

UNIVERSITÉ DE LILLE
École graduée MADIS-631
Mathématiques, sciences du numérique et de leurs interactions

Modeling Symbolic Music with Natural Language Processing Approaches

*Analyzing and Experimenting with
Multiple Levels of Representations*

**Modélisation de la Musique Symbolique par des
Approches de Traitement Automatique du Langage Naturel**

*Analyse et Expérimentations sur
Différents Niveaux de Représentation*

DINH-VIET-TOAN LE

PHD THESIS
COMPUTER SCIENCE

Defended on November 3, 2025 in front of the jury composed of:

M.	XAVIER HINAUT	Chargé de recherche Inria <i>Inria, Université de Bordeaux</i>	Reviewer
Ms.	CHENG-ZHI ANNA HUANG	Assistant Professor <i>Massachusetts Institute of Technology</i>	Reviewer
M.	EMMANOUIL BENETOS	Director of Research & Reader <i>Queen Mary University of London</i>	Examiner
Ms.	CHLOÉ BRAUD	Chargée de recherche CNRS <i>Inst. de Recherche en Informatique de Toulouse</i>	Examiner
M.	MARIUS BILASCO	Professeur des Universités <i>CRISTAL, Université de Lille</i>	Examiner
M.	PATRICK BAS	Directeur de recherche CNRS <i>CRISTAL, Université de Lille</i>	Examiner Jury president
M.	LOUIS BIGO	Professeur des Universités <i>LaBRI, Université de Bordeaux</i>	Thesis co-director
M.	MARC TOMMASI	Professeur des Universités <i>CRISTAL, Université de Lille</i>	Thesis co-director
Ms.	MIKAELA KELLER	Maîtresse de conférence <i>CRISTAL, Université de Lille</i>	Co-supervisor

Abstract

Keywords – Music Information Retrieval, Symbolic Music, Natural Language Processing, Machine Learning.

Music is often described as a *language* because of its similarities to natural language. These include their respective representations through symbolic music notation and textual form. Therefore, the field of Music Information Retrieval (MIR) has often borrowed several tools from the Natural Language Processing (NLP) field to adapt them to process symbolic music data. In particular, this phenomenon has been increasingly popular with the breakthrough of Transformer models in the NLP field.

This thesis first provides a structured overview of adaptations of NLP methods developed in the MIR field for symbolic music processing. They are presented along three axes, each addressing the use of diverse representations of symbolic music at different levels. Symbolic music *represented as sequential data* has led to the development of several tokenization strategies, which we propose to organize within a unified taxonomy. These representations are subsequently processed through *models*, such as recurrent or attention-based architectures initially developed for text data, giving rise to multiple adaptations for symbolic music processing. Finally, these abstract representations are used to perform *tasks*, where both parallels and distinctive characteristics emerge between MIR and NLP.

These aspects then structure the three technical contributions of this thesis. First, we study the expressiveness of sequential *representations* of music through the development of interval-based tokenization strategies, and the analysis of a subword tokenization strategy, Byte-Pair Encoding, applied to symbolic music tokens. We then propose a framework for *model* explainability which leads to the analysis of the attention mechanism of a Transformer-based model trained for functional harmony analysis. Finally, we develop a model adapted from NLP tools for a *task* of re-orchestration, framed as a case of multi-track music generation.

Ultimately, this thesis defends that NLP methods first remains a toolbox from which MIR studies can take some tools from. Beyond the analogies between music and natural language, the main motivation guiding a MIR study should be *musical* questions.

Résumé

Mots-clés – Extraction automatique d'information musicale, Représentations symboliques de la musique, Traitement automatique du langage naturel, Apprentissage automatique.

La musique est souvent comparée à un *langage*. Cette comparaison est notamment dûe au fait que musique et langage naturel partagent de nombreuses similarités. Parmi celles-ci figurent leurs représentations respectives à travers la notation musicale symbolique – ou partition musicale – et la forme écrite textuelle du langage. Ainsi, le domaine de la recherche d'information musicale *MIR* a fréquemment emprunté des outils provenant du domaine du Traitement automatique du langage naturel (TALN) afin de les adapter au traitement de données musicales symboliques. Ce phénomène s'est particulièrement intensifié avec l'essor des modèles de type Transformer dans le domaine du TALN.

Cette thèse propose tout d'abord une synthèse structurée des adaptations des méthodes de TALN développées dans le champ du MIR pour le traitement de la musique symbolique. Elles sont présentées selon trois axes, chacun portant sur l'utilisation de différentes représentations de la musique symbolique à divers niveaux. La musique symbolique *représentée comme des données séquentielles* a conduit au développement de plusieurs stratégies de tokenization, que nous proposons d'organiser au sein d'une taxonomie unifiée. Ces représentations sont ensuite traitées par des *modèles*, tels que les architectures basées sur des mécanismes de récurrence ou d'attention. Celles-ci, initialement conçues pour les données textuelles, ont donné lieu à de multiples adaptations pour le traitement de la musique symbolique. Enfin, ces représentations abstraites sont utilisées pour accomplir des *tâches*, où émergent à la fois des parallèles et des spécificités distinctives entre MIR et TALN.

Ces aspects structurent ensuite les trois contributions techniques de cette thèse. Dans un premier temps, nous étudions l'expressivité des *représentations séquentielles* de la musique à travers le développement de stratégies de tokenization basées sur les intervalles musicaux, ainsi que l'analyse d'une stratégie de tokenization en sous-mots, le *Byte-Pair Encoding*, appliqué aux tokens musicaux symboliques. Nous proposons ensuite un cadre pour l'explicabilité de *modèles*, qui est utilisé pour l'analyse du mécanisme d'attention d'un modèle basé sur Transformeur, entraîné sur une tâche d'analyse d'harmonie fonctionnelle. Enfin, nous développons un modèle adapté des outils du TALN pour une tâche de ré-orchestration, considérée comme un cas de génération automatique de musique multi-instrumentale.

Par ces contributions, cette thèse soutient que les méthodes de TALN restent avant tout une boîte à outils dans laquelle le MIR peut s'inspirer. Malgré les analogies entre ces deux domaines, la principale motivation guidant une étude en MIR devrait avant tout être d'ordre *musical*.

Acknowledgements

Remerciements

Thanks!

謝謝!

Merci !

Contents

Abstract	iii
Résumé	v
Acknowledgements	vii
Table of contents	xii
List of figures	xv
List of tables	xvii
Glossary	xix
1 Introduction	1
2 On parallelisms between music and natural language	7
2.1 Music and language as communication means	8
2.1.1 Functions of music and natural language	8
2.1.2 Cognitive responses to music and natural language	9
2.2 Symbolic music and text as structured data	10
2.2.1 Hierarchical representations	10
2.2.2 Musical and textual <i>grammatical</i> rules?	13

I	A structured overview of NLP methods for symbolic MIR	17
3	MIR and NLP tasks: similarities and specificities	23
3.1	Tasks learned from labeled data	24
3.1.1	Sequence classification and regression	24
3.1.2	Token classification and regression	25
3.2	Tasks relying on unlabeled data	27
3.2.1	Unsupervised clustering and segmentation	28
3.2.2	A prominence of generative tasks in MIR an NLP	29
3.2.3	Unlabeled text and symbolic music datasets	31
4	Representations of symbolic music as sequences	35
4.1	Tokenization strategies	36
4.1.1	Time-slice-based tokenization	37
4.1.2	Event-based tokenization	38
4.2	Comparing tokenization strategies	47
4.3	Embedding music tokens for model processing	48
5	NLP-based models for symbolic music processing	53
5.1	Shallow models	54
5.2	Sequential neural models	55
5.2.1	Neural networks	55
5.2.2	Recurrent models	57
5.3	Attention-based models	61
5.3.1	Training paradigms	64
5.3.2	Model architecture	67
5.3.3	Adapting attention models to symbolic music	71

II	Technical contributions	75
6	Improving music sequential representations' expressiveness	79
6.1	Interval-based tokenization for pitch representation	80
6.1.1	Intervalization	81
6.1.2	Evaluating intervalization on downstream tasks	83
6.1.3	Towards improving token alphabet expressiveness	90
6.2	Analyzing byte-pair encoding for monophonic and polyphonic music	92
6.2.1	Analyzing music byte-pair encoding	92
6.2.2	Evaluating BPE on musical phrase segmentation	97
6.2.3	Towards improving token grouping expressiveness	100
6.3	Tokenization expressiveness or model power?	101
7	Exploring attention-based model inner mechanisms	103
7.1	Model explainability through mechanistic interpretability	104
7.2	Models for functional harmony analysis	105
7.2.1	Functional harmony analysis	105
7.2.2	An end-to-end model for roman numeral analysis	107
7.2.3	Comparing hidden states from pre-trained and end-to-end models for RNA	109
7.3	Analyzing attention behavior	111
7.3.1	Attention span	112
7.3.2	Attention span & harmonic diversity	114
7.4	Understanding attention heads' relevance	115
7.4.1	Layer-wise relevance propagation	116
7.4.2	Attention heads' relevance and musical tonality	119
7.4.3	Is tonality learnt during pre-training? Applying LRP to a MLM	129

8	Adapting NLP methods for a task of multi-track music generation	135
8.1	METEOR: melody-aware texture-controllable symbolic music re-orchestration	139
8.1.1	Textural attributes	139
8.1.2	Tokenization, model & control strategies	141
8.1.3	Inference guidance for melodic fidelity	143
8.1.4	Zero-shot lead sheet orchestration	145
8.2	Evaluating METEOR	145
8.2.1	Baseline models	146
8.2.2	Objective & subjective metrics	147
8.2.3	Results	150
8.3	Towards realistic and humanly playable symbolic music generation .	154
9	Conclusion and future directions	157
9.1	Discussion and future directions	158
9.1.1	To what extent can NLP be applicable to MIR?	158
9.1.2	Towards lighter models	160
9.1.3	Towards model explainability	161
9.1.4	A need for benchmarking and comparative analysis	162
9.1.5	Exploring further models for symbolic MIR	163
9.2	Conclusion	164
	Bibliography	167
A	NLP tools for MIR: representations	197
B	NLP tools for MIR: models	207
C	LRP for multi-hot time-slice representation	223
D	Résumé étendu en français	227

List of Figures

1.1	Evolution of the number of research articles containing NLP-related words.	2
1.2	Thesis outline.	3
2.1	Possible segmentation levels in text and symbolic music.	11
3.1	Examples of sequence classification tasks in NLP and MIR.	24
3.2	Examples of token classification tasks in NLP and MIR.	26
4.1	Taxonomy of tokenizations for symbolic music.	36
4.2	Time-sliced based tokenization: piano roll representation.	37
4.3	Artificial sequentiality possibly introduced in a tokenization strategy.	38
4.4	Elementary tokens examples: MIDI-like, REMI and DadaGP.	40
4.5	Multi-track parsing.	42
4.6	Byte-Pair Encoding (BPE) algorithm on text data.	44
4.7	Composite tokens examples: Octuple and Compound Word.	46
4.8	Word2Vec training paradigms in text: skip-gram and CBOW.	49
5.1	Fully connected neural network.	56
5.2	Configurations of sequential models according to the input and output sizes.	58
5.3	Transformer architecture, multi-head attention and self-attention mechanism.	62
5.4	Training paradigms: end-to-end training vs. pre-training + fine-tuning. . . .	65
5.5	Text and musical self-attention visualization.	72
6.1	Interval-based tokenization.	81
6.2	Performance comparison between absolute and intervalized tokenization strategies.	87
6.3	Best references of intervalized tokenizations on downstream tasks.	88

6.4	Analysis of labeled vertical pitch interval tokens for chord inversion identification.	89
6.5	Intervalized tokenizations based on interval classes.	91
6.6	Byte-Pair Encoding (BPE) algorithm applied to music tokens.	93
6.7	Comparison between BPE applied to text in multiple languages and music with various instrumentations.	94
6.8	Common start-of-phrase and end-of-phrase supertokens built through BPE.	96
6.9	Performance of the models improved with BPE on the task of start-of-phrase detection with polyphonic and monophonic datasets.	99
6.10	Ratio of supertokens containing <Pitch> atomic elements through BPE merges.	100
7.1	Roman numeral annotations used for functional harmony analysis.	106
7.2	Model architecture for roman numeral analysis based on Transformer encoders.	108
7.3	Logit lens on RNBert and our end-to-end model for the sub-task of local key.	111
7.4	Examples of attention spans.	112
7.5	Average attention span width by layer.	113
7.6	Examples of attention heads with attention spans impacted by harmonic diversity.	114
7.7	Average attention span width by layer, for low and high harmonic diversity.	115
7.8	Application of LRP in NLP.	116
7.9	Qualitative intuition behind LRP.	117
7.10	Elements involved in the LRP back-propagation.	118
7.11	Pixel relevances obtained via LRP back-propagation for different target labels.	118
7.12	Example of an LRP map.	119
7.13	Hypothesis 1a (label-wise): distribution of distances between label-wise Layer-wise Relevance Propagation (LRP) maps grouped by same tonality vs. different tonality.	124
7.14	Impact of the musical content on the attention head relevances when approaching a modulation.	125
7.15	Hypothesis 1b (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality.	127
7.16	Hypothesis 2b (bar-wise): dimensionality reduction of bar-wise LRP maps through t-SNE.	128
7.17	Hypothesis 2b (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality.	129
7.18	Hypothesis 1b MLM (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality.	131

7.19 Hypothesis 2b MLM (bar-wise): dimensionality reduction of sequence-wise LRP maps through t-SNE.	132
7.20 Hypothesis 2b MLM (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality.	132
8.1 Example of re-orchestration in Haydn’s Symphony No. 94 (Mvnt. 2).	136
8.2 METEOR’s re-orchestration task.	138
8.3 Pianoroll of a 8-bar re-orchestration generation by METEOR with various textural and instrumentation constraints.	139
8.4 Examples of bars with different levels of bar-wise textural features.	140
8.5 Examples of bars with different levels of bar-wise and track-wise textural features.	141
8.6 Architecture of METEOR, based on MuseMorphose.	142
8.7 Example of a token sequence used as input by METEOR.	143
8.8 Screenshots of the user study for METEOR subjective evaluation.	149
8.9 Projection through t-SNE of <Track> tokens embeddings.	152
8.10 Model comparison results according to the user study.	153
8.11 Impact of the number of source and target instruments on METEOR’s orchestrations.	153
8.12 Extract of a score generated by METEOR with difficult fingerings which may not respect physical constraints of the instrument.	154
C.1 LRP for multi-hot representation.	223

List of Tables

6.1	Tokenization studied in this chapter, including absolute-REMI and interval-based tokenizations based on REMI.	83
6.2	Description of the datasets used for pre-training and downstream tasks to evaluate intervalization.	85
6.3	Ratio of overlapping BPE supertokens between two musical phrases.	95
7.1	Performances of multi-objective RNA models.	110
7.2	Jensen-Shannon Divergences and average distances between LRP maps computed from pairs of same and different labels.	123
7.3	Performance of the masked language model pre-training.	131
8.1	Models related to the style transfer sub-tasks performed by METEOR.	137
8.2	Token vocabulary of METEOR.	144
8.3	Objective metrics for the re-orchestration task, comparing METEOR and baseline models.	150
8.4	Average pitch of melodic instruments with octave inference in the generated music by METEOR compared to their real instrumental range.	151
A.1	Overview of event-based tokenization strategies based on <i>elementary</i> tokens.	198
A.2	Overview of event-based tokenization strategies based on <i>composite</i> tokens.	202
B.1	Recurrent models applied to symbolic music.	208
B.2	<i>End-to-end</i> Transformer-based models applied to symbolic music.	212
B.3	<i>Pre-trained</i> Transformer-based models applied to symbolic music.	217

Glossary

BERT Bidirectional Encoder Representations from Transformers.

BOW Bag-of-words.

BPE Byte-Pair Encoding.

CRF Conditional Random Field.

GAN Generative Adversarial Network.

GPT Generative Pre-trained Transformer.

GRU Gated Recurrent Unit.

GTTM Generative Theory of Tonal Music.

HMM Hidden Markov Model.

ISMIR International Society of Music Information Retrieval.

JSD Jensen-Shannon Divergence.

LLM Large Language Model.

LRP Layer-wise Relevance Propagation.

LSTM Long-Short Term Memory.

MIDI Musical Instrument Digital Interface.

MIR Music Information Retrieval.

MLM Masked Language Model.

NLP Natural Language Processing.

NLP4MusA NLP for Music and Spoken Audio.

RNA Roman Numeral Analysis.

RNN Recurrent Neural Network.

TALN Traitement automatique du langage naturel.

TF-IDF Term Frequency-Inverse Document Frequency.

VAE Variational Auto-Encoder.

Chapter 1

Introduction

Natural language and music

Musical phrase, musical discourse, call and response. . . Musicians, conductors, musicologists, and even simple music enthusiasts often use these terms to describe, analyze, interpret music, or communicate with each other about music. Yet, these terms originally come from the lexical field of natural language. The need for music to borrow linguistic terms to describe itself is not accidental as it reflects a deep connection between music and natural language.

Leonard Bernstein declared about this relationship between these two forms of expression ([Bernstein, 1976](#), p. 10):

Music is the Universal Language of Mankind. [...] by building analogies between musical and linguistic procedures, couldn't that cliché about the Universal Language be debunked or confirmed, or at least clarified?

(L. Bernstein, The Unanswered Question)

This relation is complex as they can share multiple similarities, – such as common functions, structures, or perceptive phenomena – yet each also exhibits aspects that the other lacks or for which there is no direct equivalent, such as polyphony in music or semantics in language ([Jackendoff, 2009](#)). These similarities and specificities between language and music will be further developed in [Chapter 2](#).

This analogy between music and natural language, particularly symbolic music and text, also manifests when considering computer science research. Recognizing the many parallels between language and music, the Music Information Retrieval (MIR) community has quickly taken advantage of the advances in Natural Language Processing (NLP) to borrow and adapt their methods for computational music data processing.

Natural language processing and symbolic music information retrieval

Natural Language Processing (NLP) is a field at the crossroads between linguistics and computer science that focuses on the interaction between computers and human language. Its main purpose is to allow computers to deal with human languages while taking into account their characteristics, such as syntactic or semantic properties which are essential for language understanding, interpretation or generation. While *natural language* can be depicted either under an oral format (*i.e.* speech) or a written form (*i.e.* text), this thesis only studies language as *text*.

The field of **Music Information Retrieval (MIR)** combines musicology and computer science to develop techniques for analyzing music or retrieving music-related data. It has been extended in recent years to encompass techniques for music generation as well. While audio files encode music as sound, at a low representation level such as waveforms or spectrograms (Müller, 2015), *symbolic music* consists in abstract notations representing concepts such as notes, chords, or intervals, that compose musical scores. In practice, symbolic music remains prevalent in digital music production mainly relying on the MIDI format, which stands as an ubiquitous standard within digital audio workstations (DAWs). The scope of this thesis is limited to *symbolic* music representations.

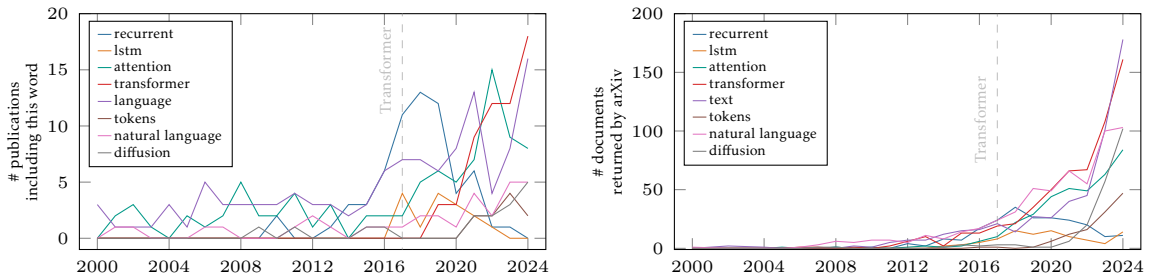


Figure 1.1: Evolution of the number of research articles containing NLP-related words. (Left) Number of ISMIR papers with NLP-related words in their abstracts from 2000 to 2024. (Right) Number of arXiv preprints returned by the API query “music AND <term>”.

To address tasks that sometimes overlap between text and symbolic music, as well as their shared nature as sequential representations, the MIR community has closely followed advances in NLP by adapting successful tools from this field. As a result, a prominent amount of symbolic music generation or analysis approaches have been adapted or inspired from NLP methods. Figure 1.1 describes the number of publications from the International Society of Music Information Retrieval (ISMIR) conference that include NLP-related terms in their abstract as well as music/NLP-related arXiv preprints. The rise of Transformers (Vaswani et al., 2017) from 2017

has largely contributed to increase these references and a large number of the NLP-derived state-of-the-art models in symbolic MIR are now based on this model.

This trend has encouraged dedicated initiatives in the MIR community, such as the organization of the workshop NLP for Music and Spoken Audio (NLP4MusA)¹ held between 2020 and 2024 or the new workshop LLM4A (Large Language Models for Music & Audio)² introduced in ISMIR 2025. In addition, more and more overviews of deep learning approaches for music generation, including NLP-based methods, are presented as tutorials or keynotes at conferences such as ISMIR³ or CMMR⁴.

In this thesis, we propose a taxonomy of existing NLP methods adapted to the MIR field, structured into three main dimensions: *sequential representations*, *models* and *tasks*. This overview enables the identification of potential areas for improvement across these three axes of NLP applications in MIR— axes to which we contribute in this thesis through targeted technical developments. While these technical contributions can result in marginal improvements when considered individually, this thesis supports that NLP can serve as a source of inspiration for MIR research – not merely as a collection of tools to be applied to symbolic music data, but as a framework that must be initially driven by fundamental musical questions.

