the preservation of performance information in the output. In this thesis, we have made efforts to retain such information as much as possible, recognising its importance for downstream tasks such as performance analysis. However, most existing work omits these expressive details, limiting the utility of score-level transcription systems in fine-grained musicological or performance-based applications.

6.4 Final remarks

Despite the various challenges discussed above, we are optimistic about the future of automatic music transcription research. Each of these challenges presents not only obstacles, but also opportunities for future progress. In this thesis, we presented our work as a small step toward developing audio-to-score transcription systems for piano music, in the scope of a narrow subproblem within the broader AMT research. From a methodological standpoint, the central lesson is that progress toward real-world generalisation requires not only larger and more diverse datasets, but also a deeper investigation of how to fully exploit model capacity when labelled data is scarce.

There is still much to explore, far beyond what has been discussed in this thesis. While we do not aim to summarise all possible future directions, we hope this thesis offers useful insights and practical guidance for developing more effective audio-to-score transcription systems. We believe that this field will continue to grow, and that the upcoming years will see exciting and impactful research and development.

Bibliography

- María Alfaro-Contreras, Jose J Valero-Mas, José M Iñesta, and Jorge Calvo-Zaragoza. Late multimodal fusion for image and audio music transcription. *Expert Systems with Applications*, 216:119491, 2023.
- María Alfaro-Contreras, Antonio Ríos-Vila, Jose J Valero-Mas, and Jorge Calvo-Zaragoza. A Transformer approach for polyphonic audio-to-score transcription. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pp. 706–710, Seoul, Korea, 2024. doi: 10.1109/ICASSP48485.2024.10447162.
- Parnia Bahar, Christopher Brix, and Hermann Ney. Towards Two-Dimensional Sequence to Sequence Model in Neural Machine Translation. *arXiv preprint*, pp. 3009–3015, 2018. doi: 10.18653/v1/d18-1335.
- Parnia Bahar, Albert Zeyer, Ralf Schluter, and Hermann Ney. On Using 2D Sequence-to-sequence Models for Speech Recognition. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2019-May:5671–5675, 2019. ISSN 15206149. doi: 10.1109/ICASSP.2019.8682155.
- Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–15, 2015.
- Mert Bay, Andreas F. Ehmann, and J. Stephen Downie. Evaluation of multiple-F0 estimation and tracking systems. In *ISMIR, International So-*

- ciety for Music Information Retrieval Conference*, pp. 315–320, 2009. ISBN 9780981353708.
- J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler. A tutorial on onset detection of music signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 13(5):1035–1047, September 2005.
- E. Benetos and A. Holzapfel. Automatic transcription of Turkish makam music. In *14th International Society for Music Information Retrieval Conference*, pp. 355–360, Curitiba, Brazil, November 2013.
- E. Benetos and T. Weyde. An efficient temporally-constrained probabilistic model for multiple-instrument music transcription. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pp. 701–707, Malaga, Spain, October 2015.
- E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434, December 2013. doi: 10.1007/s10844-013-0258-3.
- Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30, 2019. doi: 10.1109/MSP.2018.2869928.
- R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. Bello. Medleydb: A multitrack dataset for annotation-intensive mir research. In *International Society for Music Information retrieval Conference*, pp. 155–160, Taipei, Taiwan, October 2014.
- Rachel Bittner and Juan J. Bosch. Generalised metrics for single-f0 estimation evaluation. In *Proceedings of the 20th International Society of Music Information Retrieval Conference, ISMIR*, pp. 738–745, Delft, Netherlands, November 2019.
- Rachel M. Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan P. Bello. Deep salience representations for F0 tracking in polyphonic music. In *Pro-*

- ceedings of the International Society for Music Information Retrieval Conference*, pp. 63–70, Suzhou, China, 2017. doi: 10.5281/zenodo.1417937.
- Rachel M. Bittner, Brian McFee, and Juan P. Bello. Multitask learning for fundamental frequency estimation in music, 2018. arXiv:1809.00381 [cs.SD].
- Rachel M. Bittner, Juan José Bosch, David Rubinstein, Gabriel Meseguer-Brocal, and Sebastian Ewert. A lightweight instrument-agnostic model for polyphonic note transcription and multipitch estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 781–785, Virtual and Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9746549.
- S. Böck and M. Schedl. Polyphonic piano note transcription with recurrent neural networks. In *IEEE International Conference on Audio, Speech and Signal Processing*, pp. 121–124, Kyoto, Japan, March 2012.
- Sebastian Böck and Matthew E. P. Davies. Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 574–582, Montreal, Canada, 2020.
- Sebastian Böck, Florian Krebs, and Gerhard Widmer. A multi-model approach to beat tracking considering heterogeneous music styles. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 603–608, Taipei, Taiwan, 2014.
- Juan J Bosch, Ricard Marxer, and Emilia Gómez. Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music. *Journal of New Music Research*, 45(2):101–117, 2016.
- N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *29th International Conference on Machine Learning*, Edinburgh, Scotland, UK, 2012.