Thesis outline and contributions

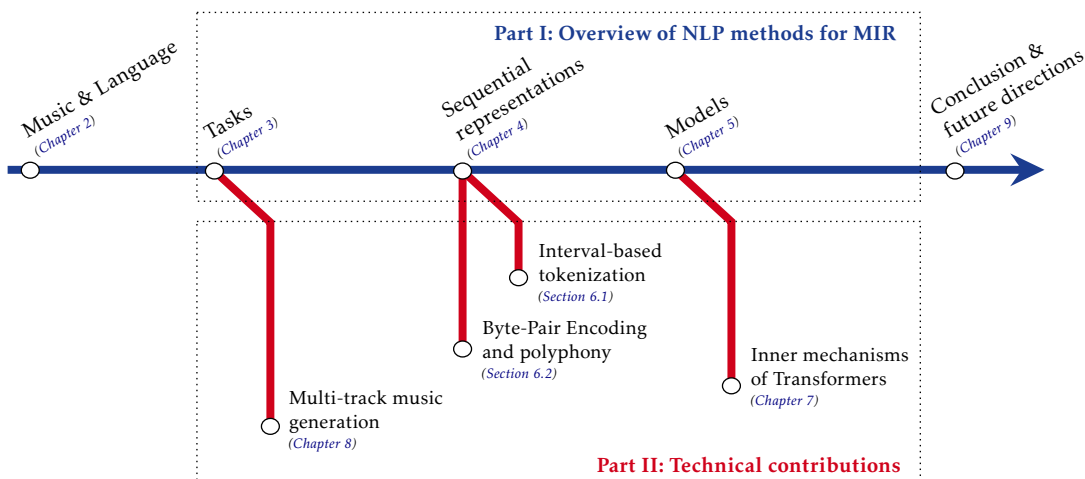


Figure 1.2: Thesis outline. The thesis is divided into two parts: an overview of NLP methods for symbolic MIR (blue line) and technical contributions (red lines). Both parts follow a common structure organized around *sequential representations*, *models*, and *tasks*.

¹<https://sites.google.com/view/nlp4musa-2024/home>

²<https://m-a-p.ai/LLM4Music/>

³Tutorial at ISMIR 2023: [Transformer-based Symbolic Music Generation](#)

⁴Keynote at CMMR 2023: [Deep Learning-based Automatic Music Generation: An Overview](#)

This thesis is organized following the structure presented in [Figure 1.2](#). First, [Chapter 2](#) presents high-level similarities between language and music, in particular, text and symbolic music, which partly motivate the use of NLP methods to process symbolic music. They can share similar high-level characteristics such as their purposes or induce cognitive reactions, but also similarities when seen as data such as their sequential aspect. This proximity between text and music lead to practical technical convergences when applied to NLP and MIR.

The manuscript then presents the various contributions of this thesis, organized into two main parts: a structured survey of existing NLP methods for symbolic music processing, and technical contributions that were achieved by adapting NLP methods to symbolic music analysis and generation.

A structured overview of NLP methods for MIR

[Part I](#) is dedicated to an extensive overview of existing NLP methods adapted for symbolic music information retrieval and symbolic music generation ([Le et al., 2025a](#)). This overview is organized into three key axes:

- Starting from a musical perspective, [Chapter 3](#) reviews the links between text and symbolic music **tasks**. While text and music remain distinct types of data, leading to in domain-specific tasks in NLP and MIR, these two fields still share common analysis tasks (*e.g.* token classification tasks, sequence classification tasks...), as well as generative tasks (*e.g.* priming, infilling...). Such similarities between text and music tasks may contribute in motivating the use of the following NLP methods in the MIR community.
- [Chapter 4](#) reviews existing **representations** of symbolic music as sequences. *Tokenizing* (*i.e.* representing data under a sequential format) music in the same way as text is not straightforward due to numerous differences between them, such as polyphony in music or the multiplicity of information contained in a single musical note. Therefore, several tokenization strategies for symbolic music have been developed for different purposes. We propose a taxonomy for describing and classifying music tokenization strategies.
- Such sequential representations of music are often used as inputs of NLP-based **models** presented in [Chapter 5](#). These models started with shallow models followed by recurrent networks. However, the rise of Transformers has led to the prominence of this model in several MIR tasks. We also propose a taxonomy describing these models through their technical aspects such as their architecture, training paradigm or music-specific mechanisms.

Technical contributions

Beyond this formal organization of the field, [Part II](#) presents technical contributions inspired by NLP methods for symbolic music processing in each of these three axes.

- Two contributions on *sequential representations* of symbolic music are detailed in [Chapter 6](#). These contributions focus on improving the expressiveness of music tokenization strategies. A first work analyses the impact of **Byte-Pair Encoding (BPE)** on monophonic and polyphonic music in a task of musical phrase detection ([Le et al., 2024](#)). A second work explores alternative **musical interval-based tokenization strategies** in the context of multiple analysis tasks ([Le et al., 2025b](#)).
- In [Chapter 7](#), we study *models* through the lens of model explainability. In particular, we analyze how the **attention mechanism** of an encoder-only model behaves in a task of functional harmony analysis, in terms of attention span and attention heads' relevance. We quantify this relevance through Layer-wise Relevance Propagation (LRP), which has been used for NLP models explainability.
- [Chapter 8](#) presents METEOR, a model for **automatic multi-track music generation**, as an application of NLP methods for a MIR *task* ([Le and Yang, 2025](#)). The model performs a task of automatic re-orchestration with melodic fidelity and textural controls. It is based on a variational auto-encoder with Transformer blocks and implements token constraints for these controls.

This overview of NLP methods for symbolic music and these technical contributions highlight potential limitations of NLP approaches in effectively adapting for symbolic music processing. [Chapter 9](#) discusses of such limits and suggests future directions for how the MIR community can continue drawing inspirations from NLP advances for future research. This directions include advances towards lighter models, improved model explainability, and the development of benchmarks for symbolic music research. We finally highlight the main takeaways of this work.

A summary of the works published and released during this thesis is provided in the [bibliography](#) of this manuscript. Moreover, audio extracts corresponding to the musical scores illustrated in the figures throughout the manuscript are available at <https://dinhviettoanle.gitlab.io/phd-thesis-companion>.

Chapter 2

On parallelisms between music and natural language

2.1	Music and language as communication means	8
2.1.1	Functions of music and natural language	8
2.1.2	Cognitive responses to music and natural language	9
2.2	Symbolic music and text as structured data	10
2.2.1	Hierarchical representations	10
2.2.2	Musical and textual <i>grammatical</i> rules?	13

Several works draw connections between music and language as evidenced by book titles such as *The Language of Music* (Cooke, 1959), *The Music between Us: Is Music a Universal Language?* (Higgins, 2012), *Music and Discourse: Towards a Semiology of Music* (Nattiez, 1990). Indeed, parallels between music and natural language are often drawn, as music is often considered as a linguistic system (Jackendoff, 2009). These parallels occur in many dimensions such as their human usage as communication means (Section 2.1), but also, from a more objective viewpoint, as structured types of data when considering text and symbolic music representations (Section 2.2). These parallels are however most often nuanced by significant differences specific to each domain.

All these high-level alignments between text and symbolic music likely motivated the adaptation of NLP computational methods for symbolic music processing, as described throughout this thesis. In this chapter, we propose to gather links between music and language that might contribute to justify this influence.

2.1 Music and language as communication means

Both music and natural language are specific to human species and are learned through imitation. Both can be deployed under two modalities: an annotated form (text, sheet music) and an auditory form (speech, musical performance) (Fornäs, 1997). However, both similarities and differences emerge when examining the functions of music and natural language, as well as humans' physiological responses when exposed to both modalities.

2.1.1 Functions of music and natural language

Communication is a central function in language because it conveys ideas, thoughts, concepts or propositions. In contrast, the question of defining functions of music has been extensively studied and discussed (Jackendoff, 2009; Fornäs, 1997; Zbikowski, 2009). In music, communication is often considered only one of several functions (Merriam and Merriam, 1964). This musical communication is often seen as serving other purposes than conveying ideas: the concept of semantics, which is pivotal in language, is missing or at least not essential to the appreciation of music. A popular maxim states (Eckstein and Reinfandt, 2006):

Writing about music is like dancing about architecture.

Music may not carry any literal meaning, or at least that cannot be compared to linguistic meaning (Lerdahl and Jackendoff, 1996). Bernstein declared on this topic (Bernstein, 1959, p. 33):

Music, of all the arts, stands in a special region, unlit by any star but its own, and utterly without meaning [...] except its own, a meaning in musical terms, not in terms of words.

Music is instead more associated with *affect* and serves as an *emotional expression based on aesthetics* (Jackendoff, 2009). Beyond being provoked by music, this emotional characteristic is sometimes considered as *intrinsic* to the music: some elements in the music composition can represent or symbolize an emotion (Carr, 2004). Cooke illustrates this phenomenon by describing third intervals in Western music (Cooke, 1959, p. 57):

Western composers, expressing the 'rightness' or happiness by means of the major third, expressed the 'wrongness' of grief by means of the minor third [...].

This point of view that interprets musical meaning in terms of emotional descriptions is highly debatable, as these considerations often originate from cultural effects (Ozaki et al., 2023) or musical education. Indeed, in music and language, musical or textual symbols do not inherently carry meaning. Instead, meaning is attributed to these symbols because a particular community, from a specific era or culture, collectively establishes an agreement to associate a certain set of signs with a particular concept (McClary, 2002, p. 21). Moreover, in the same way that language can reuse past content to construct metaphors or allegories, typically as references to existing works, music can also use reification of motives or chord progressions to evoke concepts or emotions (Wingstedt et al., 2010) or even text painting (Zbikowski, 2009) aiming at illustrating musically a text.

Additionally, there is a notable asymmetry in music and language learning (Waller, 2010): a majority of people can read and write text nowadays, whereas only a minority can read music, and even fewer can compose it. A gap between *perception* and *understanding* can exist in music, compared to language, where reading a text is usually directly linked to understanding the meaning of a text. In other words, interpreting music needs making actual music (Adorno and Gillespie, 1993). This is especially true since analyzing music relies on personal interpretation, which can be biased by culture, potentially leading to distinct levels of understanding or even diverse interpretations from different individuals.

Thus, the concept of sense, meaning, or semantics, which is prevalent and central in text, is far from being well defined in music. The question of attaching meaning or semantics to music has been a subject of extensive debate for centuries and is unlikely to have a universal answer. Returning to a technical standpoint, this epistemological debate underscores the need of carefulness when applying NLP tools for symbolic music.

2.1.2 Cognitive responses to music and natural language

Music and language can be produced or received by humans under an auditive or written form. Several common physiological phenomena have been found using language or music as stimuli.

Grammatical and syntactic rules induce expectancy in both language and music (Jackendoff, 2009; Pearce, 2018). Interestingly, transgressing these rules lead to similar cognitive reactions for the interlocutor or the listener (Pulvermüller and Assadollahi, 2007). For example, it has been proved that semantic processing in language induces comparable cognitive behaviors as harmonic processing in music in terms of event-related potentials (Besson and Schön, 2001). These perceptive similarities with natural language based on underlying rules in music may hint towards an existing *grammar* in music (Section 2.2.2).

This expectancy extends to stylistic or language enculturation, beyond local phe-

nomena as grammar. In music, cognition studies have shown that Western listeners' perception of particular rhythms differ in comparison with listeners exposed to non-Western music (Haumann et al., 2018). Interestingly, for language, this ability to discriminate phonetics in different languages tends to decline with age. Instead, this decline can be counter-balanced more effectively through social interaction than through long-term exposure to foreign languages (Kuhl et al., 2003).

To a larger extent, parallels can be found when writing music and text. There is typically a gap in hand-writing speed between experienced and beginner musicians, as well as differences in the stroke order used to draw note heads, stems, and beams (Bertiaux et al., 2023). This phenomenon is also observed in text handwriting, particularly in logographic languages like Chinese, where children exhibit varying stroke order and precision (Chang and Yu, 2022). In addition, handwriting speed also increases with age, which can be seen as language proficiency level (Graham et al., 1998).

2.2 Symbolic music and text as structured data

Language and music are often directly associated due to their similarities when seen as structured data. For example, this is the case with *lyrics* which are words aligned with a musical melody. Not every melody can pair with any text, as both must share compatible structural patterns. For instance, in the second half of the 20th century, with the adoption of sacred music texts, originally sung as Gregorian melodies, to vernacular languages, multiple technical recommendations of such process have been prescribed in order to make the music reflecting the structure of the text:

The melody adorns the words, grows out of them and is closely united to them by the prosody, the meter, the quantitative and qualitative characteristics of the language, the tonic accent, the sentence structure and the very organic nature of the words themselves.

(Sacred Music, Volume 110, Number 3, 1983)

This link in terms of prosody and syllabic meter, and rhythmic patterns, is one of the multiple structural similarities between language and music seen as hierarchical structures (Section 2.2.1). Going further, this structural aspect may be governed by high level *grammatical* rules (Section 2.2.2).

2.2.1 Hierarchical representations

Text and symbolic music representations are both semiotic systems (Chomsky, 1980) based on arrangements of symbols. Text is built on characters or ideograms and

written music can be transcribed with a variety of symbols derived from various notation systems such as standard notation, numbered notation or tablature. Similarities can be found between levels of segmentations of text and symbolic music. However, when focusing on what these symbols truly represent, key differences, such as rhythm or simultaneity in music, make these two modalities distinct.

Segmenting text and music – Both can be represented as sequences of elements which can be segmented or grouped at different levels (Figure 2.1). Text can be segmented into characters, syntactic phrases, sentences and paragraphs, In the same way, music can be segmented into temporal units such as notes, motifs, musical phrases, or sections (Lerdahl, 2012).

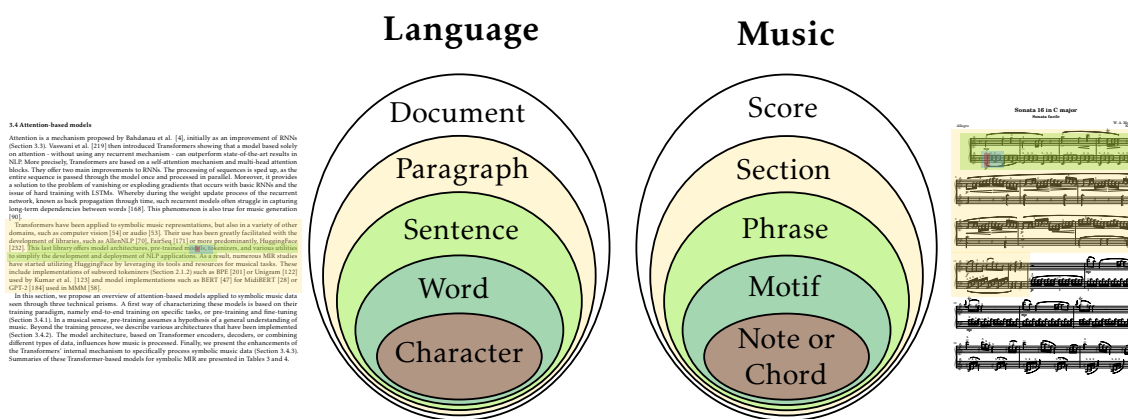


Figure 2.1: Possible segmentation levels in text and symbolic music. Such segmentations can, however, include more or less fine-grained levels and their delimitations can be ambiguous.

Text and music rely on common high-level structures. In text, the global structure of a discourse or an argumentation might neither include the same sections nor the same writing order with a novel. Similarly, music often follows established forms, like the fugue (Mann, 1987) or the sonata form (Hepokoski and Darcy, 2006) which have served as standards across different eras. Even in specific music genres such as religious chants, high-level structural organizations define forms such as tropes, canticles, or hymns.

At a lower level, identifying boundaries of musical motives and phrases remain subjective (Lidov, 1997) or can even overlap (Hentschel et al., 2021). Indeed, even if musical segmentation in motives or phrases is widely accepted because such motives are repeated at the level of a musical piece, as compared to language (Margulis, 2013), identifying the boundaries to such elements is often highly subjective. This is also reflected in the ambiguous terminology to denote such segmentations, including terms such as “motif, theme, period, or phrase” that can be used interchangeably (Cenkerová, 2017). This ambiguity is also present when defining these segments, such as the circular definition of a cadence in Western classical tonal music as presented by Bockmon and Starr (1962, p. 193):

A phrase is a unit of music structure terminated by a cadence. [...] A cadence is an effect of temporary or permanent conclusion marking the termination of a phrase.

In text, some languages can rely on whitespaces as a good approximation for word delimiters, even though segmentation remains unclear in some languages, such as Chinese (Huang and Xue, 2012), even in languages where whitespaces are present (Dien et al., 2001). In this sense, music might be more easily compared to unsegmented languages (Palmer, 2000) where word segmentation can be unclear (Huang and Xue, 2012).

Time dimension in text and music – Text and music can be perceived as elements unfolding in time (Zbikowski, 2009). Unlike language represented as sequences, Western music *must* be described following at least two dimensions, including time and pitch. While speech might have a temporal dimension in terms of speech rate (Wallin et al., 2001), text does not explicitly encode any of these rhythmic modulations. In contrast, musical rhythm is based on an isochronic grid (Jackendoff, 2009) in most of musical genres. In this grid, notes are notated with rigorous timings, in terms of onsets and durations, beyond some microtimings linked to performance embellishments or tempo changes. Moreover, this rhythmic dimension of music is a central characterization of a piece: two melodies with identical pitches but with different rhythms differ in their sheet music retranscription. This stands in contrast to speech, where discourses delivered with a fast and low speech rate are transcribed as text in exactly the same way. Rests are also integral to sheet music, unlike natural language, where silences are implicitly encoded through punctuation but appear mainly in speech for rhetorical or discourse effects.

However, despite this differences of rhythm annotation, a common point between text and symbolic music is the *implicit* existence of *accentuated* beats or syllables (Brown, 2017; Reich and Rohrmeier, 2024). Interestingly, this analogy goes further as accentuated syllables for a same language may depend on regional cultures (Schwab et al., 2012), in the same way as strong and weak beats may depend on the origin of the musician (Haumann et al., 2018).

Pitch and simultaneity in music – Language viewed as speech may also have a pitch dimension relating to prosodic contours (Wallin et al., 2001), but such pitch modulation is not explicitly represented in text. In contrast, pitch is a central characteristic of a musical melody even though its absolute value matters less than the intervals between pitches when recognizing a piece (Dowling and Fujitani, 1971).

However, while sequences of notes in monophonic music might be compared to words in text, *polyphony* adds a dimension that does not find any analogous element in text (Besson and Schön, 2001). Polyphony is based on simultaneous sequences of notes that create harmony. In text, such a concept of concurrent sequences of words is not applicable. The number of simultaneous sequences can vary throughout a piece and can be vastly different from each other, resulting in

different musical textures (Huron, 1989). This aspect leads to a whole set of tasks specific to music, such as harmonization as explored in Chapter 8, which aims at writing these concurrent events.

Though, an interesting parallel between musical pitch and textual words is their ability to represent the same concept through different forms. In music, this is referred to as *enharmonic notes*, where a common sounding pitch is named differently (e.g. C# and Db). This typically occurs in the context of tonal harmony (Section 2.2.2), often to enhance readability or to conform to conventional chord spelling and harmonic standards. Similarly, in text, a common concept can be referred to as from multiple words, a phenomenon called *synonymy*.

Symbol polymorphism – Finally, with a perspective of computational representation of text and symbolic music data, it may be noted that elements which constitute a musical score are less homogeneous than text data. While textual elements quite homogeneous (mostly characters or ideograms, some structural elements like punctuation or brackets), music symbols combine structural elements (bar, beat, etc.), note-related information (pitch, duration, dynamics, etc.) and global information (tempo, instrument, etc.). Such heterogeneity of information contained in a score may be linked to the fact that music is ultimately intended to be *performed*, in contrast with text which has its own finality.

Regarding sequential representations of symbolic music, this polymorphism possibly introduces an artificial sequentiality when modeling music because multiple musical features describing one temporal event must be ordered. For instance, characteristics, such as note-related information cited above, all describe a single temporal event but must be ordered for it to be presented sequentially. Going further, this artificial sequentiality is even accentuated by the modeling of simultaneous events mentioned above. Indeed polyphonic music is defined as simultaneous musical streams (chords or instruments) but *must* be encoded as a unidimensional stream of events. This aspect of artificial sequentiality is discussed further in Section 4.1.2, presenting solutions to overcome this issue.

2.2.2 Musical and textual *grammatical* rules?

Grammar is a set of rules governing how single elements (e.g. words) combine with other ones to build coherent structures (e.g. phrases or sentences). While grammar is central for natural language, the existence of a global grammar describing music is also not unanimously accepted, even in a specific style (Dempster, 1998). Multiple grammars have been proposed to describe music from a general point of view, such as the implication-realization model (Narmour, 1990). A notable example is the *Generative Theory of Tonal Music (GTTM)* (Lerdahl and Jackendoff, 1996), which relies on musical rules to generate or analyze music in a way that reflects the perception of tonal structures. GTTM comprises several components, each implementing rules to

construct musical structures, including:

- *grouping structure*, which divides music into hierarchical groups, such as motifs, musical phrases;
- *metrical structure*, which organizes music as alternations of weak and strong beats;
- *time-span reduction*, which identifies more important notes within a given time-span;
- *prolongational reduction*, which structures music as harmonic tensions and resolutions.