- Judith C. Brown. Calculation of a constant Q spectral transform. *Journal of the Acoustical Society of America*, 89(1):425–434, 1991. ISSN NA. doi: 10.1121/1.400476.
- Umberto Cappellazzo, Minsu Kim, Honglie Chen, Pingchuan Ma, Stavros Petridis, Daniele Falavigna, Alessio Brutti, and Maja Pantic. Large language models are strong audio-visual speech recognition learners. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025.
- Ralf Gunter Correa Carvalho and Paris Smaragdis. Towards end-to-end polyphonic music transcription: Transforming music audio directly to a score. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 151–155, October 2017.
- William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, Attend and Spell. pp. 1–16, 2015. URL <http://arxiv.org/abs/1508.01211>.
- Sungkyun Chang, Emmanouil Benetos, Holger Kirchhoff, and Simon Dixon. YourMT3+: Multi-instrument music transcription with enhanced Transformer architectures and cross-dataset stem augmentation. In *IEEE International Workshop on Machine Learning for Signal Processing*, London, UK, 2024.
- C. H. Chen. *Handbook of Pattern Recognition and Computer Vision*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 5th edition, 2016. ISBN 9789814656528, 9814656526.
- Ching-Yu Chiu, Meinard Müller, Matthew E. P. Davies, Alvin Wen-Yu Su, and Yi-Hsuan Yang. Local periodicity-based beat tracking for expressive classical piano music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2824–2835, 2023.
- Ching-Yu Chiu, Lele Liu, Christof Weiß, and Meinard Müller. 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, 2025.

- Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end Continuous Speech Recognition using Attention-based Recurrent NN: First Results. pp. 1–10, 2014. URL <http://arxiv.org/abs/1412.1602>.
- Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-Based Models for Speech Recognition. *Advances in Neural Information Processing Systems*, 2015-Janua:577–585, jun 2015. ISSN 10495258. URL <http://arxiv.org/abs/1506.07503>.
- Andrea Cogliati. *Toward a human-centric automatic piano music transcription system*. PhD thesis, University of Rochester, 2018.
- Andrea Cogliati and Zhiyao Duan. A metric for music notation transcription accuracy. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 407–413, 2017.
- Andrea Cogliati, David Temperley, and Zhiyao Duan. Transcribing human piano performances into music notation. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 758–764, New York, USA, 2016.
- Andrea Cogliati, Zhiyao Duan, and Brendt Wohlberg. Piano transcription with convolutional sparse lateral inhibition. *IEEE Signal Processing Letters*, 24(4):392–396, 2017.
- Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT Press, 2022.
- CPMDatabase. Classical Piano-Midi Database. <http://www.piano-midi.de>, Accessed on 18 Aug, 2021, 2021.
- Frank Cwitkowitz, Kin Wai Cheuk, Woosung Choi, Marco A Martínez-Ramírez, Keisuke Toyama, Wei-Hsiang Liao, and Yuki Mitsufuji. Timbre-trap: A low-resource framework for instrument-agnostic music transcription. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1291–1295, 2024.

- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pp. 2978–2988, 2019. doi: 10.18653/v1/p19-1285.
- Adrien Daniel, Valentin Emiya, and Bertrand David. Perceptually-based evaluation of the errors usually made when automatically transcribing music. In *International Society for Music Information Retrieval Conference, ISMIR*, pp. 550–555, 2008.
- Matthew E. P. Davies, Norberto Degara, and Mark D Plumbly. Evaluation methods for musical audio beat tracking algorithms. *Centre for Digital Music, Queen Mary University of London, Technical Report, C4DM-TR-09-06*, 2009.
- Peter Desain and Henkjan Honing. The quantization of musical time: A connectionist approach. *Computer Music Journal*, 13(3):56–66, 1989. doi: 10.2307/3680012.
- Z. Duan, B. Pardo, and C. Zhang. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):2121–2133, November 2010.
- Zhiyao Duan, Jinyu Han, and Bryan Pardo. Multi-pitch streaming of harmonic sound mixtures. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):138–150, January 2014.
- Valentin Emiya, Roland Badeau, and Bertrand David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6): 1643–1654, 2010. doi: 10.1109/TASL.2009.2038819.
- Finale. Finale: Music notation software that lets you create your way. <https://www.finalemusic.com>, Accessed on 30 Mar, 2022, 2022.
- Francesco Foscarin, Andrew McLeod, Philippe Rigaux, Florent Jacquemard, and Masahiko Sakai. ASAP: A dataset of aligned scores and performances

- for piano transcription. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 534–541, Montreal, Canada, 2020.
- Josh Gardner, Ian Simon, Ethan Manilow, Curtis Hawthorne, and Jesse H. Engel. MT3: Multi-task multitrack music transcription. In *Proceedings of the International Conference on Learning Representations*, Online, 2022.
- Shayan Gharib, Konstantinos Drossos, Emre Cakir, Dmitriy Serdyuk, and Tuomas Virtanen. Unsupervised adversarial domain adaptation for acoustic scene classification. In *Workshop on Detection and Classification of Acoustic Scenes and Events*, pp. 138–142, Surrey, UK, 2018.
- Yoav Goldberg. *Neural Network Methods for Natural Language Processing*. Morgan & Claypool, 2017. ISBN 9781627052986.
- E. Gòmez and J. Bonada. Towards computer-assisted flamenco transcription: An experimental comparison of automatic transcription algorithms as applied to a cappella singing. *Computer Music Journal*, 37(2):73–90, June 2013. doi: 10.1162/COMJ_a.00180.
- Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, 2016. URL <http://www.deeplearningbook.org>.
- M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: music genre database and musical instrument sound database. In *International Conference on Music Information Retrieval*, Baltimore, USA, October 2003.
- Alex Graves. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*, 2012.
- Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. *ACM International Conference Proceeding Series*, 148:369–376, 2006. doi: 10.1145/1143844.1143891.
- Zixun Guo and Simon Dixon. Moonbeam: A MIDI foundation model using both absolute and relative music attributes. *arXiv preprint arXiv:2505.15559*, 2025. URL <https://arxiv.org/abs/2505.15559>.

- William M. Hartmann. Pitch, periodicity, and auditory organization. *The Journal of the Acoustical Society of America*, 100(6):3491–3502, 1996. doi: 10.1121/1.417248.
- Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse H. Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 50–57, Paris, France, 2018.
- Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *Proceedings of the International Conference on Learning Representations*, 2019.
- Curtis Hawthorne, Ian Simon, Rigel Swavely, Ethan Manilow, and Jesse H. Engel. Sequence-to-sequence piano transcription with Transformers. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 246–253, Online, 2021.
- Yuki Hiramatsu, Eita Nakamura, and Kazuyoshi Yoshii. Joint estimation of note values and voices for audio-to-score piano transcription. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 278–284, Online, 2021.
- Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. Compound Word Transformer: Learning to Compose Full-Song Music over Dynamic Directed Hypergraphs. 2021. URL <http://arxiv.org/abs/2101.02402>.
- Yu Siang Huang and Yi Hsuan Yang. Pop music Transformer: Beat-based modeling and generation of expressive pop piano compositions. *MM 2020 - Proceedings of the 28th ACM International Conference on Multimedia*, pp. 1180–1188, 2020. doi: 10.1145/3394171.3413671.
- E. J. Humphrey, S. Durand, and B. McFee. OpenMIC-2018: An open dataset for multiple instrument recognition. In *19th International Society for Music*

- Information Retrieval Conference*, pp. 438–444, Paris, France, September 2018.
- D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Pearson, 2nd edition, 2008.
- Rainer Kelz and Gerhard Widmer. An experimental analysis of the entanglement problem in neural-network-based music transcription systems. In *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*, Jun 2017.
- Rainer Kelz and Gerhard Widmer. Towards Interpretable Polyphonic Transcription with Invertible Neural Networks. In *ISMIR*, 2019. URL <http://arxiv.org/abs/1909.01622>.
- Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. On the potential of simple framewise approaches to piano transcription. In *Proc. International Society for Music Information Retrieval Conference*, pp. 475–481, 2016.
- Rainer Kelz, Sebastian Böck, and Gerhard Widmer. Multitask learning for polyphonic piano transcription, a case study. In *International Workshop on Multilayer Music Representation and Processing (MMRP)*, pp. 85–91, January 2019a. doi: 10.1109/MMRP.2019.8665372.
- Rainer Kelz, Sebastian Böck, and Gerhard Widmer. Deep polyphonic ADSR piano note transcription. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 246–250, Brighton, UK, 2019b. doi: 10.1109/ICASSP.2019.8683582.
- Jong Wook Kim and Juan Bello. Adversarial learning for improved onsets and frames music transcription. In *International Society for Music Information Retrieval Conference*, pp. 670–677, Delft, The Netherlands, November 2019.
- A. Klapuri and M. Davy (eds.). *Signal Processing Methods for Music Transcription*. Springer-Verlag, New York, 2006.

- Anssi Klapuri and Tuomas Virtanen. Automatic Music Transcription. *Computer Music Journal*, 1(4):24–31, 1977. ISSN 0913-5693. doi: 10.1007/978-0-387-30441-0_20.
- Qiuqiang Kong, Bochen Li, Xuchen Song, Yuan Wan, and Yuxuan Wang. High-resolution piano transcription with pedals by regressing onset and offset times. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3707–3717, 2021. doi: 10.1109/TASLP.2021.3121991.
- Florian Krebs, Sebastian Böck, and Gerhard Widmer. An efficient state-space model for joint tempo and meter tracking. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 72–78, Málaga, Spain, 2015.
- Bernd Krueger. Classical Piano-Midi Database, 2018. URL <http://www.piano-midi.de>. URL: <http://www.piano-midi.de> [Accessed: Aug 28, 2021].
- Taegyun Kwon, Dasaem Jeong, and Juhan Nam. Towards efficient and real-time piano transcription using neural autoregressive models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.
- Yizhi Li, Ruibin Yuan, Ge Zhang, Yinghao Ma, Xingran Chen, Hanzhi Yin, Chenghao Xiao, Chenghua Lin, Anton Ragni, Emmanouil Benetos, Norbert Gyenge, Roger B. Dannenberg, Ruibo Liu, Wenhua Chen, Gus Xia, Yemin Shi, Wenhao Huang, Zili Wang, Yike Guo, and Jie Fu. MERT: acoustic music understanding model with large-scale self-supervised training. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.
- Lele Liu and Emmanouil Benetos. From audio to music notation. *Handbook of Artificial Intelligence for Music: Foundations, Advanced Approaches, and Developments for Creativity*, pp. 693–714, 2021.
- Lele Liu and Christof Weiß. Utilizing cross-version consistency for domain adaptation: A case study on music audio. In *The Second Tiny Papers*

- Track at International Conference on Learning Representations*, Vienna, Austria, 2024.
- Lele Liu and Christof Weiß. Unsupervised domain adaptation for music transcription: Exploiting cross-version consistency. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025.
- Lele Liu, Veronica Morfi, and Emmanouil Benetos. ACPAS: A Dataset of Aligned Classical Piano Audio and Scores for Audio-to-Score Transcription. In *Extended Abstract for the Late-Breaking Demo Session of the ISMIR Conference*, 2021a.
- Lele Liu, Veronica Morfi, and Emmanouil Benetos. Joint Multi-Pitch Detection and Score Transcription for Polyphonic Piano Music. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 281–285, June 2021b. doi: 10.1109/ICASSP39728.2021.9413601. ISSN: 2379-190X.
- Lele Liu, Qiuqiang Kong, Veronica Morfi, and Emmanouil Benetos. Performance MIDI-to-score conversion by neural beat tracking. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 395–402, Bengaluru, India, 2022.
- Minh Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, (c):1–10, 2016.
- Ilaria Manco, Emmanouil Benetos, Elio Quinton, and György Fazekas. Muscaps: Generating captions for music audio. In *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2021.
- Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le Roux. Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, October 2019.