Harmonic concepts have been modeled as a grammar for music, in particular in the context of tonal music¹. They notably give directions to the musical stream by carrying different functions (Temperley, 2004; Adorno and Gillespie, 1993) to create tension, release or closure patterns like cadences (Fornäs, 1997). Interestingly, in the same way that words are polysemous and take on a meaning thanks to their local context², an isolated musical note or motif can also carry multiple functions: only the context in which it occurs within the piece defines its potential role.

Such harmonic rules are established by a specific musical style or era (Klein and Jacobsen, 2012): however, something which is considered “regular” in a style can appear as an “irregularity” in another style, while still being considered as music. Ducros (2012) notably argues that, because most people are predominantly exposed to tonal music, deviations in tonal contexts are easily perceived, whereas in atonal music, the notion of expectation may become irrelevant. He stated:

Non-specialists can spot two wrong notes in a tonal music, whereas specialists cannot spot 79 wrong notes in an atonal music. [...] Tonal music composers have at their disposal a language in which the public masters a large part of the rules, even if it is not able to state them.

(Des non-spécialistes repèrent deux fausses notes dans une musique tonale, alors que des spécialistes ne repèrent pas 79 fausses notes dans une musique atonale. [...] Les compositeurs de musique tonale avaient et ont à leur disposition un langage dont le public maîtrise une bonne partie des règles même quand il est incapable de les énoncer.)

¹Motivated by this parallel, our work in Chapter 7 typically explores whether deep learning models learn functional harmony in a manner similar to how NLP models learn grammar.

²For example, the word *key* can be used in different contexts: “a C major *key*”, “a *key* to unlock a door”, “a piano *key*”, “the Enter *key* on a keyboard” or “the *key* to success”.

This absence of “rightness” in music consolidates the idea that aesthetics plays a prominent role in music (Krausz, 2019). Going further, music might be closer to poetry than *natural* language when considering this aesthetic dimension. On this question regarding aesthetics in language and music, Bernstein stated (Bernstein, 1976, p. 79):

You see, language leads a double life ; it has a communicative function and an aesthetic function. Music has an aesthetic function only. For that reason, musical surface structure is not equatable with linguistic surface structure.

Consequently, in MIR, the evaluation of systems performing generative or even analysis tasks can be delicate due to this aesthetic dimension.

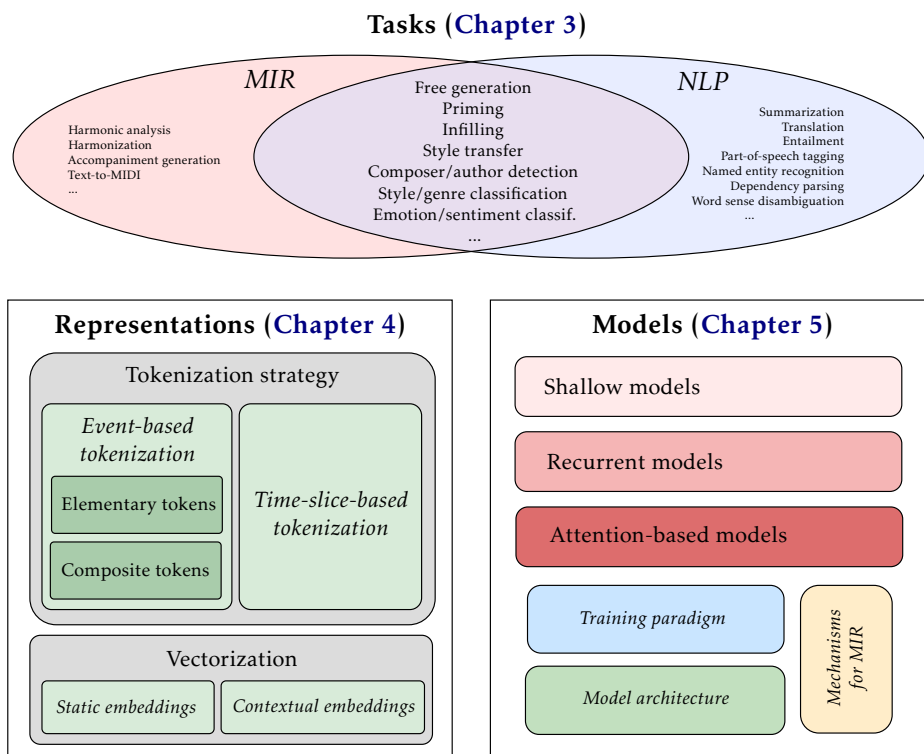
Part I

A Structured Overview of Natural Language Processing Methods for Symbolic Music Information Retrieval and Generation

This part is based on a journal article published in *ACM Computing Surveys* (Le et al., 2025a). This work has been realized in the context of a collaboration following an international research visit in the Audio, Music, and AI Lab (AMAAI) from Singapore University of Technology and Design (SUTD) in October 2023.

As depicted in Chapter 1, Natural Language Processing methods have been becoming more and more prominent in the field of Music Information Retrieval. The motivations of such adoption of NLP tools for symbolic music is likely motivated by the high-level similarities between language and music in general (Chapter 2).

On a more technical point of view, these contributions coming from NLP are of diverse nature. Therefore, this part is dedicated to a structured overview of NLP methods for symbolic music processing, organized around three axes: *similarities and specificities* of NLP and MIR tasks motivating *representations* of symbolic music inspired by NLP and *models* adapted from NLP for symbolic music.



- **Chapter 3:** The **task** is the purpose of a system, whether involving an objective evaluation or interactions with the final user. The shared ground between symbolic music and natural language processing in terms of tasks is wide, encompassing numerous analogous tasks from generative or analysis purposes, while keeping some specificities.

- **Chapter 4:** From a technical point of view, choosing a **representation** refers to encoding content (text or symbolic music) into a format suitable for computational processing. Adapting NLP models to symbolic MIR mainly involves sequential representations.
- **Chapter 5:** The **model** performs a task by processing a *representation* of the input content. Such a model can be rule-based or learned, with specific architectures, in particular for neural networks and different training paradigms. Inside these models initially developed for text processing, mechanisms can be specifically tailored for symbolic music data.

These three aspects are far from being independent from each other (*e.g.* a *model* needs a *representation* as inputs and a *task* to be trained on, while the *representation* itself may depend on the nature of the data relevant to the target *task*). Nevertheless, we have opted for this organization to present the three core components of a MIR project that can draw inspiration from NLP methodologies.

Chapter 3

MIR and NLP tasks: similarities and specificities

3.1	Tasks learned from labeled data	24
3.1.1	Sequence classification and regression	24
3.1.2	Token classification and regression	25
3.2	Tasks relying on unlabeled data	27
3.2.1	Unsupervised clustering and segmentation	28
3.2.2	A prominence of generative tasks in MIR an NLP	29
3.2.3	Unlabeled text and symbolic music datasets	31

Beyond the high-level existing parallels between text and symbolic music, the application of NLP methods to symbolic music processing is primarily driven by their potential to effectively address MIR *tasks*. In this chapter, we review task similarities that encourage MIR to adopt representations and models from NLP, even though many NLP tasks have no direct counterpart in MIR due to the differences between natural language and music.

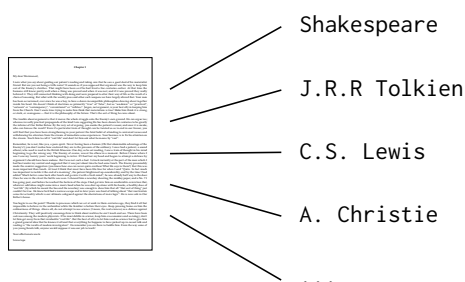
Nowadays, most of the tasks performed in NLP and MIR rely on data-driven learning methods. One possible way to organize these tasks is based on the type of data required to train the models, namely tasks depending on *labeled data* (Section 3.1) and tasks which can rely on *unlabeled data* (Section 3.2). From a model point of view, the first category of tasks rely on *supervised learning* while the second category may be considered as *unsupervised learning*. Naturally, all these tasks rely on *corpora* or *datasets* which have been compiled or created by the MIR and NLP communities.

3.1 Tasks learned from labeled data

Tasks relying on labeled data typically require annotations by experts both in music and text. These tasks are most often *analysis* tasks, in contrast with generative tasks. We assume text and symbolic music can be represented as *sequence* of elements, referred to as *tokens*. In practice, textual tokens can be words or subwords and music tokens can be notes, characteristics of a note, such as pitch, duration or instrument, or higher-level objects such as chord labels or slices of multiple notes. Representations of text and music as sequential data will be further detailed in [Chapter 4](#). Assuming this sequential representation of text and music, these tasks can be categorized into *sequence classification or regression* tasks ([Section 3.1.1](#)), where the full sequence is evaluated with a unique label, and *token classification or regression* ([Section 3.1.2](#)), where each single element of the sequence is assigned a label.

3.1.1 Sequence classification and regression

(a) Example of a text sequence classification task: author classification.



(b) Example of a symbolic music sequence classification task: composer classification.

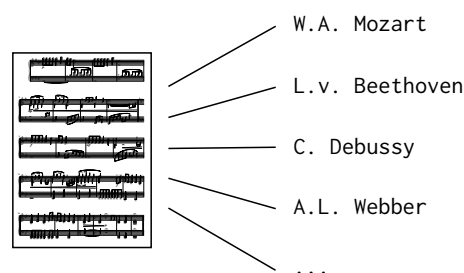


Figure 3.1: Examples of sequence classification tasks in NLP and MIR. The full sequence (*i.e.* a text or a score in these examples) is classified under a common label.

Tasks involving labeled data that aim to classify whole textual document or music piece, or text or music section often share similarities ([Figure 3.1](#)). Identifying or distinguishing its author can be expressed as a task of **text authorship attribution** ([Stamatatos et al., 1999](#)) or **music composer classification** ([Pollastri and Simoncelli, 2001](#)). More recently, an emerging task is the detection of AI-written text ([Elkhataf et al., 2023](#)) or AI-generated music which is starting to be considered in the audio field ([Vila et al., 2025](#)) but not in the symbolic music field. These tasks are closely related to **music genre classification** ([Corrêa and Rodrigues, 2016](#)) or **text style classification** ([Kessler et al., 1997](#)). To a larger extent, **folk song origin classification** ([Hillewaere et al., 2018](#)) can be analogous to **language detection** ([Jauhiainen et al., 2019](#)). All of these tasks involve recognizing and identifying across an entire sequence specific patterns that are characteristic to a composer/author, genre or regional structure which can be stylistic or structural. For training data-driven models,

datasets including symbolic music with basic metadata such as composers (Zhang et al., 2022), folk song origin (Schaffrath, 1995), or musical genres (Ens and Pasquier, 2021) are numerous. In the same way, text datasets often include metadata such as authors¹, writing style (Jasleen Kaur, 2014) or language (de Gibert et al., 2024). Beyond these objective tasks, tasks called **music emotion recognition** (Hung et al., 2021) and **sentiment analysis** from text (Wankhade et al., 2022), which both aim at labeling a full sequence with an emotion or sentiment, may be analogous as they share a part of subjective assessment from a user. In practice, these “emotions” are quantified: for music, the EMOPIA dataset (Hung et al., 2021) annotates emotion as valence and arousal values, while for text, sentiment is often restricted to particular scales, such as three-point scales composed of positive, neutral or negative sentiments (Mejova, 2009). This objectivity is also found in a task of difficulty assessment: in the same way as text must be understood by the reader to a certain degree, music can be more or less easily interpreted by a musician. Therefore, a task of **text readability assessment** (Deutsch et al., 2020) or **music difficulty assessment**, notably based on chord progressions (Vásquez et al., 2023) or fingering displacements (Chiu and Chen, 2012), can be both modeled as sequence regression tasks. This parallel should, however, be nuanced as the music difficulty assessment task is more closely related to the musical *performance*, rather than a possible “comprehension difficulty” of the written score.

Naturally, multiple tasks in a field do not find a direct counterpart in the other. In MIR, a task of key detection (Temperley, 1999) assumes a local context and builds upon the concept of key signature, a notion that has no equivalent in text. Similarly, in NLP, some tasks have no direct analogue in music. This includes topic analysis (Jelodar et al., 2021), textual entailment (Poliak, 2020), or author age or gender profiling (HaCohen-Kerner, 2022), which rely on sociolinguistic cues and semantic content that music does not directly convey or is debated (Sergeant and Himonides, 2016).

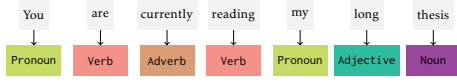
3.1.2 Token classification and regression

While sequence classification tasks aim at classifying a full sequence into a label, *token classification* tasks aim to assign one (or multiple) label(s) to each individual element – or token – within the sequence. Seen from a model viewpoint, such tagging task is typically performed by a sequence-to-sequence (seq2seq) model, where the input is a sequence of tokens and the output is the corresponding a sequence of labels of the same sequence length.

However, as further explored in Chapter 4, the nature or tokens involved in sequential representations of text and symbolic music differs significantly. Therefore, while similar model architectures can be used to perform such tasks, the nature of the labels annotating the tokens also significantly differ in their nature (Figure 3.2).

¹e.g. the [Gutenberg project](#) which gathers full books with metadata.

(a) Example of a text token classification task: part-of-speech tagging. A token is a word.



(b) Example of a symbolic music token classification task: roman numeral analysis. A token is a quarter note-long slice.



Figure 3.2: Examples of token classification tasks in NLP and MIR. Each token of the sequence (*i.e.* word or score slice in these examples) is assigned an individual label.

Text token classification tasks – Multiple tasks relate to *tagging tasks* which consist in assigning each token a syntactical or grammatical label. These includes **part-of-speech tagging** (Harris, 1962) which consists in assigning to each word of a text a particular part-of-speech (*i.e.* noun, verb, adjective, adverb...). This task is typically a disambiguation task, as a word can assume different roles depending on the context, regardless of its semantics. For example, the word “*play*” typically has a different tag in the sentence “I *play* the trumpet” and “I watched a *play* at the theater”. Similarly, **named-entity recognition** (Nasar et al., 2021) aims to identify and classify words into predefined categories, typically including entities such as people, locations, dates, etc. This task is also a disambiguation task as a word can take different category, or even not be a name entity: the word *may* can be used in the sentence “She was born in *May*” (date), “Brian *May* is a guitarist” (people) or “*May* I help you?” (not a named entity).

Beyond token-by-token tagging tasks, *segmentation tasks* can also be framed as token classification problems, where a contiguous sub-sequence of tokens is assigned a common label. In some languages, tasks of **phrase segmentation** (Huang et al., 2010) or **word segmentation** (Xue, 2003; Lee et al., 2003) are non-trivial as phrases, words or characters are not separated by punctuation marks such as spaces or periods. These segmentation tasks are not limited to languages using non-Latin alphabet. They can also occur in languages such as Vietnamese (Hieu Nguyen et al., 2025), where words are often considered as being formed from morphemes that are themselves separated by spaces. These tasks can thus be formulated as binary token classification tasks, where each token is assigned a label indicating whether it represents a phrase or word boundary.

Music token classification tasks – Token classification tasks in MIR can also refer to segmentation tasks. Similarly to textual structure segmentation, **musical structure retrieval** has been explored under the form of musical phrase retrieval (Guan et al., 2018) or musical form analysis (Zhao et al., 2023b) such as fugue form analysis (Giraud et al., 2015) or sonata form analysis (Allegraud et al., 2019).

At a lower level, single token tagging tasks are less common in MIR. In contrast with text where a single token can carry information by itself (*e.g.* word conveying semantic meaning, or subword carrying morphological functions, serving as prefix,

suffix, radical, ...), the lack of tagging tasks in MIR may be due to the lack of information carried by a *single* musical token in several representations (e.g. MIDI-based tokenizations or representations separating pitch, duration, onset, ...). However, MIR tagging tasks can instead operate on symbolic music data by modeling each element in the sequence as a higher-level musical object (Figure 2.1). At a bar-level, recent studies have focused on **analyzing musical texture** at a bar-level for piano music (Couturier et al., 2022) or orchestral music (Le et al., 2022). At a (sub-)beat level, **functional harmony analysis** – or roman numeral analysis (Nápoles López et al., 2021) – is often modeled as a tagging task where each (sub-)beat is annotated with multiple attributes, such as local key, chord degree or inversion. These annotations allow for the labeling of each chord with a roman numeral label, describing the nature of a chord and its function within a tonal context. In the same way as several NLP tasks described above, these tasks are disambiguation tasks as a chord can have different labels depending on the context, and can even be described differently depending on the annotator (Koops et al., 2019). In Chapter 7, we chose to focus on this task of functional harmony analysis² because it can be modeled as a tagging task, making it closely related to an NLP task. Local token classification is also used in tasks relating to music performance, for instance with **playing technique prediction** in the case of monophonic guitar music (D’Hooge et al., 2023).

These tasks require high quality annotations so that multiple studies or initiatives aiming at building highly accurate human-annotated datasets have been initiated. Focusing on piano music, the TAVERN dataset (Devaney et al., 2015) includes chord and phrase annotations of a corpus of classical themes and variations. Similarly, (Hentschel et al., 2024) focuses on a later period of music and includes functional harmony annotations. This dataset, among others, has been annotated within a unified framework to form the DCML corpora³. This annotation framework is extended for larger orchestral ensembles and symphonic music with the S3 dataset (Lin et al., 2024b), which also gathers texture annotations.

The need for highly accurate annotations constrains the size of datasets available for training data-driven models on these music analysis tasks. On the contrary, unlabeled datasets – far more numerous, both in MIR and NLP – can be leveraged for other types of tasks, in particular, generative tasks.

3.2 Tasks relying on unlabeled data

A variety of tasks rely on *unlabeled* music and text datasets. While generative tasks, such as text generation or music composition, are nowadays often the primary motivation for building large unlabeled datasets, a few analysis tasks can also be performed with unlabeled data, such as clustering, unsupervised segmentation and

²We give further musical details on this task in Section 7.2.1.

³https://github.com/DCMLab/dcml_corpora

their variants. From a model training perspective, models for these tasks are often trained through *unsupervised* or *self-supervised* learning.

3.2.1 Unsupervised clustering and segmentation

Some analysis tasks rely on unsupervised learning, where a model is only fed with inputs and must discover its underlying structures on its own. Two main classes of tasks gather NLP and MIR tasks relying on unlabeled data: segmentation and clustering.

While *segmentation tasks* can possibly rely on labeled data modeled as a token classification task, an unsupervised approach typically permits to automatically split a text or score into segments without having an *a priori* characterization of each segment. At the scale of words or subwords, subword tokenization algorithms for text (Sennrich et al., 2016) or music (Fradet et al., 2023a) can be considered as unsupervised **word segmentation** tools, as described in Section 4.1.2.1 about token grouping. Though, segmentation most often refers to the scale of the score or the document. Therefore, in music, **musical phrase segmentation** can be performed in an unsupervised way, while being less effective than supervised methods (Bassan et al., 2022). In NLP, the semantic nature of text allows it to be segmented into coherent blocks centered around the same topic, a task referred to as **topic segmentation** (Bai et al., 2023).

Clustering tasks aim to group data into a certain number of groups, called clusters, based on their similar in terms of particular characteristics. In the same way as supervised and unsupervised segmentation, clustering can be considered as an unsupervised way of performing classification, without relying on predefined classes or explicit descriptions of the expected clusters. In NLP, **topic modeling** (Grootendorst, 2022) aims at clustering entire or parts of documents by topic. Clustering of parts of a text document can be performed on multi-author documents, where sections are grouped according to their likely author (Tschuggnall et al., 2017). In MIR by contrast, the notion of topic is absent, and multi-author contributions to a piece of music typically occur across different stages of its creation (e.g. composition, orchestration...) rather than within the composition itself. However, clustering tasks in MIR are typically based on style with **musical genre clustering** (Cilibrasi et al., 2004). This allows to find proximity between musical modes (Liumei et al., 2021) or musical genres (Corrêa et al., 2011). In particular, building such clusters helps to interpret the proximity of samples within a cluster and analyze model errors by comparing specific features such as musical harmony (Dervakos et al., 2022).

Such clusters resulting from these tasks can be then used for *substitution* tasks. Given that elements within a common cluster share similar properties, substitution tasks consist in replacing an element of a text or music with one element of the cluster. More precisely, such replacement can be performed with text using syn-

onyms built via **word sense induction** (Amrami and Goldberg, 2019) or with score slices occurring within the same context (Herremans and Chuan, 2017) found via embedding projection.

3.2.2 A prominence of generative tasks in MIR and NLP

While the field of music information retrieval did not *originally* include music generation, the MIR community has shown a growing interest for music generation tasks in the last few years. For instance, ISMIR call-for-papers now include “music generation” among their tasks, or the MIREX competition now include audio and symbolic music generation⁴. This current trend is notably reflected in the prominence of MIR survey articles that mostly focus on generative systems. These surveys generally fall into two categories. A first category classifies generative systems from a *technical perspective*. These systems rely on methods such as grammars or Markov chains (Fernández and Vico, 2013) or more recently on deep learning methods (Briot et al., 2020), organized by model architecture (Wang et al., 2023; Ma et al., 2024) or types of generation conditions (Zhu et al., 2023b; Dash and Agres, 2023). A second category of overviews bring together works that share a *common musical purpose or task* (Herremans et al., 2017) and categorize them based on the nature of the generated content (Liu and Ting, 2017) or by the context surrounding the generated content (Ji et al., 2023).