- Juan C Martinez-Sevilla, Antonio Rios-Vila, Francisco J Castellanos, and Jorge Calvo-Zaragoza. A holistic approach for aligned music and lyrics transcription. In *International conference on document analysis and recognition*, pp. 185–201, 2023.
- Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa/librosa: 0.8.0 (Version 0.8.0). <http://doi.org/10.5281/zenodo.3955228>, 2020.
- Andrew McLeod. Evaluating Non-aligned Musical Score Transcriptions with MV2H. In *Extended Abstract for the Late-Breaking Demo Session of the ISMIR Conference*. arXiv, July 2019. doi: 10.48550/arXiv.1906.00566. URL <http://arxiv.org/abs/1906.00566>. arXiv:1906.00566 [cs, eess].
- Andrew McLeod. No data required: Zero-shot domain adaptation for automatic music transcription. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025.
- Andrew McLeod and Mark Steedman. Evaluating automatic polyphonic music transcription. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 42–49, Paris, France, 2018a.
- Andrew McLeod and Mark Steedman. Meter detection and alignment of MIDI performances. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 113–119, Paris, France, 2018b.
- Andrew McLeod and Kazuyoshi Yoshii. Evaluating non-aligned musical score transcriptions with mv2h. In *Extended abstract for Late-Breaking/Demo in International Society for Music Information Retrieval Conference, ISMIR*, 2019.
- M. McVicar, R. Santos-Rodríguez, Y. Ni, and T. D. Bie. Automatic chord estimation from audio: A review of the state of the art. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(2):556–575, Feb 2014. ISSN 2329-9290. doi: 10.1109/TASLP.2013.2294580.

- MIREX. Music Information Retrieval Evaluation eXchange (MIREX). <http://music-ir.org/mirexwiki/>. (Accessed on 29 April 2020).
- E. Molina, A. M. Barbancho, L. J. Tardòn, and I. Barbancho. Evaluation framework for automatic singing transcription. In *International Symposium on Music Information Retrieval Conference*, pp. 567–572, October 2014.
- Meinard Müller. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer, 2015. ISBN 9783319219448. doi: 10.1007/978-3-319-21945-5.
- Meinard Müller. *Fundamentals of Music Processing – Using Python and Jupyter Notebooks*. Springer, 2nd edition, 2021. ISBN 978-3-030-69807-2. doi: 10.1007/978-3-030-69808-9.
- MuseScore. MuseScore: Free music composition and notation software. <https://musescore.org>, Accessed on 30 Mar, 2022, 2022.
- Eita Nakamura, Kazuyoshi Yoshii, and Simon Dixon. Note value recognition for piano transcription using markov random fields. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(9):1846–1858, 2017a. doi: 10.1109/TASLP.2017.2722103.
- Eita Nakamura, Kazuyoshi Yoshii, and Shigeki Sagayama. Rhythm transcription of polyphonic piano music based on merged-output HMM for multiple voices. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(4):794–806, 2017b. doi: 10.1109/TASLP.2017.2662479.
- Eita Nakamura, Emmanouil Benetos, Kazuyoshi Yoshii, and Simon Dixon. Towards complete polyphonic music transcription: Integrating multi-pitch detection and rhythm quantization. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 101–105, Calgary, AB, Canada, 2018. doi: 10.1109/ICASSP.2018.8461914.
- J. Nam, , J. Ngiam, H. Lee, and M. Slaney. A classification-based polyphonic piano transcription approach using learned feature representations. In *12th International Society for Music Information Retrieval Conference*, pp. 175–180, Miami, Florida, USA, October 2011.

- Han-Wen Nienhuys and Jan Nieuwenhuizen. LilyPond, a system for automated music engraving. In *Proceedings of the XIV Colloquium on Musical Informatics*, Firenze, Italy, 2003.
- Ryo Nishikimi, Eita Nakamura, Satoru Fukayama, Masataka Goto, and Kazuyoshi Yoshii. Automatic singing transcription based on encoder-decoder recurrent neural networks with a weakly-supervised attention mechanism. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2019.
- Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. This time with feeling: learning expressive musical performance. *Neural Computing and Applications*, pp. 1–24, 2018. ISSN 09410643. doi: 10.1007/s00521-018-3758-9.
- Silvan David Peter, Carlos Eduardo Cancino Chacón, Francesco Foscarin, Andrew McLeod, Florian Henkel, Emmanouil Karystinaios, and Gerhard Widmer. Automatic note-level score-to-performance alignments in the ASAP dataset. *Transactions of the International Society for Music Information Retrieval*, 6(1):27–42, 2023. doi: 10.5334/TISMIR.149.
- Martin Piszczalski and Bernard A. Galler. Automatic music transcription. *Computer Music Journal*, 1(4):24–31, 1977.
- G. Poliner and D. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, (8):154–162, January 2007.
- Colin Raffel and Daniel P. W. Ellis. Intuitive analysis, creation and manipulation of midi data with pretty_midi. In *Proceedings of the 15th International Conference on Music Information Retrieval Late Breaking and Demo Papers*, 2014.
- Imad Rahal, Ryan Strelow, Jeremy Iverson, and Katherine Mendel. Separated feature learning for music composition using memory-based neural networks. In *Proceedings of ISCA 30th International Conference*, volume 77, pp. 41–51, 2021.