Generative tasks are most often performed through self-supervised learning (*i.e.* predicting parts of the input itself, by learning representations and patterns without external annotations). Several generative tasks between NLP and MIR are similar. Such tasks often involve a predefined context, which prompts the model to generate content conditioned on this initial input: these tasks are often referred to as *constrained generation*. **Music priming** (Huang et al., 2019) and **text continuation** (Radford et al., 2019) involve generating a coherent continuation from a given starting music or text sequence. Similarly, **music infilling** (Guo et al., 2022) and **text infilling** (Donahue et al., 2020), consist in completing a segment of a music piece or text between two given boundaries. At the scale of a piece or a document, *style transfer* or *controlled generation* can exist in both MIR and NLP. This task aims at rewriting a music piece or text into another “style” while preserving the core “content”. In audio music, timbre style transfer refers to the modification of sound characteristics of an audio signal, such as transformation of the sound of the perceived instrument. For symbolic music, **composition style transfer** (Dai et al., 2018) aims at transforming musical genre (Cífka et al., 2020) or textural characteristics (von Rütte et al., 2023) of a reference music. Similarly, in NLP (Jin et al., 2022), **text style transfer** can be defined through high-level elements such as formality (Chawla and Yang, 2020), politeness (Madaan et al., 2020), or diachronic language style (Jhamtani et al., 2017). To a larger extent, **machine translation** (Dabre et al., 2020) can also be interpreted as a type of style transfer, where an underlying

⁴2024 MIREX competition: https://www.music-ir.org/mirex/wiki/2024:Main_Page

semantic content is preserved, while the form changes through a different language. More recently, text-conditioned generation has become more and more popular for the general public. In NLP, this includes **chatbot dialog**⁵ which internally relies on a prompt-based system, and **text-conditioned music generation** (Lu et al., 2023) which must involve multimodal models. In the audio domain, commercial tools have been released such as Suno⁶ or Udio⁷.

However, NLP and MIR also include numerous tasks that are inherent to one field. For example, **harmonization** or **accompaniment generation** (Zhao et al., 2024b) in music relies on the concept of harmony which exists due to the existence of music polyphony, a concept which is inexistant in text. In contrast, **text summarization** operates under the assumption that the semantic content of a text can be condensed, whereas music may lack a direct equivalent to semantics, making a parallel less straightforward for this task. These tasks specific to each field also reflect fundamental differences between these two types of data.

In contrast with tasks relying on labeled data, unsupervised tasks, in particular generative tasks, are not evaluated with respect to a ground truth. To this end, a variety of task evaluation methods have been implemented in both fields for such generative tasks. In NLP, generative models are usually evaluated on benchmarks with a variety of metrics (Celikyilmaz et al., 2021), such as BLEU-score (Papineni et al., 2002) for machine translation or Semantic Textual Similarity (Agirre et al., 2016) for text summarization. MIR generative models are usually evaluated through user studies, taking the form of preference selection (Shu et al., 2024), ranking (von Rütte et al., 2023) or scoring (Luo et al., 2024). To counterbalance this subjective aspect, multiple quantitative music-related metrics have been proposed to evaluate music generation (Yang and Lerch, 2020). These include pitch-related metrics (Wu and Yang, 2023b), rhythm-related metrics (Wu and Yang, 2020) and harmony-related metrics (Yeh et al., 2021). Inspired by the Fréchet inception distance in computer vision (Heusel et al., 2017) which was later adapted to audio music with the Fréchet audio distance (Kilgour et al., 2019), the Fréchet music distance (Retkowski et al., 2025) has been proposed to evaluate the realisticness of generated music. It is evaluated as a distance between distributions of reference and generated symbolic music embeddings. The question of evaluating symbolic music generation without user study is still a challenge: while the task of symbolic music generation from the 2024 MIREX challenge does include objective metrics as a reference, the final ranking of participants still relies on subjective metrics⁸.

⁵<https://chat.openai.com>

⁶<https://suno.com/home>

⁷<https://www.udio.com/>

⁸https://www.music-ir.org/mirex/wiki/2024:Symbolic_Music_Generation

3.2.3 Unlabeled text and symbolic music datasets

In order to train models on these generative tasks from a self-supervised way, several large unlabeled datasets of text and symbolic music have been proposed.

In NLP, such datasets are often compiled through web crawling (Wenzek et al., 2019), and have been released for model training. These datasets may be unstructured, aggregating vast amounts of text from webpages⁹ or books (Bandy and Vincent, 2021), and can be used for general generation tasks such as infilling or text continuation, or as pre-training datasets. Other datasets are task-specific, such as those for translation – where bilingual texts are collected¹⁰ – or for question answering which include texts, questions and corresponding answers (Rajpurkar et al., 2016). For some languages, text datasets sizes typically range from several of gigabytes to terabytes (Liu et al., 2024). In comparison, symbolic music dataset are much smaller.

MIR in contrast, features multiple symbolic music file formats which can be used to encode music in different manners. *Sheet music* can be distributed under text formats, with ABC or plaine-and-easie¹¹ which are more suitable for monophonic tunes, or **kern¹² and lilypond¹³ which aims at also describing score layout. Structured format, in particular through markup language, are also often used such as musicXML or MEI (Music Encoding Initiative)¹⁴. For particular instruments or type of music, specific format can be used, such as gabc¹⁵ derived from ABC notations for Gregorian chants or gpif for guitar tablatures (Cournut et al., 2021), based on a markup language. However, symbolic music datasets are most often shared under a MIDI (Musical Instrument Digital Interface) format, which originally encodes a musical *performance*, more than its sheet music. For example, score layout or some musical characteristics such as enharmonic pitches are not encoded in the original format.

These include large crawled MIDI collections such as LakhMIDI (Raffel, 2016) or MetaMIDI Dataset (Ens and Pasquier, 2021) later extended with the GigaMIDI Dataset (Lee et al., 2025). Beyond these general-purpose music datasets, some can be specific to music styles or instrumentations, focusing on orchestral music (Liu et al., 2022), piano music (Kong et al., 2020), chorales (Boulanger-Lewandowski et al., 2012), Japanese vocal music (Nakamura et al., 2023), folk tunes (Schaffrath, 1995) or pop music (Wang et al., 2020a). Other datasets with specific music representations, such as guitar tablatures (Sarmiento et al., 2021) or chords-only (de Berardinis et al., 2023), have been built for non-MIDI generative systems. Datasets linking symbolic music and other types of data are built for multimodal models for audio-MIDI

⁹<https://commoncrawl.org/>

¹⁰<https://huggingface.co/wmt>

¹¹<https://www.iaml.info/plaine-easie-code>

¹²<https://www.humdrum.org/>

¹³<https://lilypond.org/index.html>

¹⁴<https://music-encoding.org/>

¹⁵<https://gregorio-project.github.io/gabc/>

alignment gathering MIDI and their corresponding audio such as the MAESTRO dataset (Hawthorne et al., 2019) or more recently, text-to-MIDI (Melechovsky et al., 2024) and video-to-MIDI (Cardoso et al., 2024). Such video and MIDI alignment can also serve for specific tasks such as piano gesture analysis (Gan et al., 2025).

These datasets can also serve as a foundation for creating new ones, in particular as training datasets for audio-to-MIDI models (Hawthorne et al., 2018). Given the abundance of music audio compared to symbolic music, such models can be used to build MIDI datasets from transcribed audio – examples include Aria-MIDI (Bradshaw and Colton, 2025) for piano music and Pop1k7 (Hsiao et al., 2021) which gathers piano transcription of pop songs. For a comprehensive overview of music generation datasets, refer to Ji et al. (2023) or on the ISMIR website¹⁶.

This chapter provided an overview of MIR tasks that share analogies with NLP tasks. The similarities in these tasks partly motivates the use of NLP approaches in MIR, but also the possibility for symbolic music and text to be represented as sequences. These *representations* of music as sequential data are presented in the next chapter.

¹⁶<https://www.ismir.net/resources/datasets/>

Chapter 4

Representations of symbolic music as sequences

4.1	Tokenization strategies	36
4.1.1	Time-slice-based tokenization	37
4.1.2	Event-based tokenization	38
4.1.2.1	Elementary tokens: music as sequence of individual features	39
4.1.2.2	Composite tokens: music as sequence of feature combinations	45
4.2	Comparing tokenization strategies	47
4.3	Embedding music tokens for model processing	48

Text data inherently follows a sequential structure composed of elements spanning from individual characters to full sentences. In contrast, representing musical content as a sequence of homogeneous elements is not as straightforward. The multiplicity of information included in a single note (pitch, duration, position, etc.) and the common occurrences of simultaneous notes (polyphony, chords and melody, etc.) make the problem more complex than with text. However, this sequential representation is necessary for the musical data to be subsequently processed by sequential models, which were initially designed to handle text data.

Naturally, other representations of symbolic music have been proposed to be processed by other types of models. *Piano rolls* (Walder, 2016) were originally conceived as matrix representations of music represented with time along the horizontal axis and pitches along the vertical axis (Figure 4.2) and can be typically processed by models brought from image processing studies. More recently, music has also been explored as *graph data* (Jeong et al., 2019b; Karystinaios and Widmer, 2022) to be processed by graph neural networks. Under this representation, notes are vertices and relations between notes (same onset, consecutive notes, overlapping notes, ...) are represented by edges.

In the realm of NLP models applied to symbolic music, this chapter presents and proposes a structured overview of the various methods that have been proposed to represent *music as sequences of elements*. [Appendix A](#) gathers tables presenting exhaustive descriptions of existing sequential representations of music.

4.1 Tokenization strategies

Tokenization refers to the process of representing complex content into a sequence of elements for computational processing. In NLP, tokenization is the task of segmenting a sequence of atomic elements - characters - by grouping them together into informative *tokens* ([Mielke et al., 2021](#)), such as subwords, words, or multiple-word expressions. The rise of NLP models in MIR has naturally encouraged the adoption of this term for music representations. Though, a *musical token* can take various forms and represent diverse types of musical information (*e.g.* the pitch of a note, its duration, a slice of a *piano roll*, ...). Therefore, we propose a taxonomy of tokenization strategies in symbolic MIR represented in [Figure 4.1](#).

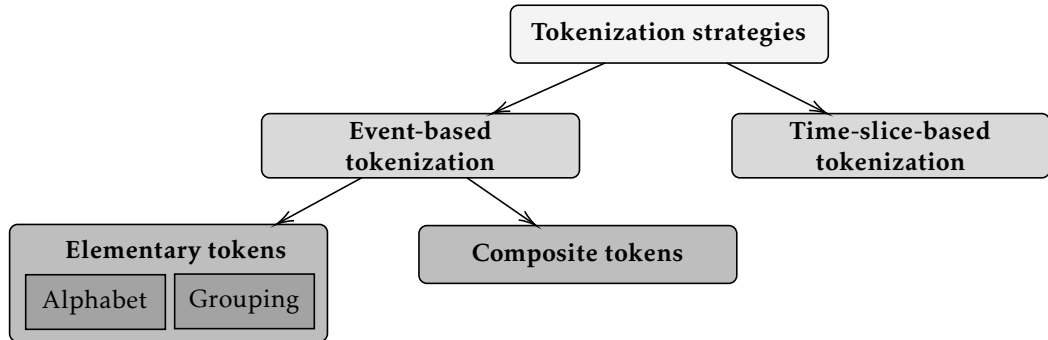


Figure 4.1: Taxonomy of tokenizations for symbolic music. Tokens are either based on regular time-slices or events. Among event-based tokenization strategies, tokens encode various features of these events: composite (or multidimensional) tokens encapsulate all these features in a single token, in contrast with elementary tokens where each musical feature is processed one after the other.

We organize tokenization strategies within two classes: *time-slice-based tokenization* and *event-based tokenization*. Indeed, time plays a special role in music since the time position of notes fundamentally contributes to the conveyed information. Musical events are commonly thought as occurring on an underlying isochronic grid ([Jackendoff, 2009](#)) in which notes have rigorous timings annotated on sheet music. Such exact timings can, however, be altered in a performance context where musicians have the freedom to distort this time grid leading to expressive effects such as *rubato*, *accelerando*, or *ritardando*. Therefore, representing time properties of musical elements has led to multiple approaches ([Briot et al., 2020](#), §4.8) including representations based on regular time steps ([Section 4.1.1](#)) referred to as **time-sliced-**

based tokenization, or driven by events occurring through time (Section 4.1.2) referred to as **event-based tokenization**.

4.1.1 Time-slice-based tokenization

Dividing time at evenly spaced timings is a natural approach to representing music since musical elements such as note onset or durations are notated on scores with specific and regular timings according to particular rhythms. The approaches described in the following section represent symbolic music as a sequence of fixed-time interval tokens.

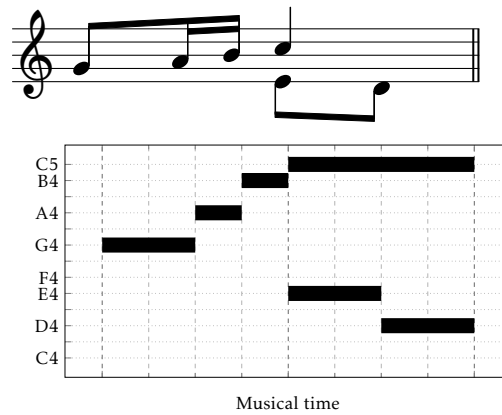


Figure 4.2: Time-sliced based tokenization: piano roll representation. The vertical axis represents the pitches and the horizontal axis represents the time.

DeepBach (Hadjeres et al., 2017) is a model that aims to generate 4-part chorales, for which time is evenly divided at the level of 16th notes. As the number of simultaneous notes is upper-bounded in 4-part chorales, a time step can be represented as a vector containing 4 pitches. In the same way, a concept of “musical words” defined by slices of three beats is proposed (Herremans and Chuan, 2017; Chuan et al., 2020) to model musical context and semantic relationships in polyphonic music. Beyond pitches, this time-slice representation has also shown to be adapted to the context of drum music (Zhang and Callison-Burch, 2023).

More generally, these representations can be seen as specific cases of **piano rolls** (Figure 4.2). A piano roll representation relies on a matrix in which the horizontal axis represents time, and pitches are encoded along the vertical axis, with possible additional characteristics such as velocity as a third dimension. Piano rolls are usually portrayed as an alternative to sequential representations by using matrices. However, a piano roll can be converted into a sequential format by considering it as a sequence of piano roll slices - *i.e.* fixed-size multi-hot vectors representing pitches quantized at a specific duration. In particular, these piano roll slices do not directly capture note onsets: for example, in Figure 4.2, only considering the piano roll without its score transcription, the top C5 could represent either a sustained quarter note

or repeated eighth or sixteenth notes. To address this ambiguity, models based on time-slice sequences often add an extra “replay matrix” (Mao et al., 2018) indicating whether a note in the piano roll corresponds to an onset or a continuation. The actual tokens in these serialized piano rolls can represent a window of slices (Chen and Su, 2019) or a full musical bar (Brunner et al., 2018). From a musical viewpoint, such time-sliced-based tokenization is sometimes required by the task, like Roman Numeral Analysis (Nápoles López et al., 2021; Sailor, 2024), where single notes can be annotated differently through time. This tokenization will be given a more detailed description in Chapter 7, in which we present our work on model explainability for a functional harmony analysis system.

This piano roll representation for music contrasts with text. It is particularly well suited for capturing simultaneous events – *i.e.* musical polyphony, which has no direct equivalent in language – unlike the sequential representations discussed in the following, which induce an artificial sequentiality on simultaneous events.

4.1.2 Event-based tokenization

Unlike time-slice-based tokenization in which tokens are triggered at constant time steps, *event-based tokenization strategies* involve tokens occurring only when a specific event takes place (*e.g.* a note being played, the start of a bar, etc.). Most tokenization strategies have shifted towards this event-centric approach, helped by the large amount of available MIDI data. The MIDI protocol was first developed to handle communication between music software and hardware. The serial transmission of MIDI messages provides a natural way to encode music as sequences of events. The large adoption of this format in the music community has led to the availability of multiple datasets (Ji et al., 2023) which are essential for training deep learning models.

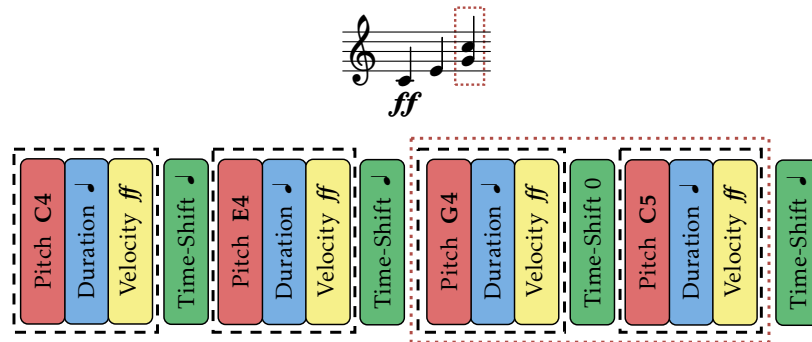


Figure 4.3: Artificial sequentiality possibly introduced in a tokenization strategy (*e.g.* inspired from Structured (Hadjeres and Crestel, 2021)). A note is characterized by musical features such as pitch, duration, velocity, and time-shift. The sequentiality of the blocks (black dashed) follows the temporality, but the order of the inner musical features is arbitrary. The sequentiality of these blocks can even be artificial for simultaneous events (red dotted).

In contrast with characters in text, tokens derived from a MIDI-derived tokenization can have various types, reflecting the multiple features of musical notes such as duration, pitch, or velocity. Since these features characterize a single temporal event, representing such features sequentially may necessitate introducing an “artificial” sequentiality on top of the temporal sequentiality as illustrated in Figure 4.3. This sequentiality is even more artificial when representing *simultaneous* notes by *consecutive* tokens.

We propose to classify these event-based tokenization strategies into two main classes that we refer to as *elementary tokens* (Section 4.1.2.1, Table A.1) and *composite tokens* (Section 4.1.2.2, Table A.2). Sequences of **elementary tokens** explicitly integrate this artificial sequentiality where each token is a single musical feature. This can possibly result in two adjacent tokens describing the same temporal event (e.g. the pitch of a note followed by its duration). On the contrary, sequences of **composite tokens** partly bypass this artificial sequentiality by considering tokens as objects aggregating all the musical features describing a temporal event in a unique “super-token”.

4.1.2.1 Elementary tokens: music as sequence of individual features

The constitutive elements of a sequence composed of musical elementary tokens can be described at two levels: the choices of an initial *alphabet* of atomic elements encoding various musical features and a *grouping* of these atomic elements into higher level elements, presumably more expressive.

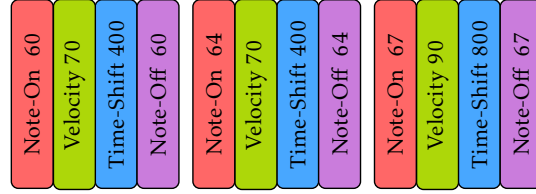
Alphabet – In text, *tokens* frequently denote words or subwords, which themselves are combinations of smaller elements – characters. In symbolic music, *tokens* most often refer to the *atomic* elements of the sequence that constitute what we refer to as an *alphabet*. This alphabet can be composed of a wide range of entities, such as chord labels, notes, inner characteristics of a note (e.g. pitch, duration, etc.), or structural events such as beat position or bars. Thus, choosing an alphabet implies choosing a level at which to describe music and a set of attributes to represent it.

Grouping strategy – Atomic elements can be *grouped* together to form more informative elements. These groupings can be established using fixed-size segmentations, statistically derived groupings, or expert-defined rules. In text, atomic elements (characters) are directly merged together to constitute tokens (words or subwords) leading to a vocabulary of increasing size. Similarly, music atomic elements can be grouped together to enrich the vocabulary with more informative tokens.

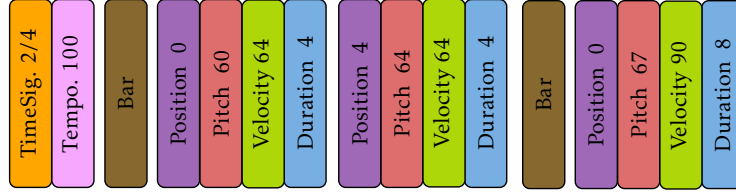
In the following, we present various developed strategies of *alphabets* and *groupings* for symbolic music tokenization. An extensive list of tokenization strategies based on elementary tokens is presented in Table A.1.

Building an *alphabet* of atomic elements to encode music – Symbolic music alphabets can rely on two paradigms: “MIDI Performance” or a “MIDI Score” (Oore

(a) Performance-based tokenization: MIDI-like (Huang et al., 2019).



(b) Score-based tokenization: REMI (Huang and Yang, 2020).



(c) Instrument-specific tokenization: DadaGP (Sarmiento et al., 2021).

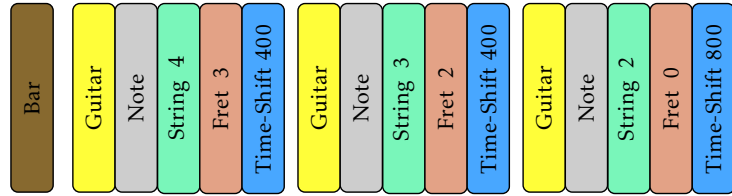


Figure 4.4: Elementary tokens examples. (a) Time can be encoded following a *performance-based* paradigm, where music is represented as successive performance events (<note-on>, <note-off>). (b) Instead, time can be encoded following a *score-based* paradigm, with music is organized along a metrical time grid structured by bars and beats. (c) Event-based tokenization can also be specific to particular instrumental notations, such as guitar tablatures.

et al., 2018). The first one is often a human performance encoded into the MIDI protocol or directly recorded with a MIDI keyboard while the second one is typically a MIDI file converted from a sheet music format (musicXML, kern...) which follows a strict metrical grid. Performance data includes velocity and performance variations such as local tempo or played dynamics whereas scores include information such as musical timings and enharmonics¹. In the following, we follow this distinction to organize existing alphabets for symbolic music tokenization.

On the one hand, *performance-based* tokenization focuses on encoding music as sequences of performance events, nearly translating the gesture of an on-stage performer. The **MIDI-like** tokenization (Huang et al., 2019), represented in Figure 4.4a, follows MIDI events from the basic MIDI protocol, including a vocabulary of 4 event types: <note_on>, <note_off>, <time_shift>, and <velocity>. <time_shift> tokens are typically expressed in absolute timings in milliseconds. This tokenization can be adapted for monophonic melodies (Roberts et al., 2018) or a polyphonic ensemble with a fixed number of instruments (Donahue et al., 2019) by having <note_on/off> tokens specific to each instrument. **TSD** (Time-Shift-Duration) (Fradet et al., 2023a)

¹Enharmonic notes refer to two notes which produce the same sound but are called differently e.g. G \sharp and A \flat .

adapts the MIDI-like tokenization, using `<duration>` and `<time_shift>` to replace pairs of `<note_on/off>`. The **Structured MIDI encoding** (Hadjeres and Crestel, 2021) is similar to TSD but enforces the order of tokens describing a same event. This avoids syntax errors in the context of live music generation and improves token sequence consistency by implicitly reducing the vocabulary size at each generation step. Finally, **Pertok** (Lenz and Mani, 2024) encodes performance timing irregularities using `<micro_time>` tokens. This allows for a balance between high level temporal information encoded via the traditional `<duration>` tokens, and the expressivity of a human performance conveyed via these `<micro_time>` tokens.

In contrast, *score-based* tokenizations describe music as a time-structured system based on multiple discretization levels of time, such as sub-beats, beats, or bars linked to common note values in music notation (e.g. whole, half, quarter, eighth note, ...). Prior to MIDI-based tokenization, early sequential representations of music rely on the various score dimensions (Conklin and Witten, 1995). These representations, called *viewpoints*, can encode properties of the note itself, such as its pitch or its position within a bar, or relations between successive events, such as melodic contours or lengths of the musical phrases. A token is thus a vector constituted of multiple viewpoints which describes each event, in particular each note. **REMI** (Revamped MIDI-derived events) (Huang and Yang, 2020), represented in Figure 4.4b, is derived from a MIDI-based tokenization but employs a set of score-related elements to tokenize musical data, in particular `<bar>`, `<position>` (i.e. beat position within a bar) and `<duration>` both being expressed in musical time instead of absolute timings. The use of such time encoding appears to bring consistency in rhythm. Multiple extensions of REMI have been implemented. Regarding the encoding of the pitch information, developments based on pitch classes and octaves tokens instead of raw MIDI numbers have been proposed for both analysis (Liang et al., 2020a) and generative tasks (Li et al., 2023c). REMI has also been enriched with additional tokens to include metadata (Lee et al., 2022), musical features (von Rütte et al., 2023; Shih et al., 2023), control tokens (Guo et al., 2022), hand positioning for piano music (Gover and Zewi, 2022) or track information (Wu and Yang, 2023b).

In addition, some specificities related to the instrument or the type of music data may prompt the need for adjustments to the tokenization strategy. Tokenization strategies for guitar tablatures have been proposed for generation tasks directly in the tablature space (Chen et al., 2020; Sarmiento et al., 2021) by adding guitar-specific tokens, such as a combination of `<string>` and `<fret>` instead of traditional MIDI numbers, as shown in Figure 4.4c.

Moreover, unlike text in language, which consists of a unique stream of words, the challenge of encoding *multi-track* music (i.e. multi-instrument, with potentially polyphonic tracks) involves identifying a way to represent simultaneous events as a single sequence of tokens. The representations **Multi-Track** (Ens and Pasquier, 2020), the **MMR** (Multi-track Multi-instrument Repeatable) representation (Liu et al., 2022), **REMI+** (von Rütte et al., 2023) deal with this issue by adding a track-related

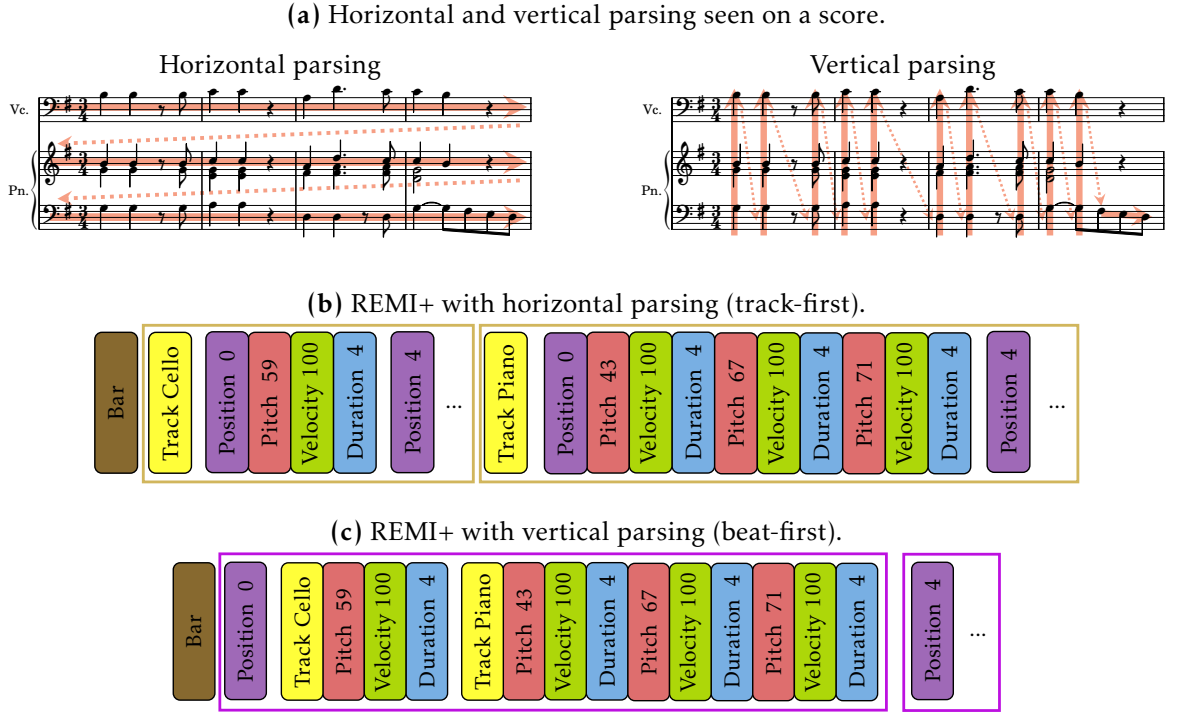


Figure 4.5: Multi-track parsing. In order to represent multi-track music as a *unidimensional* sequence, the multiple tracks can be parsed either horizontally (left) – *i.e.* each track is parsed and concatenated one after each other – or vertically (right) – *i.e.* the music is parsed time-wise, and each beat is concatenated one after each other. A horizontal parsing promotes melodic patterns, while a vertical parsing promotes a “harmonic” reading of the music. A practical example of such parsing paradigm is shown using the REMI+ tokenization (von Rütte et al., 2023).

token in the alphabet. However, MMR and REMI+ interleave the different tracks to represent the multiple tracks into one sequence. Instead, Multi-Track concatenates all the tracks horizontally to get this single sequence. In other words, comparing these multi-track tokenizations as shown in Figure 4.5, Multi-Track has a *horizontal* reading of the score by concatenating single-instrument tracks, while MMR and REMI+ have a *vertical* reading of the score by firstly concatenating simultaneous bars or events from multiple tracks.

Beyond MIDI-derived tokenizations, the **ABC notation** has also been used as a direct way of encoding monophonic scores (Sturm et al., 2016) where tokens are considered to be text characters. Basic NLP models can be simply trained on these textual data for generation (Sturm et al., 2016). With the breakthrough of efficient Large Language Models (LLMs) handling text, this representation has been used for LLM-based text-to-music systems such as ChatMusician (Yuan et al., 2024), MelodyT5 (Wu et al., 2024) or ComposerX (Deng et al., 2024a). As this notation has been originally developed to encode monophonic music, multiple ways of adapting it for multi-track music have been proposed. NotaGen (Wang et al., 2025) proposes an

interleaved ABC notation and MuPT (Qu et al., 2025) implements a “**Synchronized Multi-Track ABC Notation**” (SMT-ABC): these representations groups bars between particular labels (such as a line return for interleaved ABC notation or a $\langle | \rangle$ token for SMT-ABC) between which all the voices are separated by an extra label.

Finally, a few approaches consider encoding music under their **sheet music notations**, beyond their musical content only. Therefore, such must describe each visual element of a score, including tokens for staves, clefs, stem directions, or beam properties (Suzuki, 2022). This approach is denoted as **semantic tokens** by Acosta et al. (2022) who also explore further methods to encode sheet music. In particular, in the same way as ABC notation, they leverage the textual format of MEI format – derived from an XML format – to generate MEI tokens. In addition, they also consider sheet music seen as visual data, either as sequences of **image patches** which are equal size square subparts of the score, or as sequences of **visual words** where each word correspond to a graphical symbol (e.g. a staff beginning, a treble clef, a multi-measure rest sign, ...). In particular, this “visual words” vocabulary is drawn from datasets of atomic score elements observed from optical music recognition research (Calvo-Zaragoza and Rizo, 2018).

Grouping atomic elements for shorter sequences and more informative tokens – When comparing text and music, textual sentences are often composed of hundreds of characters or around a dozen words, which is an amount of tokens that models such as Transformers can handle well. In contrast, musical sequences may be considerably longer, possibly reaching several thousand event-based tokens even in a musical phrase, due to various factors such as polyphony or multiple existing token types. To address this complexity issue, two approaches can be considered: adapting the model mechanisms to handle this type of data (developed further in Chapter 4) or manipulating the representation of music in order to compress the sequence length by *grouping* tokens together.

A textual **n-gram** (Jurafsky, 2000, Chap. 3) is a sequence of n elements (characters, words, etc.) grouped together based on a *fixed number of elements* to constitute a token. N-grams have been one of the earliest representations of music borrowed from NLP (Downie, 1999). However, while grouping characters is straightforward for text data, musical n-grams can be of a diverse nature with groupings occurring at multiple levels. Musical n-grams can be composed of note intervals or rhythm ratios (Wołkiewicz et al., 2008) or musical descriptors called “viewpoints” (Conklin and Witten, 1995) that complement the description of music with characteristics such as pitch contour or intervals. Instead of single notes, chord n-grams can represent music through harmony (Ogihara and Li, 2008). This assumes a choice regarding the restriction of the chord space, which instead, could have lead to a huge vocabulary size when considering all the possible n-gram combinations. These representations appear to be satisfactory for basic analysis tasks, such as composer or style classification. **Skip-grams** (Guthrie et al., 2006) are an improvement of n-grams, allowing a skip of some elements when constructing the n-gram, and thus

extending the scope of the token within the sequence. Therefore, skip-grams can be applied to music to model harmony (Sears et al., 2017) or to discover voice-leading patterns (Sears and Widmer, 2021), confirming musicological observations on harmonic progressions.

These musical n-grams also show statistical phenomena initially observed in text data representations. Various laws such as the Heaps’ law or the Zipf’s law can be observed with musical n-grams. The Heaps’ law states that the length of a text and its inner vocabulary are linked by a power law: this phenomenon have also been observed on music encoded as chroma² vectors (Serra-Peralta et al., 2021), leading to high-level conclusions regarding harmony in terms of vocabulary richness. Similarly, the Zipf’s law states that the word frequencies follow a power decay in relation to the rank of these words. This phenomenon also appears when transcribing music into n-grams (Wołkiewicz et al., 2008; Perotti and Billoni, 2020). This law implies defining or choosing what a musical word is. According to (Perotti and Billoni, 2020), a musical Zipf’s law is therefore obtained when the musical word (or “zipfian unit”) is chosen as a combination of notes and chords instead of single notes.

Musically-informed groupings can be derived from the musical structure of a sequence. The CLaMP model (Wu et al., 2023), which is based on the ABC notation that includes pipe characters to represent bars, considers a bar-based grouping. Though, such musically-informed groupings are little studied because note-level groupings are more suited as composite tokens (Section 4.1.2.2), and higher-level structures, such as motifs or phrases, are often not well defined.

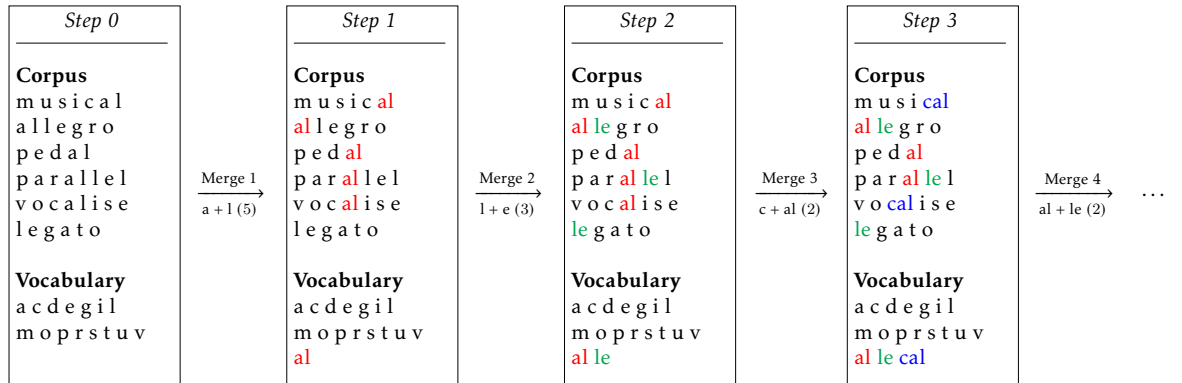


Figure 4.6: Byte-Pair Encoding (BPE) algorithm on text data. The process starts with a vocabulary composed of all characters occurring in the corpus considered as initial tokens. At step 1, the most occurring pair of tokens is ($\langle a \rangle$, $\langle l \rangle$) which occurs 5 times: they are merged and $\langle al \rangle$ is added to the vocabulary. At step 2, a new token $\langle le \rangle$ is added to the vocabulary. At step 3, a single character can be merged with an already created token ($\langle c \rangle$, $\langle al \rangle$) creating a new token $\langle cal \rangle$ in the vocabulary. The algorithm then continues for a number of merges.

Finally, NLP studies have developed *subword tokenization* methods (Mielke et al.,

²A chroma is a pitch class *i.e.* C, C \sharp , D, ..., B

2021) where a vocabulary of subwords is statistically learned on a training corpus. These include **Byte-Pair Encoding (BPE)** (Gage, 1994; Sennrich et al., 2016), **WordPiece** (Schuster and Nakajima, 2012) or **UnigramLM** (Kudo, 2018). Some of them have been adapted for music to create musical subwords as tokens. The BPE algorithm is adapted for orchestral data (Liu et al., 2022) by exploiting the invariance of note order within a chord, to shorten sequence lengths. More than a simple tool for shortening sequences, BPE has also been studied for its specific effects on musical data. Multiple studies applied it on multiple encodings in order to examine how training Transformer models with input reduced by BPE affects both generation and analysis tasks. Although BPE builds a more structured embedding space (Fradet et al., 2023a), experiments studying the impact of BPE in music analysis tasks do not show a significant increase in performance (Zhang et al., 2023), unlike BPE applied to text (Sennrich et al., 2016). In a more restricted musical context, **Mel2Word** (Park et al., 2024) implements BPE with monophonic tunes and enables the retrieval of style-specific motifs. Finally, UnigramLM subword tokenization is also specifically evaluated on music generation, applied to score-based music and guitar tablatures (Kumar and Sarmiento, 2023). Their findings indicate that both approaches contribute to improved data representation, enhance the structural quality of generated music, and enable the generation of longer sequences.

In this thesis, we further explore technical aspects of these event-based tokenization strategies (Chapter 6) to improve the expressiveness of these representations. Regarding the *alphabet* choice, we study the impact of choosing interval tokens instead of absolute pitch for pitch encoding on analysis tasks. In addition, in terms of *grouping*, we then analyze the different behaviors of BPE in text and music, and we focus on their impact on analysis tasks involving monophonic or polyphonic music.

4.1.2.2 Composite tokens: music as sequence of feature combinations

While sequences of elementary tokens need to introduce an artificial sequentiality by ordering musical features that describe a single event, *composite tokens* encapsulate the entirety of a temporal event by combining all its musical features into a single *super-token*. Considering a single token as an agglomerate of multiple musical features also helps shortening sequence lengths, but may also involve model adaptations to handle this particular vocabulary. The choice of the type of super-tokens, the musical features encapsulated within them, and the method used to construct the vector representing each super-token are the key variables in the approaches reviewed below. Figure 4.7 gives examples of possible composite tokenizations and Table A.2 presents an exhaustive list of approaches relying this type of tokenization.

On the one hand, *homogeneous super-tokens* denote a representation where each super-token contains the same set of features no matter the nature of the event it describes. The representation developed by Zixun et al. (2021) is based on the concatenation of multiple one-hot vectors describing pitch, duration, chords, and

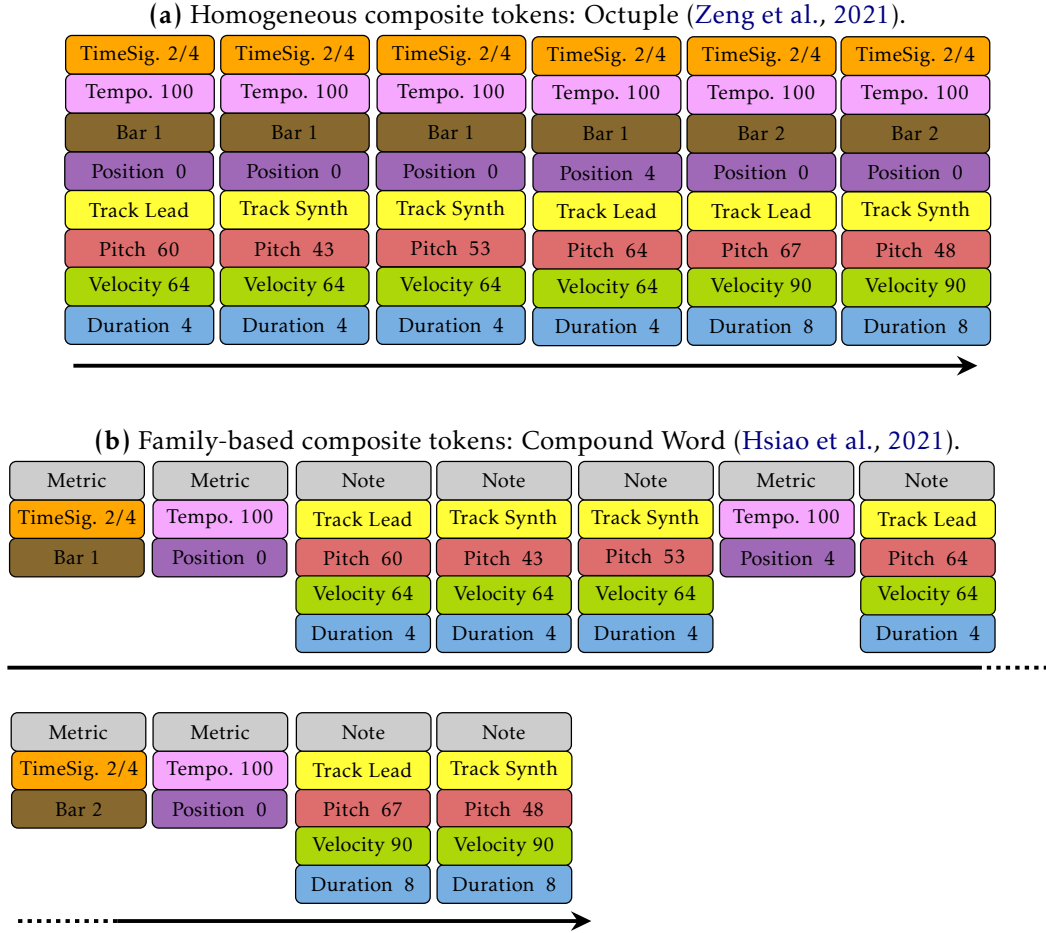


Figure 4.7: Composite tokens examples. These tokens can be homogeneous (a), where each token consistently encapsulates the same set of elements. In contrast, family-based tokenizations (b) distinguish tokens derived from note events or structural events.

bar. **Octuple** (Zeng et al., 2021) is instead based on the embedding of 8 musical features (*i.e.* time signature, tempo, bar, position, instrument, pitch, duration and velocity) which are concatenated to form the single vector considered as token representing a single note. Such homogeneous representations are also used by **PiRhDy** (Liang et al., 2020a) encoding pitch classes and octave instead of MIDI value, and **MMT** (Dong et al., 2023) for multi-track music. Instead of vectors, **MuseBERT** (Wang and Xia, 2021) embeds matrices constructed from onsets, pitches, and durations to describe both musical attributes with their relations. Beyond notes, the **Chordinator** model (Dalmazzo et al., 2024) encodes chords described by a root, a nature, extensions, and a set of notes composing the chord.