- Christopher Raphael. Automated rhythm transcription. In *Proceedings of the International Society for Music Information Retrieval Conference*, Bloomington, Indiana, USA, 2001.
- Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre R. S. Marcal, Carlos Guedes, and Jaime S. Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, Oct 2012. doi: 10.1007/s13735-012-0004-6.
- Yi Ren, Jinzheng He, Xu Tan, Tao Qin, Zhou Zhao, and Tie Yan Liu. Pop-MAG: Pop Music Accompaniment Generation. *MM 2020 - Proceedings of the 28th ACM International Conference on Multimedia*, (1):1198–1206, 2020. doi: 10.1145/3394171.3413721.
- Xavier Riley, Drew Edwards, and Simon Dixon. High resolution guitar transcription via domain adaptation. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1051–1055, 2024.
- Miguel A Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. An End-To-End Framework for Audio-To-Score Music Transcription on Monophonic Excerpts. In *ISMIR, International Society for Music Information Retrieval Conference*, pp. 34–41, 2018. URL <https://github.com/cemfi/meico>.
- Miguel A. Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. A holistic approach to polyphonic music transcription with neural networks. In *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pp. 731–737, Delft, Netherlands, 2019.
- Miguel A Román, Antonio Pertusa, and Jorge Calvo-zaragoza. Data representations for audio-to-score monophonic music transcription. *Expert Systems With Applications*, 162:113769, 2020. ISSN 0957-4174. doi: 10.1016/j.eswa.2020.113769. URL <https://doi.org/10.1016/j.eswa.2020.113769>.
- Sebastian Ruder. An overview of multi-task learning in deep neural networks. arXiv eprint 1706.05098, 2017.

- J. Salamon, E. Gomez, D.P.W. Ellis, and G. Richard. Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, March 2014. doi: 10.1109/MSP.2013.2271648.
- Julian Salazar, Katrin Kirchhoff, and Zhiheng Huang. Self-attention Networks for Connectionist Temporal Classification in Speech Recognition. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2019-May:7115–7119, 2019. ISSN 15206149. doi: 10.1109/ICASSP.2019.8682539.
- Daniel Schneider, Nikolaus Korfhage, Markus Mühling, Peter Lüttig, and Bernd Freisleben. Automatic transcription of organ tablature music notation with deep neural networks. *Transactions of the International Society for Music Information Retrieval*, 4(1):14–28, 2021. doi: 10.5334/TISMIR.77.
- Christian Schörkhuber, Anssi Klapuri, Nicki Holighaus, and Monika Dörfler. A Matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution. In *Proceedings of the AES International Conference*, pp. 232–239, 2014. ISBN 9781632662842.
- Dmitriy Serdyuk, Otavio Braga, and Olivier Siohan. Transformer-based video front-ends for audio-visual speech recognition for single and multi-person video. In *Proceedings of Interspeech*, pp. 2833–2837, 2022.
- X. Serra, M. Magas, E. Benetos, M. Chudy, S. Dixon, A. Flexer, E. Gómez, F. Gouyon, P. Herrera, S. Jorda, O. Paytuvi, G. Peeters, J. Schlüter, H. Vinet, and G. Widmer. *Roadmap for Music Information ReSearch*. Creative Commons BY-NC-ND 3.0 license, 2013. ISBN 978-2-9540351-1-6.
- Tao Shen, Jing Jiang, Tianyi Zhou, Shirui Pan, Guodong Long, and Chengqi Zhang. Disan: Directional self-attention network for RnN/CNN-free language understanding. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 5446–5455, 2018.

- Kentaro Shibata, Eita Nakamura, and Kazuyoshi Yoshii. Non-local musical statistics as guides for audio-to-score piano transcription. *Information Sciences*, 566:262–280, 2021. doi: 10.1016/J.INS.2021.03.014.
- S. Sigtia, E. Benetos, and S. Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939, May 2016. doi: 10.1109/TASLP.2016.2533858.
- Federico Simonetta, Stavros Ntalampiras, and Federico Avanzini. Audio-to-score alignment using deep automatic music transcription. In *IEEE 23rd International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6, 2021.
- Paris Smaragdis and Judith C Brown. Non-negative matrix factorization for polyphonic music transcription. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 177–180. IEEE, 2003.
- L. Su and Y.-H. Yang. Escaping from the abyss of manual annotation: New methodology of building polyphonic datasets for automatic music transcription. In *International Symposium on Computer Music Multidisciplinary Research*, June 2015.
- Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 4:3104–3112, 2014. ISSN 10495258.
- Masahiro Suzuki. Score Transformer: Generating musical score from note-level representation. In *Proceedings of the ACM Multimedia Asia*, pp. 31:1–31:7, Gold Coast, Australia, 2021. doi: 10.1145/3469877.3490612.
- Haruto Takeda, Naoki Saito, Tomoshi Otsuki, Mitsuru Nakai, Hiroshi Shimodaira, and Shigeki Sagayama. Hidden Markov model for automatic transcription of MIDI signals. In *IEEE Workshop on Multimedia Signal Processing*, pp. 428–431, St. Thomas, Virgin Islands, USA, 2002. IEEE. doi: 10.1109/MMSP.2002.1203337.