On the other hand, *methods separating events by families* have been developed to highlight the distinction between note events and structural events such as the beginning of a bar. For polyphonic music, **MuMIDI** (Ren et al., 2020) represents a token as a sum of the embeddings of bars, position, and tempo, and note characteris-

tics (*i.e.* pitch, duration, velocity) are added for note events. Similarly, **Compound Word** (Hsiao et al., 2021) gathers tokens into two families: event-related or note-related and concatenates these embedded atomic elements to build the token. It has also been adapted for a task of drum accompaniment generation (Makris et al., 2022). This representation is also enhanced by Di et al. (2021) in the context of video-to-music, by incorporating a token family related to rhythm, encapsulating rhythm density and strength. **Unsupervised Compound Word** (Tian et al., 2024) is based on the original Compound Word tokenization and includes Byte-Pair Encoding which learns the atomic element groupings instead of relying on predefined families resulting in variable-length composite tokens. **REMI_Track** (Luo et al., 2024) improves REMI+ and also combines composite tokens and BPE. Tokens are defined as 5-long vectors, with note-related elements of a token (pitch, velocity, duration) being possibly grouped together under a BPE-element, which has shown to improve inference efficiency when generating long sequences.

4.2 Comparing tokenization strategies

Various tokenization methods naturally lead to various performance depending on tasks or data. In NLP, different tokenizers, which initially aim at segmenting text, can result in different vocabularies, so that they can result in unequal performance on various tasks or languages (Domingo et al., 2023). The choice of the tokenization strategy is even more fundamental in non-space-segmented languages such as Korean (Park et al., 2020). Few studies have conducted such comparisons between tokenization strategies in MIR contexts.

Multiple strategies for pitch (pitch-class vs. absolute) and time grid (time resolution) encodings are compared in the context of monophonic music generation (Li et al., 2023b). Using a set of objective metrics, it appears that pitch representations using pitch class and octave outperform MIDI-event representations and higher temporal resolution can contribute to the performance of the model, while being more inclined to overfit, resulting in worse beat- or bar-level modeling. Fradet et al. (2023b) focus specifically on time encoding by comparing note positioning (<time-shift> vs. <bar> + <position>) and duration encoding (<note-on> + <note-off> vs. <duration>) on generative, classification, and representation tasks. While being highly dependent on the task, explicit time information in tokenization strategies appears to be helpful.

Beyond only considering tokenization, a comparison between matrix, graph, and sequence representations of symbolic music is performed on analysis tasks (Zhang et al., 2023). When examining sequence representations, non-embedding-grouping tokenization strategies such as REMI or MIDI-like appear to provide better performance than embedding grouping strategies in the context of analysis tasks. When comparing data representation modalities, matrix and sequence representations appear to perform better on MIDI performance compared to score data.

These studies most often focus on how tokenization affects model performance on downstream tasks. However, the choice of tokenization also influences the resulting sequence length, which in turn affects the suitability of different models, depending on their capacity to process longer or shorter sequences, as well as the data used to train them. Consequently, improving the expressiveness of tokenization strategies is a direction we have chosen in one of our contributions, as detailed in [Chapter 6](#).

More technically, multiple libraries have been developed to process symbolic music data under sheet music formats, such as `musicXML` or `**kern`. Among them, **Music21** ([Cuthbert and Ariza, 2010](#)) and **Partitura** ([Cancino-Chacón et al., 2022](#)) offer object-oriented data structures for representing symbolic music, which are less inherently adapted to direct sequential processing. For sequential representations, multiple libraries are available to handle MIDI data, such as **Pretty MIDI** ([Raffel and Ellis, 2014](#)) or **MidiToolkit**³, with the main difference being the first one handles time in seconds while the former considers MIDI ticks instead. These libraries allow for sequential processing but does not extend beyond the MIDI-like tokenization strategy. More recently, the **MidiTok** Python package ([Fradet et al., 2021](#)) has been developed to provide a consistent interface for handling multiple tokenization strategies. It currently supports around a dozen encoding methods, including some of those described above, and offers various tools designed to manipulate sequential symbolic music data, such as data augmentation or BPE. Multiple other tokenizers derive from this library, including a `musicXML` tokenizer ([Zhang et al., 2023](#)) or a component integrated into a processing pipeline coupled with the **HuggingFace** library ([Kumar and Sarmiento, 2023](#)), initially developed to handle text tokenization and models. Similarly, **Musicaiz** ([Hernandez-Olivan and Beltran, 2023](#)) offers a tokenization framework, with extensive visualization, generation, and analysis frameworks for symbolic music.

These tools allow for a easy manipulation of tokenization strategies, aiming at comparing them, starting with a simple comparison of musical token sequence sizes ([Fradet et al., 2023a](#)). However, objective and unbiased studies aiming at evaluating the choice of a tokenization in specific tasks still lack and represent a main future direction for the MIR community ([Section 9.1](#)).

4.3 Embedding music tokens for model processing

The previous sections describe music encoded as sequential elements and operations that can be applied to them while keeping their high-level musical meaning. When used as inputs of most machine learning models, these elements need to be *embedded* or converted into numerical values so that the model can process them. Text, sub-words, words, or documents need to be projected into a particular space in order

³<https://github.com/YatingMusic/miditoolkit>

to be processed (Li and Yang, 2018) leading to multiple distributional vector space models and embedding methods.

Earliest word representations simply relied on basic one-hot vectors, each with a length equivalent to the vocabulary size. A document is represented by summing all these word vectors, leading to a co-occurrence counts vector, also called **Bag-of-words (BOW)** (Jurafsky, 2000, Chap. 4). This representation is improved by **Term Frequency-Inverse Document Frequency (TF-IDF)** (Jurafsky, 2000, Chap. 6) that takes into account the total number of documents in which a word appears. In symbolic music, such BOWs or TF-IDFs have been implemented for music similarity analysis (Wołkiewicz and Kešelj, 2012), mode classification in Gregorian chant (Cornelissen et al., 2020), Chinese folk music clustering (Zhang and Jiang, 2021), or guitar chord difficulty prediction (Vásquez et al., 2023).

However, these approaches do not capture any sequential information and the resulting space is often sparse, preventing the ability to capture possible proximity between musical elements. Therefore, multiple methods have been developed in the NLP field aiming at representing words as vectors in a dense and continuous space including static and contextual embeddings.

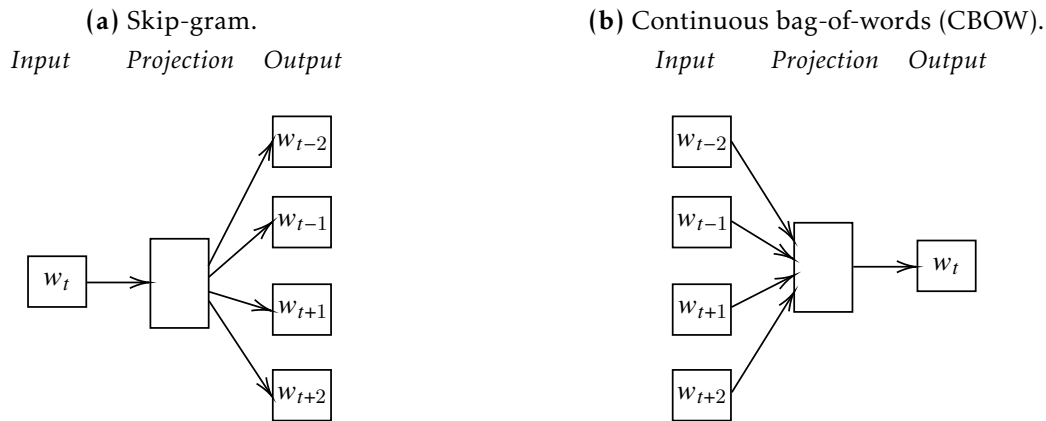


Figure 4.8: Word2Vec training paradigms in text (Mikolov et al., 2013). Word embeddings can be learned following a (a) *skip-gram* approach, which predicts surrounding words given a single word, or a (b) *CBOW* approach, which predicts a single word given a surrounding context.

Static embeddings assume that each word can be encoded using the same vector regardless of the surrounding context in which the word occurs. **Word2Vec** (Mikolov et al., 2013) is based on a shallow neural network that builds such static embeddings. This model is often composed of only one hidden layer and is trained to learn word associations. As represented in Figure 4.8, word embeddings can be learned following two paradigms: *skip-gram*, which predicts surrounding words given a target word, and continuous bag-of-words (CBOW), which predicts a target word from its surrounding context. The model learns vector representations where words with similar occurrence contexts are placed close together in the vector space. This

method has been adapted for music, implicitly leading to multiple interpretations of the definition of a “musical word”, including chords or musical phrases.

Multiple chord-based Word2Vec have been developed (Madjiheurem et al., 2016; Huang et al., 2016). Such chord embeddings exhibit musical relations (e.g. the circle of fifths), and are evaluated on downstream tasks like chord prediction and composer classification (Lahnala et al., 2021). **PitchClass2Vec** (Lazzari et al., 2023) embeds chords with Fasttext (Bojanowski et al., 2017) which relies on subwords instead of words. In particular, instead of embedding the whole set of pitches constituting a chord, Pitchclass2vec decomposes the chord as intervals in the same way as Fasttext breaks words into n-grams.

An alternative approach, as opposed to embedded chords, is to consider words as temporal chunks of music. **Melody2Vec** (Hirai and Sawada, 2019) uses Word2Vec on monophonic melodies by assuming such words as musical phrases segmented by GTTM rules (Lerdahl and Jackendoff, 1996). Word2Vec has also been adapted for polyphonic music (Herremans and Chuan, 2017), by considering words as equal-length and non-overlapping slices of polyphonic music. Visualizing these embeddings shows a structure and organization of the space that follows the rules of tonal harmony (Chuan et al., 2020). Considering distance between these slices allows for a task of melody replacement by exchanging slices of an original piece by a close slice in the embedding space.

Unlike static embeddings, *contextual embeddings* represent a same word with different vectors depending on the context in which the word occurs because of the polysemous nature of words. The same word can indeed appear in multiple contexts, carrying significantly different meanings (e.g. “bank” can refer to a financial institution or as the side of a river). Although polysemy and semantics are not directly applicable in music, these contextual embeddings can be useful for symbolic music because the context in which a note appear is fundamental, for instance in functional harmony (i.e. where chords are identified by their function relative to an overall tonality).

Technically, contextual embeddings are built concurrently with model training, such as recurrent or attention-based models described in Chapter 5. Yet, while analyses of learned contextual embeddings are numerous in NLP (Liu et al., 2020), only very few studies have specifically observed the contextual aspect of such embeddings when applied to symbolic music. Such contextual embeddings have been analyzed from BERT embeddings (Han et al., 2023) or from an LSTM model (Garcia-Valencia, 2020). In the same way as text embedding spaces can show semantic relations (Wiedemann et al., 2019), its resulting embedding space notably shows musical characteristics such as interval relations between pitches or a space structure based showing alterations and natural pitches. Fradet et al. (2023a) has shown that the learned contextual embedding space from BERT is more structured than the one learned from GPT-2. Musical context can also be defined by the relationship between simultaneous elements, extending beyond the typical temporal context encoded by

classic contextual embeddings. **PiRhDy** embeddings (Liang et al., 2020a) encode such musical-specific context encapsulating melodic and harmonic contexts.

The choice of a tokenization strategy followed by an embedding method can significantly impact how they are processed by the *model* performing the task. Therefore, beyond the developments of representations of music as sequence, the widespread adoption of NLP methods in symbolic MIR is primarily motivated by the apparent effectiveness of these models, such as Transformers, to process text data.

Chapter 5

NLP-based models for symbolic music processing

5.1	Shallow models	54
5.2	Sequential neural models	55
5.2.1	Neural networks	55
5.2.2	Recurrent models	57
5.3	Attention-based models	61
5.3.1	Training paradigms	64
5.3.1.1	End-to-end models	65
5.3.1.2	Pre-trained models	66
5.3.2	Model architecture	67
5.3.2.1	Encoder only	67
5.3.2.2	Decoder only	68
5.3.2.3	Encoder-decoder	69
5.3.2.4	Multimodal models	70
5.3.3	Adapting attention models to symbolic music	71

In order to perform the described tasks in [Chapter 3](#), symbolic music encoded into a chosen representation ([Chapter 4](#)) is processed by *models*. This chapter reviews such *models* that have been borrowed or inspired from NLP and adapted to address MIR tasks.

This transfer primarily arises from the temporal nature of music, which facilitates its representation as sequences of elements, thus facilitating its processing by NLP-based models, which are mostly data-driven. Historically, *shallow* machine learning models were prominent for many years in NLP. A shift from shallow models to deep learning models began in the 1990s. In particular models based on recurrent cells, like Recurrent Neural Networks (RNNs), became widely popular for NLP tasks. This

trend based on deep learning models continued with the breakthrough of attention-based models in the mid-2010s. MIR studies also followed these trends, adapting these models to symbolic music in various ways.

In this chapter, we organize the overview of NLP models for MIR by following a historical timeline, starting with rule-based and shallow models (Section 5.1) and sequential models, particularly recurrent models (Section 5.2). A substantial part is then dedicated to attention-based models (Section 5.3), which currently represent the state of the art in a variety of MIR tasks. These attention-based models are examined through multiple lenses, including their training paradigms, architecture designs, and the internal mechanisms originally developed for language and adapted to symbolic music.

5.1 Shallow models

Prior to the widespread adoption of data-driven methods, natural language modeling was mostly addressed by rule-based systems, such as formal grammars. A **formal grammar** is a set of rules that defines the syntactic structure of sentences in a language, specifying how words and phrases can be combined to form grammatically correct sentences. They are used in text for syntactic parsing or semantic analysis such as dependency parsing, representing text as tree structures. Musical grammars (Roads and Wieneke, 1979) have also been formalized, in particular based on harmony, for tasks such as jazz chord analysis (Steedman, 1984). Generative grammars (Chomsky, 1957), aiming at generating sentences based on rules, have also been applied in music, in particular with GTTM (Lerdahl and Jackendoff, 1996).

Such grammars are often used in conjunction with shallow sequential models. **Hidden Markov Models (HMMs)** and **Conditional Random Fields (CRFs)** are sequential models that were applied to NLP tasks much earlier than symbolic music. HMMs rely on the assumption that each observed element of a sequence is the result of a hidden process with the Markov property (short span dependencies). As a generalization of HMMs, CRFs are discriminative models that can impose dependencies on arbitrary elements of the sequence. In NLP, HMMs and CRFs have been implemented for part-of-speech tagging (Kupiec, 1992), named entity recognition (McCallum and Li, 2003) or text classification (Frasconi et al., 2002). These models have then been widely used in early MIR studies for various symbolic music tasks such as style classification (Vercoe, 2001), melody prediction (Sentürk and Chordia, 2011), harmonization (Groves, 2013), generation (Van Der Merwe and Schulze, 2011), chord recognition (Masada and Bunesco, 2017) or key detection (Nápoles López et al., 2019).

Neural networks, in particular deep neural networks, have since demonstrated greater performances, notably due to the increase of available computational power that enable the processing of large datasets for training more complex data-driven

models. This leads to architectures based on sequential neural networks that offer an alternative way of modeling time and therefore handling sequential data.

5.2 Sequential neural models

As described in [Chapter 4](#), symbolic music can be encoded using sequential representations. Therefore, as sequential text representations can be processed by sequential models, symbolic music can likewise be processed by similar models to perform particular tasks. Sequential models are typically based on neural networks ([Section 5.2.1](#)) and are adapted with recurrent mechanisms ([Section 5.2.2](#)) in order to handle sequential inputs so that each element of the input is successively processed while incorporating information from previous elements. An exhaustive summary of such sequential models is presented in [Table B.1](#).

5.2.1 Neural networks

Artificial neural networks are a class of machine learning objects designed to approximate a function f , which maps an x into an output $y = f(x)$, in a data-driven manner given a large set of couples (x, y) . Historically, these artificial neural networks take inspiration from human neural systems, modeled as sets of *neurons* ([McCulloch and Pitts, 1943](#)). Each neuron is characterized by a set of weights $[w_1, \dots, w_n]$, a bias b , and a non-linear activation function $\sigma(\cdot)$. Given an input $[x_1, \dots, x_n]$, a neuron processes this input to produce an output z following:

$$z = \sigma \left(\sum_{i=1}^n w_i x_i + b \right)$$

These neurons can be stacked into interconnected *layers*, organized into input layers, hidden layers and output layers. The number of neurons per layer is part of the model's architectural design, or can also be determined by the nature of the task – for example, a n -class classification task implies n neurons in the output layer, or a unidimensional regression task typically involves a single output neuron. The entire network gathering these layers is called a *fully connected network* ([Figure 5.1](#)). The number of layers can then determine whether the network can be considered as a *deep neural network*.

As presented above, the primary objective of the entire network is to approximate a target function. To achieve this, it must be *trained* on datasets containing a large number of input-output pairs, allowing the network to learn the mapping from inputs to their corresponding expected outputs. To this end, the model is trained to minimize a *loss function*, which quantifies the difference between the model's

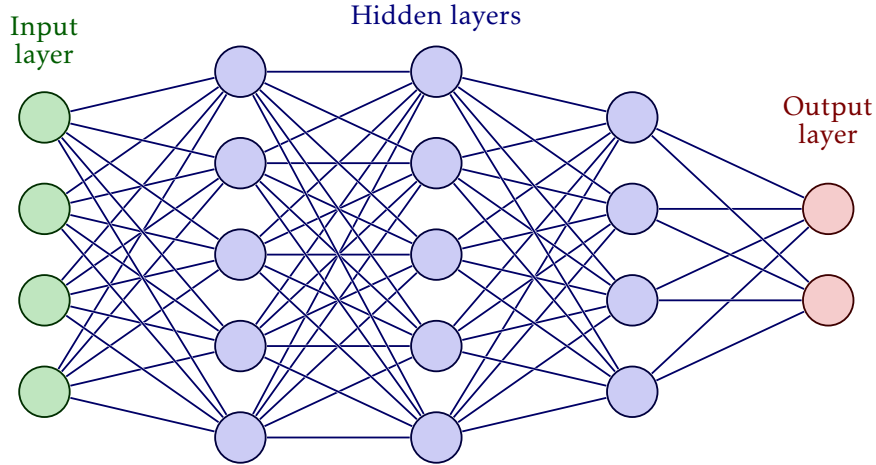


Figure 5.1: Full connected neural network composed of a 4-neuron input layer, three hidden layers and a 2-neuron output layer.

predicted outputs for a given input and the corresponding ground truth from the data. Typically, such loss function can be chosen as a mean squared error for a regression task, or a cross-entropy in the case of classification tasks.

The parameters (*i.e.* weights and biases) of the network are optimized to minimize this loss function L through *back-propagation* using *gradient descent* to update the model's parameters (Rumelhart et al., 1986). More formally, given a weight w and a chosen hyperparameter η , called *learning rate*, this weight can be updated following:

$$w := w - \eta \cdot \frac{\partial L}{\partial w}$$

This process can be repeated for each input-output pair from the datasets. This can then be repeated for a certain number of iterations, called *epochs*. However, this training process can lead to overfitting (*i.e.* a model learns the training data too precisely, including its noise and outliers, leading to poor generalization on unseen data.). Thus, the data is typically divided into a *training* set and a *validation* set, and possibly a *test* set. The validation and test sets are excluded from the training data: the validation set is used to assess the model's performance on unseen data *during* training and helps prevent overfitting, while the test set is used to evaluate the final and fully trained model. Further techniques can improve the model's training, such as regularization (Krogh and Hertz, 1991), dropout (Srivastava et al., 2014), batch normalization (Ioffe and Szegedy, 2015), early stopping, or data augmentation.

Several classes of artificial neural networks have been implemented, such as convolutional neural networks, based on convolutional and pooling layers. These are typically applied in image processing tasks, where images are represented as matrices. Because text and music can often be represented as sequential data, a common model employed in NLP and MIR has been *recurrent neural networks*.

5.2.2 Recurrent models

Recurrent Neural Networks (RNNs) (Rumelhart et al., 1986) are a class of artificial neural networks designed to process sequential data by maintaining an internal memory. Beyond the standard components of fully connected networks – namely input, hidden, and output layers – RNNs are characterized by their *recurrent connections* within the hidden layers. Unlike traditional feedforward networks presented in the previous section, recurrent connections involve a hidden state $h^{(t)}$ at time t which is passed and updated through time. This hidden state $h^{(t)}$ depends both on the previous hidden state $h^{(t-1)}$ and the current input $x^{(t)}$. This recurrence is controlled by learned parameters matrices W_{hh} and W_{hx} , which are shared across all time steps:

$$h^{(t)} = \sigma_1 \left(W_{hh}h^{(t-1)} + W_{hx}x^{(t)} + b_h \right)$$

At each time step, these hidden states are then used to compute an output $y^{(t)}$ through a linear transformation defined by the weight matrix W_{yh} :

$$y^{(t)} = \sigma_2 \left(W_{yh}h^{(t)} + b_y \right)$$

In order to train a recurrent model, parameters are updated through a process called *back-propagation through time*, which propagate the gradient of the loss both backward through the sequence of time steps and across the layers of the network.

This basic recurrence mechanism can be further refined to address particular issues, such as capturing long-term memory which lead to improved recurrent units such as Long-Short Term Memory (LSTM) or Gated Recurrent Unit (GRU), as detailed in the following paragraphs.

RNNs are widely used in NLP and other domains involving sequential dependencies, such as time series forecasting. Depending on the nature of the task, RNNs, and more generally sequential models including Transformers, can be implemented in various configurations (Figure 5.2):

- One-to-many (5.2a): A single input leads to a sequence of outputs (e.g. free generation, where a single input token can initiate the model and each generated output is used as input for the next steps).
- Many-to-one (5.2b): A sequence of inputs is used to predict a single output (e.g. author classification or sentiment analysis)
- Many-to-many (or *sequence-to-sequence*): A sequence of inputs produces a sequence of outputs. This setup includes two cases:
 - Equal-length input and output sequences (5.2c), such as in named entity recognition or part-of-speech tagging.

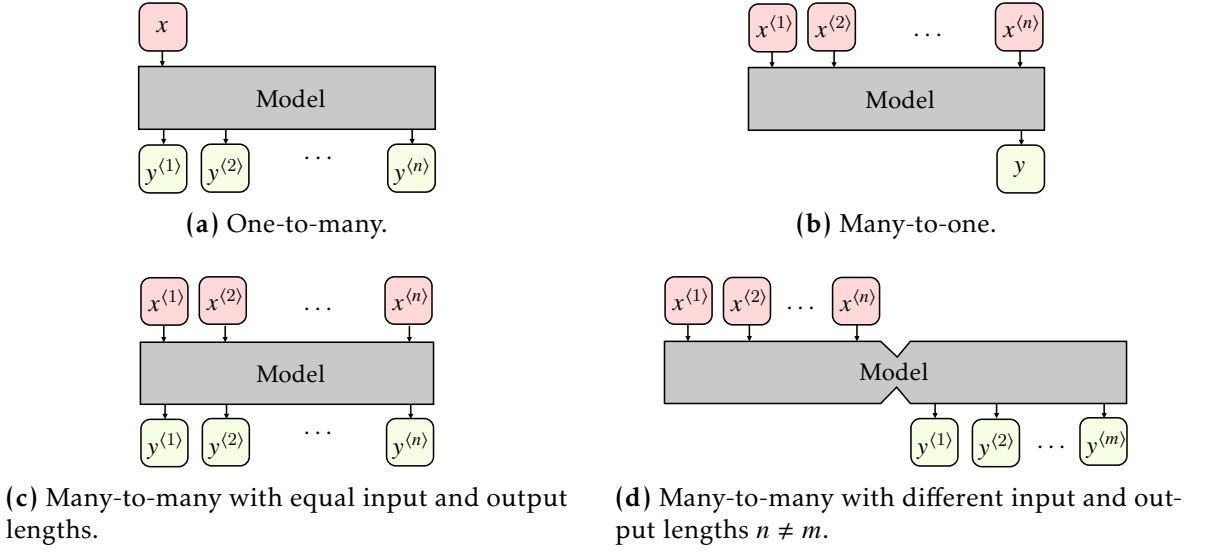


Figure 5.2: Configurations of sequential models according to the input and output sizes.

- Different-length input and output sequences (5.2d), as seen in text summarization or machine translation.

In MIR, only a few studies used basic RNN models, such as RNN-RBM (Boulanger-Lewandowski et al., 2012) or RNN-DBN (Goel et al., 2014) combining RNN, Restricted Boltzmann Machine, and Deep Belief Network for polyphonic music generation. Such RNNs however have shown to suffer from the issue of vanishing gradient occurring with long sequences, which is often the case in symbolic music, as mentioned in Section 4.1.2.1.

Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) have been developed to specifically address this issue. Alongside with the hidden states, an additional cell state $c^{(t)}$ is passed through time steps which serves as the long-term memory of the network by mitigating the vanishing gradient problem. This cell state is modified through three gates – a “forget gate”, an “input gate” and an “output gate” – each playing a distinct and interpretable role in controlling the cell state. It is then used to compute the updated hidden state for the next time step. An other improvement of recurrent networks then emerged with the introduction of **Gated Recurrent Units (GRUs)** (Cho et al., 2014). Compared to LSTM models, GRUs are based on a simpler architecture, in particular by merging the forget and input gates into a single “update gate”. This reduces the total number of parameters and consequently shortens training time, while maintaining similar performances to LSTM (Chung et al., 2014).

Multiple studies in MIR have implemented these models for analysis tasks with a bi-directional LSTM performing harmonic analysis (Chen and Su, 2018). These recurrent models are often used as baselines or model benchmarks to compare multiple methods on an analysis task, such as melody extraction (Kosta et al., 2022)

from a polyphonic texture, emotion classification (Hung et al., 2021), or genre classification based on LSTMs (Angioni et al., 2023) or GRUs (Kong et al., 2020).

For generative tasks, constraints enforced to the generated content can influence the choice or design of the model architecture. For music infilling, Hadjeres and Nielsen (2017) proposes an LSTM-based model composed of a “token-RNN” and a “constraint-RNN” operating on the sequence from opposite directions. Similarly, such model based on two opposite directions is also used for chorale generation and harmonization (Hadjeres et al., 2017). Long-term memory of LSTMs is often leveraged to ensure long-term consistency, whether in terms of structure (Medeot et al., 2018) or harmony for folk melodies (Chen et al., 2019) or jazz (Trieu and Keller, 2018). Since recurrent models were first applied to text data in NLP, MIR studies based on RNNs have also taken advantage of this textual representation to process music as chords (Choi et al., 2016) or as ABC notation (Sturm et al., 2016). These models can also be parts of multimodal systems, which process both text and music for tasks such as lyrics-conditioned generation based on LSTMs (Yu et al., 2021) or GRUs (Bao et al., 2019) which aim at generating monophonic melodies that match given lyrics. In between music analysis and conditioned generation, BUTTER (Zhang et al., 2020) is a multi-purpose GRU-based model that processes separately music and textual data for text-based music query and generation. In particular, this latter model also fits within the category of multi-agent models. Recurrent layers can serve as agents within a broader architecture, allowing a complex task to be decomposed into smaller sub-tasks each performed by dedicated sub-models. For instance, in drum accompaniment generation, Makris et al. (2019) introduces a model in which the drum set is divided into three main components, with each part generated by its own specialized sub-model. JamBot (Brunner et al., 2017) is an end-to-end chord-conditioned model consisting of a chord generator followed by a polyphonic LSTM aiming at generating melodies that match the generated chords. Going one step further, XiaoIce Band (Zhu et al., 2018) performs a chord-conditioned generation task through a GRU-based model, followed by a multi-instrumental arrangement task for pop music.

Beyond being used as vanilla models, recurrent layers are often part of larger classes of architectures such as Generative Adversarial Networks (GANs) or Variational Auto-Encoders (VAEs). These particular architectures are not limited to recurrent models and can typically be used with attention layers (Section 5.3). The recurrence mechanism might also be improved through various approaches such as reinforcement learning or attention.

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) can include recurrent layers as part of their components. This architecture consists of a generator and a discriminator which are trained simultaneously through adversarial training to generate realistic data. Specifically, the generator learns to produce increasingly realistic samples, while the discriminator is trained to distinguish between real data and the generator’s outputs. LSTM-based GANs have typically been developed for

chord-conditioned generation (Trieu and Keller, 2018) or lyrics-conditioned melody generation (Yu et al., 2021).

Variational Auto-Encoders (VAEs) (Kingma and Welling, 2013) are another type of generative model. They are trained to encode and decode data in a probabilistic way, allowing for the generation of new samples while capturing the underlying structure of the input data. VAEs’ architecture consists of an encoder and a decoder connected through a latent space, where the encoder maps input data into a probabilistic distribution. The decoder then generates content by balancing accurate reconstruction of the input with regularization of the latent space¹. With recurrent layers, such architecture can involve LSTM or GRU layers for generative purposes (Roberts et al., 2018), with specific tasks or styles such as style transfer (Brunner et al., 2018), orchestral music generation (Lousseief and Sturm, 2019), drum generation (Gillick et al., 2019), or Chinese folk song generation (Luo et al., 2020). Beyond their generative use, the learned latent space of VAEs can also be analyzed revealing particular directions representing musical aspects such as speed or repetitiveness (Turker et al., 2022), note and chord relations (Wang et al., 2020c), or higher level concepts such as music genre (Valenti et al., 2020) in the same way as text VAEs can highlight semantic relations in language (Deudon, 2018).

Hierarchical RNNs (Chung et al., 2017) are designed to capture dependencies across multiple temporal or structural scales—two key characteristics of both symbolic music and text. This architecture typically consists of several stacked RNNs, where each layer operates at a different temporal or structural resolution. Therefore, such a model has been widely implemented in NLP in topic-conditioned generation (Guo et al., 2020) or translation (Su et al., 2018). In MIR, this hierarchical property is exploited for structured-conditioned generation tasks (Wu et al., 2020a), with multiple levels of granularity (Jeong et al., 2019a) or chord-conditioned generation (Zixun et al., 2021). The hierarchical property of such architecture can also be leveraged to encode music theory concepts at multiple levels, then used to condition pop music generation (Chu et al., 2016).

Recurrent layers have also been employed in models trained on symbolic MIR tasks using other paradigms, in particular **reinforcement learning**, based on a model trained to make decisions by interacting with an environment by rewarding or penalizing it. In NLP, reinforcement learning is particularly used in conversational systems (Uc-Cetina et al., 2023). The choice of these rewards are often based on musical rules, such as pitch entropy or chords (Kumar and Ravindran, 2019) or note intervals and repetitiveness (Jin et al., 2020). Beyond reinforcement learning, evolutionary computation is also explored with an LSTM-based model trained with a genetic algorithm to generate ABC notation (Farzaneh and Toroghi, 2020).

Beyond the architecture and the training paradigm, inner mechanisms aiming at improving basic recurrent models have been proposed and adopted in MIR. This

¹An extensive description of VAEs is presented in Section 8.1.2, where we use this architecture for a re-orchestration task.

includes **attention** (Bahdanau et al., 2015) which aims at giving different weights of importance to the elements of the processed sequence. In particular, it has been initially implemented for encoder-decoder models, so that the decoder can attend to all encoder hidden states for each generated element in the output sequence. Attention in NLP can be implemented on most of tasks, in multiple model architectures, and can be computed through multiple attention functions (Galassi et al., 2021). This mechanism can be used with symbolic music for enhancing overall coherence in a multi-track arrangement task (Zhu et al., 2018) or enforcing temporal structure (Jeong et al., 2019a), in particular to help building a melody with a long-term structure (Waite, 2016).

By the late 2010s, Transformer models (Vaswani et al., 2017) marked a major breakthrough in the field of NLP. These models can also take as input sequential data, but does not rely on recurrent mechanisms, leveraging attention mechanisms instead. As several state-of-the-art models, in both NLP and MIR, are now built upon this architecture, a substantial part of this review part is dedicated to such models.

5.3 Attention-based models

Attention is a mechanism proposed by Bahdanau et al. (2015), initially as an improvement of encoder-decoder RNNs (Section 5.2). In this architecture, the decoder relies solely on the final hidden state resulting from the encoder, which is expected to encapsulate the entire input sequence and may act as a bottleneck. Indeed, this final state may not represent all parts of the input equally: information from earlier elements of the sequence, in particular, might be significantly diluted. Instead, the attention mechanism aims at *explicitly* assigning a weight, called *attention weight*, to each element of the input sequence allowing the model to compute each element of the output sequence by focusing more or less on different parts of the input.

Presentation of Transformers – Vaswani et al. (2017) introduced *Transformers* showing that a model based solely on attention – without using any recurrent mechanism – can outperform state-of-the-art results in NLP. More precisely, Transformers are based on a *self-attention* mechanism and *multi-head attention* blocks (Figure 5.3). The **self-attention mechanism** is similar to the original attention mechanism in that it aims at representing a token sequence by incorporating information from all tokens in the sequence, but improves upon it by eliminating the need for recurrent mechanisms. More formally, considering the embedded input sequence $X \in \mathbb{R}^{n \times d}$ of length n with its embedding size being d , we define a key vector K , a query vector Q and a value vector V such that:

$$\begin{cases} Q = XW^Q & \in \mathbb{R}^{n \times d_Q} \\ K = XW^K & \in \mathbb{R}^{n \times d_K} \\ V = XW^V & \in \mathbb{R}^{n \times d_V} \end{cases}$$

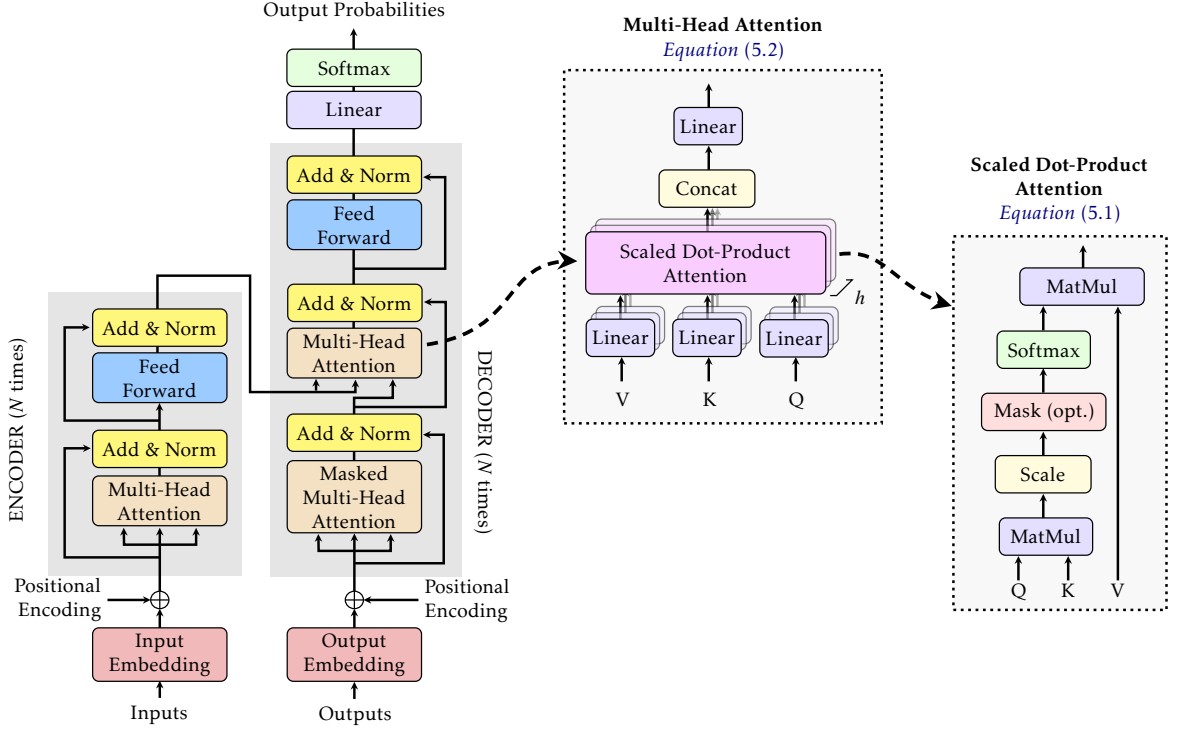


Figure 5.3: Transformer architecture, following an encoder-decoder architecture, with a focus on multi-head attention and its self-attention mechanism. The model is composed of a stack of encoders implementing self-attention which is linked through cross-attention to a stack of decoders implementing masked self-attention. Figures adapted from (Vaswani et al., 2017).

where $W^Q \in \mathbb{R}^{d \times d_Q}$, $W^K \in \mathbb{R}^{d \times d_K}$, and $W^V \in \mathbb{R}^{d \times d_V}$ are weight matrices that can be trained. In practice, the dimensions are often chosen so that they are all equal: $d = d_Q = d_K = d_V$. The output of a self-attention head $\text{Attention}(Q, K, V) \in \mathbb{R}^{n \times d}$ is then defined as:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V \quad (5.1)$$

The query, key, and value can be understood more intuitively as follows:

- The *query* Q_i token asks “how relevant is every other word to me?” and “searches” for other tokens that might be important.
- The *key* K_i token “answers” to the *query* through a dot product, indicating whether it is similar (e.g. grammatically, contextually, semantically) to the *query* so that it might be worth having a high attention weight.
- The *value* V_i is the actual information carried by a word. The next hidden state is derived from this *value* weighted according to how relevant each word is based on the interaction between the *query* and *key*.

An improvement is then to concatenate the outputs of multiple attention heads for the **multi-head attention mechanism**. By considering h attention heads in a layer, the multi-head attention is defined as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) W^O \quad (5.2)$$

$$\text{where head}_i = \text{Attention}\left(XW_i^Q, W_i^K, W_i^V\right)$$

where W^O is a learned parameter matrix. This allows heads from a same layer to focus on different aspects of the sequence. For instance, one head can specifically focus on pronouns, verbs, or positional information such as next or previous tokens (Clark et al., 2019). This can be analogous to a practice in the field of image processing, where convolutional neural networks use layers with multiple kernels. Each one can focus on different features of the image, such as edges or colors (Zeiler and Fergus, 2014). As the Transformer usually consists of multiple stacked encoders or decoders, the output from one layer is then passed as input to the next self-attention layer in the subsequent module with additional unaltered information from the previous layer incorporated through residual connections (Figure 5.3).

Since Transformers do not rely on recurrent mechanism, the order of the tokens within the sequence is incidentally not explicitly represented. To address this, a **positional encoding** is added to the input embeddings to provide information about each token's position. The original approach relies on sinusoidal functions of different frequencies (Vaswani et al., 2017). This method has since been refined through various techniques (Irani and Metsis, 2025), notably through relative positional encodings (Shaw et al., 2018; Huang et al., 2020), which represent the relative distance between tokens instead of their absolute positions.

Transformers offer two main improvements to RNNs. The processing of sequences is sped up, as the entire sequence is passed through the model once and processed in parallel. Moreover, it provides a solution to the problem of vanishing or exploding gradients that occurs with basic RNNs and the issue of hard training with LSTMs. Whereby during back-propagation through time, such recurrent models often struggle in capturing long-term dependencies between words (Noh, 2021). This phenomenon is also true for music generation (Herremans et al., 2017). Transformers have been applied to symbolic music representations, but also in a variety of other domains, such as computer vision (Dosovitskiy et al., 2020), audio (Dong et al., 2018), reinforcement learning (Li et al., 2023a) as well as multimodal applications such as speech-to-text (Latif et al., 2023) or text-to-image (Chang et al., 2023).

Their use has been greatly facilitated with the development of libraries, such as AllenNLP (Gardner et al., 2018), FairSeq (Ott et al., 2019) or more predominantly, HuggingFace (Wolf et al., 2020). This last library offers model architectures, pre-trained models, tokenizers, and various utilities to simplify the development and deployment of NLP applications. As a result, numerous MIR studies have started utilizing HuggingFace by leveraging its tools and resources for musical tasks. These include implementations of subword tokenizers (Section 4.1.2) such as BPE (Sennrich

et al., 2016) or Unigram (Kudo, 2018) used by Kumar and Sarmiento (2023) and model implementations such as BERT (Devlin et al., 2019) for MidiBERT (Chou et al., 2024) or GPT-2 (Radford et al., 2019) used in MMM (Ens and Pasquier, 2020).

Attention-based models for symbolic music processing – In this section, we propose an overview of attention-based models applied to symbolic music data seen through three technical prisms. A first way of characterizing these models is based on their **training paradigm**, namely end-to-end training on specific tasks, or pre-training and fine-tuning (Section 5.3.1). In a musical sense, pre-training assumes a hypothesis of a general understanding of music. Beyond the training process, we describe various architectures that have been implemented (Section 5.3.2). The **model architecture**, based on Transformer encoders, decoders, or combining different types of data, influences how music is processed. Finally, we present the enhancements of the Transformers’ **internal mechanisms**, originally designed for text but customized to specifically process symbolic music data (Section 5.3.3). In this thesis, we further explore these mechanisms through the lens of model explainability (Chapter 7).

Naturally, these three characteristics of models are not mutually independent and all models can be described following each three points. This overview aims first at describing models under different technical prisms which reflect complementary aspects of their design and functionality, and secondly at highlighting how these perspectives contribute to a deeper understanding of the ways attention-based models are adapted to symbolic music processing.

We provide extensive summaries of attention-based models for symbolic MIR are presented in Table B.2 and Table B.3.

5.3.1 Training paradigms

Models can first be categorized by their training paradigm (Figure 5.4). On the one hand, **end-to-end models** are models trained directly for their specific task. On the other hand, **pre-trained models** involve a pre-training step on a generic task followed by a fine-tuning step on one or multiple tasks. This approach is at the heart of Large Language Models (LLMs) in NLP. From an intuitive perspective, textual pre-trained models aim first at modeling or *understanding* language globally, by learning language grammar, structure but is not specialized for a particular task. Fine-tuning then serves to *specialize* this pre-trained model for a particular downstream task. Similarly, MIR studies that rely on pre-trained models also assume that these models first capture global musical structures, before being fine-tuned for specific musical tasks.