- Nazif Can Tamer, Yigitcan Özer, Meinard Müller, and Xavier Serra. High-resolution violin transcription using weak labels. In *International Society for Music Information Retrieval Conference*, pp. 223–230, 2023.
- Keitaro Tanaka, Takayuki Nakatsuka, Ryo Nishikimi, Kazuyoshi Yoshii, and Shigeo Morishima. Multi-Instrument Music Transcription Based on Deep Spherical Clustering of Spectrograms and Pitchgrams. In *International Society for Music Information Retrieval Conference*, 2020.
- David Temperley. A unified probabilistic model for polyphonic music analysis. *Journal of New Music Research*, 38(1):3–18, 2009. doi: 10.1080/09298210902928495.
- David Temperley and Daniel Sleator. Modeling meter and harmony: A preference-rule approach. *Computer Music Journal*, 23(1):10–27, 1999. doi: 10.1162/014892699559616.
- John Thickstun, Zaid Harchaoui, and Sham M. Kakade. Learning features of music from scratch. In *International Conference on Learning Representations (ICLR)*, 2017.
- Keisuke Toyama, Taketo Akama, Yukara Ikemiya, Yuhta Takida, Wei-Hsiang Liao, and Yuki Mitsufuji. Automatic piano transcription with hierarchical frequency-time Transformer. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 215–222, Milan, Italy, 2023. doi: 10.5281/ZENODO.10265261.
- Karen Ullrich and Eelco van der Wel. Music Transcription With Convolutional Sequence-to-Sequence Models. *International Conference on Learning Representations (rejected)*, pp. 1–9, 2018.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *International Conference on Neural Information Processing Systems*, pp. 5998–6008, Long Beach, CA, USA, 2017.

- Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. Grammar as a foreign language. *Advances in Neural Information Processing Systems*, 2015-Janua:2773–2781, 2015. ISSN 10495258.
- T. Virtanen, M. D. Plumbley, and D. P. W. Ellis (eds.). *Computational Analysis of Sound Scenes and Events*. Springer, 2018. ISBN 978-3-319-63450-0.
- Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. pp. 5797–5808, 2019. doi: 10.18653/v1/p19-1580.
- Jun-You Wang and Jyh-Shing Roger Jang. Training a singing transcription model using connectionist temporal classification loss and cross-entropy loss. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31: 383–396, 2023. doi: 10.1109/TASLP.2022.3224297.
- Qi Wang, Ruohua Zhou, and Yonghong Yan. Polyphonic piano transcription with a note-based music language model. *Applied Sciences*, 8(3), 2018. ISSN 2076-3417. doi: 10.3390/app8030470.
- Christof Weiß and Meinard Müller. From music scores to audio recordings: Deep pitch-class representations for measuring tonal structures. *ACM Journal on Computing and Cultural Heritage*, 17(3):1–19, 2024.
- Christof Weiß and Geoffroy Peeters. Comparing deep models and evaluation strategies for multi-pitch estimation in music recordings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:2814–2827, 2022. doi: 10.1109/TASLP.2022.3200547.
- Christof Weiß, Vlora Arifi-Müller, Michael Krause, Frank Zalkow, Stephanie Klauk, Rainer Kleinertz, and Meinard Müller. Wagner Ring dataset: A complex opera scenario for music processing and computational musicology. *Transactions of the International Society for Music Information Retrieval*, 6(1):135–149, 2023. doi: 10.5334/TISMIR.161.
- Minz Won, Yun-Ning Hung, and Duc Le. A foundation model for music informatics. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1226–1230, 2024.

- C. Wu, C. Dittmar, C. Southall, R. Vogl, G. Widmer, J. Hockman, M. Müller, and A. Lerch. A review of automatic drum transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1457–1483, Sep. 2018. ISSN 2329-9290. doi: 10.1109/TASLP.2018.2830113.
- Yu-Te Wu, Berlin Chen, and Li Su. Multi-instrument automatic music transcription with self-attention-based instance segmentation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2796–2809, 2020. ISSN 2329-9304. doi: 10.1109/TASLP.2020.3030482.
- Qingyang Xi, Rachel M. Bittner, Johan Pauwels, Xuzhou Ye, and Juan Pablo Bello. Guitarset: A dataset for guitar transcription. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 453–460, Paris, France, 2018.
- Yujia Yan and Zhiyao Duan. Scoring time intervals using non-hierarchical transformer for automatic piano transcription. In *International Society for Music Information Retrieval Conference*, 2024.
- Baosong Yang, Longyue Wang, Derek F Wong, Lidia S Chao, and Zhaopeng Tu. Convolutional self-attention networks. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pp. 4040–4045, 2019.
- A. Ycart and E. Benetos. A study on LSTM networks for polyphonic music sequence modelling. In *18th International Society for Music Information Retrieval Conference (ISMIR)*, pp. 421–427, 2017.
- A. Ycart, E. Benetos, and 19th International Society for Music Information Retrieval Conference Late-Breaking Demos Session. A-MAPS: Augmented MAPS Dataset with Rhythm and Key Annotations. In *Extended Abstract for the Late-Breaking Demo Session of the ISMIR Conference*, 2018. URL <https://qmro.qmul.ac.uk/xmlui/handle/123456789/45985>. Accepted: 2018-10-09T09:41:00Z.

- Adrien Ycart, Andrew McLeod, Emmanouil Benetos, and Kazuyoshi Yoshii. Blending Acoustic and Language Model Predictions for Automatic Music Transcription. In *20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.
- Adrien Ycart, Lele Liu, Emmanouil Benetos, and Marcus T. Pearce. Investigating the Perceptual Validity of Evaluation Metrics for Automatic Piano Music Transcription. *Transactions of the International Society for Music Information Retrieval*, 3(1):68–81, 2020a. doi: 10.5334/tismir.57.
- Adrien Ycart, Lele Liu, Emmanouil Benetos, and Marcus T. Pearce. Investigating the perceptual validity of evaluation metrics for automatic piano music transcription. *Transactions of the International Society for Music Information Retrieval*, 3(1):68–81, 2020b. doi: 10.5334/TISMIR.57.
- Dong Yu and Li Deng (eds.). *Automatic Speech Recognition: A Deep Learning Approach*. Springer-Verlag, London, 2015.
- Mingliang Zeng, Xu Tan, Rui Wang, Zeqian Ju, Tao Qin, and Tie-Yan Liu. MusicBERT: Symbolic music understanding with large-scale pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 791–800, 2021.
- Wei Zeng, Xian He, and Ye Wang. End-to-end real-world polyphonic piano audio-to-score transcription with hierarchical decoding. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2024.
- Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *36th International Conference on Machine Learning, ICML 2019*, 2019-June:12744–12753, 2019.
- Jingwei Zhao, Gus Xia, and Ye Wang. Beat Transformer: Demixed beat and downbeat tracking with dilated self-attention. In *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 169–177, Bengaluru, India, 2022.
- Lixia Zhao, Lingyan Zhu, Shuyan Zhao, and Xinxin Ma. Sequestration and bioavailability of perfluoroalkyl acids (PFAAs) in soils: Implications for

their underestimated risk. *Science of the Total Environment*, 572:169–176, 2016. ISSN 18791026. doi: 10.1016/j.scitotenv.2016.07.196.

Chapter A

Appendices

A.1 Detailed results for time–frequency representation comparison

In the experiments described in [Section 5.2](#), we compare different input time–frequency representations in the form of audio spectrograms for multi-pitch detection, including Short Time Fourier Transform (STFT), Mel Spectrogram, Constant-Q Transform (CQT), Harmonic Constant-Q Transform (HCQT) and Variable-Q Transform (VQT). Please refer to [Section 5.2](#) for model parameters. The model performance is evaluated using the benchmark frame-level and note-level F-measures for automatic music transcription in MIREX ([Bay et al., 2009](#)).

Here, we provide additional results on the model results with different parameters of the time–frequency representations. The results are in [Table A.1](#). Results show what the overall best input representation is the VQT spectrogram with 60×8 frequency bins and a gamma value of 20. We discover some trends in how the parameters influence model performance. For example, a window length of 2048 outperforms 1024 for STFT and Mel Spectrogram. Larger number of frequency bins tend to result in higher performance for Mel Spectrogram, CQT, HCQT and VQT.

A.2 Detailed results of repeat runs for Transformer-based models

Here we provide the detailed results for the repeat runs of the Transformer-based models described in [Section 5.3](#). The score transcription results are in [Table A.2](#). Frame- and note-level transcription results are in [Table A.3](#).

Table A.1: F-measure of piano-roll prediction on different input representations and parameters. N_w : window length, N_m : number of mel bands, N_b : number of bins per octave, N_o : number of octaves, N_h : number of harmonics.

| Input reps. | P_f | R_f | F_f | P_{on} | R_{on} | F_{on} | P_{onoff} | R_{onoff} | F_{onoff} | | |
|------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| <i>STFT</i> | | | | | | | | | | | |
| N_w | | | | | | | | | | | |
| 1024 | 90.09 | 87.42 | 87.73 | 86.20 | 76.64 | 78.74 | 61.95 | 56.83 | 58.09 | | |
| 2048 | 90.36 | 90.36 | 89.46 | 89.51 | 77.76 | 80.99 | 66.24 | 59.86 | 61.73 | | |
| <i>Mel Spectrogram</i> | | | | | | | | | | | |
| N_w | N_m | | | | | | | | | | |
| 1024 | 192 | 88.41 | 87.32 | 86.88 | 83.13 | 75.81 | 76.89 | 59.87 | 56.51 | 56.89 | |
| 2048 | 128 | 90.90 | 87.32 | 88.18 | 85.94 | 78.32 | 79.73 | 62.54 | 58.85 | 59.53 | |
| 2048 | 192 | 90.77 | 85.72 | 87.20 | 85.70 | 76.47 | 78.49 | 62.71 | 58.02 | 59.10 | |
| 2048 | 256 | 91.60 | 88.19 | 88.98 | 90.48 | 78.65 | 82.12 | 67.38 | 60.80 | 62.95 | |
| <i>CQT</i> | | | | | | | | | | | |
| N_b | N_o | | | | | | | | | | |
| 12 | 7 | 90.73 | 88.81 | 88.91 | 88.41 | 77.96 | 80.76 | 65.57 | 60.11 | 61.68 | |
| 12 | 8 | 90.92 | 88.50 | 88.69 | 89.48 | 78.00 | 81.33 | 66.02 | 59.70 | 61.69 | |
| 24 | 8 | 93.02 | 90.35 | 90.91 | 92.67 | 80.72 | 84.39 | 70.79 | 63.86 | 66.15 | |
| 36 | 8 | 93.39 | 90.67 | 91.27 | 92.99 | 80.91 | 84.71 | 70.86 | 63.94 | 66.29 | |
| 48 | 8 | 93.89 | 90.44 | 91.44 | 93.45 | 81.31 | 85.14 | 72.20 | 65.15 | 67.56 | |
| 60 | 8 | 93.79 | 91.21 | 91.85 | 93.25 | 81.96 | 85.43 | 71.82 | 65.18 | 67.40 | |
| <i>HCQT</i> | | | | | | | | | | | |
| N_b | N_o | N_h | | | | | | | | | |
| 36 | 5 | 4 | 91.47 | 89.43 | 89.76 | 91.43 | 79.79 | 83.19 | 67.76 | 61.55 | 63.52 |
| 60 | 5 | 4 | 91.85 | 88.96 | 89.55 | 90.88 | 78.88 | 82.48 | 66.79 | 60.47 | 62.57 |
| 60 | 6 | 4 | 92.17 | 89.88 | 90.24 | 90.89 | 79.55 | 82.74 | 67.07 | 61.11 | 62.93 |
| 60 | 6 | 5 | 92.97 | 90.49 | 90.95 | 91.81 | 81.06 | 84.14 | 69.27 | 63.53 | 65.31 |
| 60 | 6 | 6 | 91.60 | 89.03 | 89.43 | 88.67 | 79.48 | 81.68 | 64.49 | 59.93 | 61.09 |
| <i>VQT</i> | | | | | | | | | | | |
| N_b | N_o | γ | | | | | | | | | |
| 36 | 7 | 10 | 92.87 | 90.64 | 91.01 | 92.54 | 80.60 | 84.24 | 70.75 | 63.87 | 66.17 |
| 60 | 7 | 10 | 93.22 | 90.69 | 91.14 | 92.33 | 80.37 | 83.94 | 71.41 | 64.49 | 66.71 |
| 60 | 8 | 10 | 94.00 | 90.93 | 91.75 | 93.94 | 82.08 | 85.76 | 72.63 | 65.53 | 67.94 |
| 60 | 8 | 20 | 94.22 | 91.04 | 91.93 | 93.81 | 82.11 | 85.70 | 73.07 | 66.25 | 68.53 |
| 60 | 8 | 30 | 94.15 | 91.01 | 91.85 | 93.91 | 82.03 | 85.70 | 73.00 | 66.05 | 68.37 |

Table A.2: Score transcription results using MV2H metric on the MuseSyn test set.

| Method | Run | F_{pitch} | F_{voice} | F_{meter} | F_{value} | F_{MV2H} |
|------------------------|-----|--------------------|--------------------|--------------------|--------------------|-------------------|
| Baseline-ScoreOnly | 1 | 89.82 | 83.94 | 69.18 | 87.75 | 82.67 |
| | 2 | 88.45 | 83.27 | 68.77 | 87.25 | 81.93 |
| | 3 | 92.67 | 86.36 | 73.61 | 88.72 | 85.34 |
| Baseline-ScoreWithBeat | 1 | 91.63 | 85.21 | 74.45 | 88.39 | 84.92 |
| | 2 | 91.07 | 83.15 | 73.02 | 88.07 | 83.83 |
| | 3 | 92.31 | 84.30 | 73.82 | 88.51 | 84.74 |
| Baseline-AllElms | 1 | 92.93 | 85.71 | 72.96 | 89.25 | 85.21 |
| | 2 | 93.84 | 85.72 | 75.17 | 89.90 | 86.16 |
| | 3 | 92.67 | 85.46 | 76.36 | 89.33 | 85.96 |
| LS-Decoding | 1 | 90.25 | 85.41 | 80.13 | 88.27 | 86.02 |
| | 2 | 91.17 | 83.97 | 84.13 | 88.08 | 86.84 |
| | 3 | 91.32 | 83.47 | 84.91 | 88.09 | 86.95 |

Table A.3: Frame- and note-level transcription results on the MuseSyn test set.

| Method | Run | P_f | R_f | F_f | P_{on} | R_{on} | F_{on} | P_{onoff} | R_{onoff} | F_{onoff} |
|------------------|-----|-------|-------|-------|-----------------|-----------------|-----------------|--------------------|--------------------|--------------------|
| Baseline-AllElms | 1 | 89.14 | 81.56 | 84.21 | 95.38 | 93.23 | 94.17 | 77.07 | 75.41 | 76.15 |
| | 2 | 90.05 | 81.34 | 84.39 | 96.06 | 94.12 | 94.98 | 77.53 | 76.08 | 76.72 |
| | 3 | 88.17 | 81.15 | 83.55 | 94.31 | 93.88 | 94.00 | 75.39 | 75.13 | 75.19 |
| LS-Decoding | 1 | 89.91 | 82.80 | 85.26 | 96.06 | 93.21 | 94.44 | 79.53 | 77.33 | 78.29 |
| | 2 | 88.65 | 84.92 | 85.79 | 95.57 | 95.42 | 95.34 | 78.59 | 78.37 | 78.42 |
| | 3 | 88.26 | 85.16 | 85.71 | 95.83 | 94.91 | 95.24 | 78.64 | 77.99 | 78.26 |