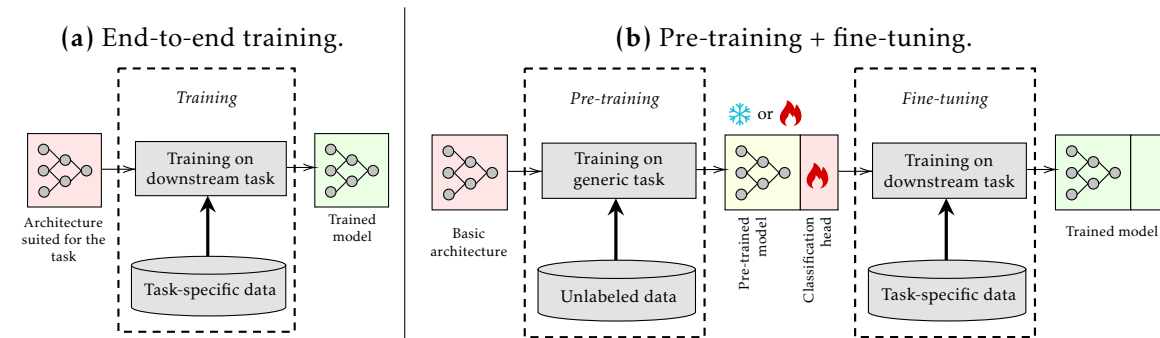


Figure 5.4: Training paradigms: *end-to-end* training involves a model trained specifically for a task, and *pre-training + fine-tuning* involves a pre-trained model trained on a generic task. ❄ indicates that the model parameters are frozen, and 🔥 indicates that they are trainable.

5.3.1.1 End-to-end models

End-to-end models are directly trained for a specific task. In this approach, the model is trained from scratch in a training step in which all parameters are learned jointly from input to output. Beyond its technical simplicity as it requires only one training phase, this paradigm offers a few advantages, notably producing models that are often well-adapted to a specific task and type of data.

In the same way as recurrent networks (Section 5.2.2), self-attention layers can be part or larger models, such as Transformer-based GANs (Goodfellow et al., 2014). Such models are used for generative tasks such as free generation (Muhammed et al., 2021), or emotion-driven generation (Neves et al., 2022). Other models include Transformer-based VAEs (Kingma and Welling, 2013). Multiple tasks are performed using this paradigm, such as priming-conditioned generation (Jiang et al., 2020b), chord-conditioned generation (Choi et al., 2021), lyrics-conditioned generation (Duan et al., 2023), or artistic-controllable generation (von Rütte et al., 2023).

End-to-end models also include several data-specific models designed to process musical data beyond notes. The Chordinator (Dalmazzo et al., 2024) model handles chord data and is based on a minGPT architecture², without its pre-training process. Various models are trained on guitar tablatures, for tablature generation (Chen et al., 2020), metadata-conditioned generation (Sarmiento et al., 2021), style-driven generation (Sarmiento et al., 2023b), or instrument-conditioned generation for bands (Sarmiento et al., 2023a).

Beyond generative tasks, a few models performing analysis tasks have been developed using this end-to-end training paradigm. They are trained on labeled datasets, such as roman numeral-annotated datasets (Chen and Su, 2019, 2021) for functional harmony analysis, or style-annotated datasets (Angioni et al., 2023) for

²<https://github.com/karpathy/minGPT>

style classification.

While the end-to-end training paradigm offers several advantages, it requires a large amount of task-specific data to achieve optimal performance, especially with labeled data which may be too scarce, which can lead to overfitting. Additionally, training a full model from scratch for a single task can be both time-consuming and computationally demanding. Therefore, pre-trained models can offer solutions to overcome part of these challenges.

5.3.1.2 Pre-trained models

In contrast with end-to-end models, *pre-trained* models are usually not task-specific and follow two training phases. The model is first *pre-trained* on a large corpus of data – generally unlabeled – via generic self-supervised tasks (*e.g.* next token prediction or masked language modeling). Once the model is pre-trained, it is *fine-tuned* on a specific downstream task by being trained on a smaller task-specific labeled dataset. This fine-tuning step is also convenient as it requires less data than the pre-training process, and takes less time to train the model instead of multiple trainings from scratch for each existing task. While pre-training was prior to attention-based models, the latest state-of-the-art NLP-derived pre-trained models have switched to Transformer-based architectures both in NLP and MIR. In the field of NLP, such pre-trained models are usually called LLMs, so that the MIR community has also shifted towards using this term (Ma et al., 2024).

State-of-the-art pre-trained language models include **Bidirectional Encoder Representations from Transformers (BERT)** (Devlin et al., 2019). Its architecture is based on a stack of Transformers encoders. BERT is based on a bidirectional training approach and a masked language model: a pre-training task includes masked word prediction by taking into account its left and right context. Multiple variations of BERT applied to symbolic music have been proposed. MuseBERT (Wang and Xia, 2021) develops a specific representation merging musical attributes and relations and processed by the attention mechanism. MusicBERT (Zeng et al., 2021) is a model designed based on RoBERTa (Liu et al., 2019) and improves the pre-training step by implementing a custom bar-level masking strategy instead of the original token masking. The model is evaluated on melody completion, accompaniment suggestion, genre and style classification. A pre-trained MusicBERT is then served in RNBert (Sailor, 2024), which fine-tunes it to perform Roman Numeral Analysis (RNA). A model combining this MusicBERT model with a Music Transformer has been evaluated on several downstream tasks, resulting in better performances than a MusicBERT only (Fu et al., 2023). Instrument-specific BERTs have been implemented such as SoloGPBERT (Sarmiento et al., 2023b) for guitar tablatures, MRBERT (Li and Sung, 2023b) for lead sheets or MidiBERT-Piano (Chou et al., 2024) for piano. This model is then extended beyond piano music and improved with musically meaningful pre-training tasks (Shen et al., 2023). BART (Lewis et al., 2019) is also a

model pre-trained via token masking and is used by PianoBART (Liang et al., 2024), implementing multiple-level token masking and resulting in better performance than other BERT models.

Generative Pre-trained Transformer (GPT) (Radford et al., 2018) is, instead, pre-trained through an auto-regressive task and is often used for tasks involving text or music generation as further explained in Section 5.3.2. In NLP, multiple improvements of GPT have been developed such as GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020) and GPT-4 (Bubeck et al., 2023). For symbolic music, Musenet (Payne, 2019), MMM (Ens and Pasquier, 2020) and MIDI-GPT (Pasquier et al., 2025) are based on GPT-2 and are trained for conditioned generation. Another approach has been implemented for drum music generation (Zhang and Callison-Burch, 2023): music is represented as textual data which and a pre-trained textual GPT-3 is fine-tuned on this textual representation of music.

Finally, beyond GPT and BERT, which are commonly considered as the foundation of pre-trained models in NLP, models that integrate pre-trained components have been developed for symbolic music purposes. LakhNES (Donahue et al., 2019) and DBTMPE (Qiu et al., 2021) avoid the lack of data for their respective downstream tasks by being pre-trained on larger corpora and then fine-tuned for chiptune music generation or genre classification.

5.3.2 Model architecture

Attention-based models can also be categorized by their architecture. In NLP, the first Transformer model for translation (Vaswani et al., 2017) was based on an *encoder-decoder* architecture (Figure 5.3). Afterwards, several NLP models based on either *encoders* (Devlin et al., 2019), *decoders* (Radford et al., 2018), or with modified mechanisms have been proposed. These two modules typically serve distinct functions: an encoder is designed to understand the full input while a decoder aims to generate from an input. MIR studies have leveraged these existing models to adapt them for symbolic music data. Additionally, unlike NLP models that usually handle text for both input and output, MIR experiments have been conducted with *multimodal* models capable of processing different types of data, in particular for tasks like text-to-symbolic music. These multimodal models have found application in domains such as audio processing with MusicLM (Agostinelli et al., 2023) or non-music fields such as image processing with Dall-E (Ramesh et al., 2021).

5.3.2.1 Encoder only

Transformer **encoders** (Figure 5.3, left part) are based on a self-attention mechanism, allowing the learning of knowledge on the *complete* sequence. Bidirectional models,

which are based on this encoder-only architecture, have led to symbolic music adaptations of BERT such as MuseBERT (Wang and Xia, 2021), MusicBERT (Zeng et al., 2021), MidiBERT-Piano (Chou et al., 2024), MRBERT (Li and Sung, 2023b), and SoloGPBERT (Sarmiento et al., 2023b). Going further, Han et al. (2023) analyze the inner embeddings from BERT when trained on symbolic music and highlight the role of specific layers on the model performance. BERT is also used as an architecture without its pre-training process by MTBert (Zhao et al., 2023b) aiming at analyzing the sections of a fugue form.

Beyond BERT, mainly characterized by its pre-training process, Transformer encoders have also been experimented with as a component of global encoder-decoder architecture, in which the encoder keeps a defined role, as detailed below. Such Transformer encoders are also widely used as the discriminator module in GAN-based models (Zhang, 2020; Muhamed et al., 2021), initially developed for generation purposes. They are usually implemented followed by an encoder-decoder or decoder-only as the GAN generator.

5.3.2.2 Decoder only

In contrast with Transformer encoders, Transformer **decoders** (Figure 5.3, right part) implement a *masked* self-attention mechanism. Such models only have knowledge of past tokens so that they are usually implemented for auto-regressive generative tasks. In practice, a mask matrix is added to the dot product between Q and K , where the upper diagonal (*i.e.* attention from past to future tokens) is set to $-\infty$. This causes the corresponding attention weights to become zero after applying the $\text{Softmax}(\cdot)$ normalization.

The first Music Transformer (Huang et al., 2019) is based on a decoder-only model for priming and harmonization tasks, and is then reused by Sulun et al. (2022) for emotion-conditioned generation. Generation is tackled by the MultiTrack Music Transformer (Dong et al., 2023) for instrument-conditioned generation then improved for genre control (Xu et al., 2023), the Choir Transformer (Zhou et al., 2023) for 4-part harmonization, and by Tang et al. (2023) for expressive performance reconstruction. Compose & Embellish (Wu and Yang, 2023a) is a framework based on two encoder-only models, in which the first one aims at generating a lead sheet, followed by the second model that generates the accompaniment.

Decoder-only models are typically trained through a pre-training and fine-tuning process (Section 5.3.1.2), in particular with **GPT-based models**, such as Musenet (Payne, 2019), MMM (Ens and Pasquier, 2020), or MIDI-GPT (Pasquier et al., 2025). By comparing multiple decoder-only architectures, such pre-trained decoder-only models appear to perform better in piano generation (Ferreira et al., 2023).

Several models combine Transformer decoders with lighter recurrent models.

Q&A (Zhao et al., 2023c) combines GRU-based PianoTree-VAEs with a Transformer decoder for arrangement generation. In the same way, Choi et al. (2021) use a bi-LSTM model as a chord encoder, followed by Transformer decoders as pitch and rhythm generators. This architecture is also implemented in the Bar Transformer model (Qin et al., 2022) for long-term structure generation, where the LSTM captures note-level dependencies and Transformer decoders capture bar-level relations.

The fixed length of the accepted context is a limiting issue with Transformers. This is notably crucial because musical sequences are generally much longer than text sequences. The **Linear Transformer** (Katharopoulos et al., 2020) improves the attention mechanism with a linear complexity. The Compound Word Transformer (Hsiao et al., 2021) takes advantage of this computational optimization, coupled with its shorter sequence representation, for piano music generation. SymphonyNet (Liu et al., 2022) is also based on this model to address the even longer length of orchestral pieces, necessitating this lightweight attention mechanism to effectively process such data. Another improvement of Transformers is **Transformer-XL** (Dai et al., 2019), also based on auto-regressive generation, which is able to take into account a much longer context than Transformers. Instead of using fixed-length inputs as in standard Transformers, Transformer-XL extends the context by reusing hidden states from previous segments, allowing the model to attend to a longer memory. Therefore, such models have been used in several generation studies involving multi-track music (Lee et al., 2022), piano music (Huang and Yang, 2020; Muhamed et al., 2021; Wu and Yang, 2023b), lead sheets (Wu and Yang, 2020; Li et al., 2023c), or guitar tablatures (Chen et al., 2020; Sarmiento et al., 2021, 2023a,b). Chang et al. (2021) performs a task of music infilling by implementing an improved Transformer-XL, **XLNet** (Yang et al., 2019b), a Transformer-based model that can attend to past and future in the same way as BERT, while maintaining an autoregressive predicting order.

5.3.2.3 Encoder-decoder

Finally, following the architecture of the vanilla Transformer, multiple models for symbolic MIR implement an **encoder-decoder** architecture (Figure 5.3). This architecture typically allows for the encoder and the decoder to perform different roles in the downstream task.

Functional harmony analysis has been tackled by the Harmony Transformer (Chen and Su, 2019, 2021). The model implements this architecture, where the encoder first perform a chord segmentation task followed by the decoder which infers the chord symbol.

For generative purposes, such architectures are typically used with an encoder which analyzes musical constraints and a decoder that generates musical content. Makris et al. (2021) implements similar architectures, with an encoder analyzing chord valence that conditions an auto-regressive decoder for a generation task. In

the Theme Transformer model (Shih et al., 2023), the encoder analyzes the recurrent theme, from which the decoder generates music depending on the conditions regarding the theme position within the generated content. MusIAC (Guo et al., 2022) is a framework based on an encoder-decoder architecture, in which an encoder is pre-trained as a masked language model, linked with a decoder which performs an infilling task.

Encoder-decoder using pre-trained models as part of their architecture have been developed. Multi-MMLG (Zhao et al., 2023a) is developed for a melody extraction task. It implements an XLNet (Yang et al., 2019b) aiming at classifying notes as main melody or accompaniment, followed by a modified MuseBERT model that extracts secondary melodies. T5 (Raffel et al., 2020) is an encoder-decoder model developed in NLP to handle text-to-text tasks. The model has been adapted for music by MelodyT5 (Wu et al., 2024) for melody-related tasks or Composer’s Assistant (Malandro, 2024) for an infilling task, both using textual representations of music to leverage the text-to-text characteristics of the backbone model. In NLP, encoder-decoder models are often implemented for translation purposes (Vaswani et al., 2017). Gover and Zewi (2022) implement BART (Lewis et al., 2019), an encoder-decoder architecture with learned positional embeddings, for a task analogous to language translation in the realm of music: music arrangement.

For multi-track music, encoder-decoder architectures are typically suited by dedicating multiple decoders or encoders to each specific track. In the context of a multi-track accompaniment generation task, AccoMontage-band (Zhao et al., 2024b) is built on such an architecture consisting of an encoder followed by multiple decoders. BandControlNet (Luo et al., 2024) implements a multiple-decoder architecture called “cross-track Transformer” and improves fidelity-related metrics in a controllable generation task. Finally, this encoder-decoder architecture is largely used in auto-encoder architectures. The Transformer VAE (Jiang et al., 2020b) implements a sampling step from a latent space, from which keys and values are derived for the cross-attention mechanism. MuseMorphose (Wu and Yang, 2023b) and FIGARO (von Rütte et al., 2023) are models based on VAEs, developed for controllable symbolic music generation, which use their latent space representations as constraints.

In Chapter 8, we further build upon this encoder-decoder VAE architecture, which benefits from analysis and generative properties, by proposing a model trained to perform a symbolic music re-orchestration task.

5.3.2.4 Multimodal models

A variety of MIR systems have been developed to integrate other types of data such as text or video, in combination with symbolic music, often referred to as **multimodal models**.

Text-to-image systems (*i.e.* generating an image from a textual description) have

been gaining in popularity these last few years. This has then naturally resulted in text-to-music systems in both audio (Agostinelli et al., 2023) and symbolic music. In symbolic MIR, studies have explored models linking text and music, including a task of lyric-to-melody with TeleMelody (Ju et al., 2022) processing musical high-level features or operating at the syllable level (Duan et al., 2023). MuseCoco (Lu et al., 2023) addresses the text-to-MIDI task by dividing it into two stages: a text-to-attribute step using a BERT model, and an attribute-to-MIDI step treated as a conditional generation task. However, its reliance on a fixed set of predefined attributes limits its flexibility. To overcome this, Text2midi (Bhandari et al., 2025) proposes a similar architecture but replace the attribute step with a frozen FLAN-T5 model that processes the text caption and passes its hidden states to a Transformer decoder trained to generate music conditioned on these states.

However, most text-to-symbolic-music tasks currently process ABC notation, as this encoding is already in a textual format (Wu and Sun, 2023). ChatMusician (Yuan et al., 2024) is based on Llama-2 (Touvron et al., 2023) and is framed as a music chatbot which can write ABC notation music and chat with a user about music theory knowledge. GPT-4 is able to perform such a text-to-ABC task, among multiple other tasks (Bubeck et al., 2023) but struggles at modeling musical concepts such as harmony. To overcome this issue, this task is split into multiple musically-meaningful subtasks in ComposerX (Deng et al., 2024a) which uses GPT-4 for melody generation, harmonization and instrument selection. Finally, beyond generative tasks, CLaMP (Wu et al., 2023) integrates two BERT-based models – one for text encoding and the other for music encoding – for a tune query task based on natural language descriptions. Improvements of the CLaMP model (Wu et al., 2025) are used as the reward feedback in a reinforcement learning framework developed by Wang et al. (2025), which generates ABC Notation conditioned on musical period, composer, and instrumentation.

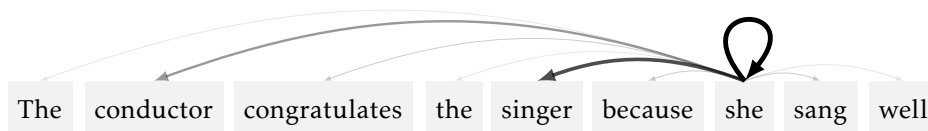
Multiple systems have been experimenting with symbolic music generation for video considering the use of music in videos like soundtracks in movies. Di et al. (2021) relies on a linear Transformer to generate symbolic music for videos that are analyzed in terms of motion speed and saliency conditioning the generated music rhythm. Kang et al. (2023) add a semantic and emotion analysis of the scene, and more specifically generate chords matching these video features.

5.3.3 Adapting attention models inner mechanisms and training paradigm to symbolic music

Extensive studies have been conducted regarding the mechanisms of Transformers applied to text data, including positional encoding and attention mechanisms. When applied to symbolic music, these mechanisms may be improved to be tailored for this different context.

Similarly to text, multiple MIR studies have also developed *positional encodings* and customized them for the specificities of music. With the Music Transformer model (Huang et al., 2019), a **relative positional self-attention** mechanism is developed for music generation. This improves the relative positional encodings from (Shaw et al., 2018), which consider the tokens’ relative positions in the attention mechanism rather than their absolute positions. The proposed enhancement effectively reduces the time complexity, enabling the processing of much longer sequences that are characteristic of symbolic music data. Similarly, the **stochastic positional encoding** (Liutkus et al., 2021) aims to be compatible with linear complexity attention, improving the consistency of generated chunks of music on long sequences. The specificities of multi-track music inspired the SymphonyNet model to develop a **3-D positional embedding** (Liu et al., 2022) in order to make the model semi-permutation invariant. Tokens are embedded over three axes – note, measure and track – in which the track order is completely permutation invariant (*i.e.* the order of tracks is inconsequential), but the temporal position of the notes or the measures must remain time-dependent. Similarly, musically meaningful positional encodings have been developed based on notes attributes and relations (Wang and Xia, 2021), bars (Chang et al., 2021), musical themes (Shih et al., 2023), structure and musical time (Payne, 2019), or instruments (Zhao et al., 2023c, 2024b).

(a) Text self-attention visualization. Arrows represent self-attention weights between each token (here, a full word) and the token <she>.



(b) Musical self-attention visualization through a piano roll representation. The tokenization in use is MIDI-like. Reproduced from (Huang et al., 2018).

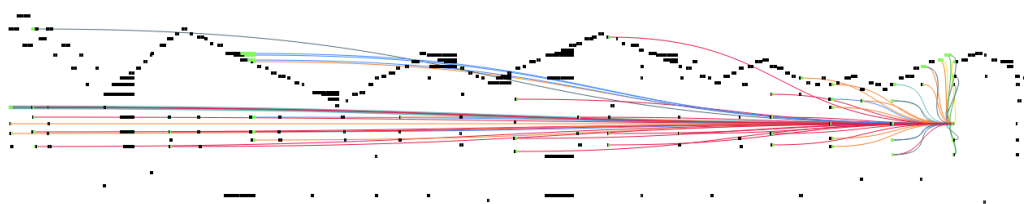


Figure 5.5: Text and musical self-attention visualization.

Transformers implement a *self-attention* mechanism. Beyond further adapting it, this mechanism can be easily *visualized* both in text and music (Figure 5.5). However, while attention in language models can often be interpreted through grammatical or semantic roles (Clark et al., 2019), attention links between musical tokens are instead guided by distinct musical aspects. Such visualization can show differences between attention heads being more or less specialized in chords or melody (Huang et al., 2018). Self-attention has also been studied as a source of high-level interpretations, such as music theory insights, in terms of motifs, harmony, or temporal dependen-

cies. Such musical objects captured by attention are numerous, including cadential passages (Loiseau et al., 2021) or musical phrases or modulating sequences (Jiang et al., 2020a). Dong et al. (2023) notably introduce a metric, the **mean relative attention**, designed to measure how much attention the model gives to elements in the input that are a certain distance away (in terms of pitch, beat, etc.) from the current element. For instance, their model appears to highlight consonant musical intervals such as octaves, fifths or fourths.

Beyond visualization, this attention mechanism itself has also been adapted to symbolic music. The Museformer model (Yu et al., 2022) is based on a **fine-grained and coarse-grained attention** aiming at reducing the complexity of the mechanism, leveraging the expected repetitive aspect of music. This mechanism is based on an attention divided into a fully developed attention mechanism on more relevant bars (*i.e.* the repeated ones) and a weaker attention on less relevant ones. The **RIPO (Relative Index, Pitch and Onset) attention** (Guo et al., 2023) is proposed with the **fundamental music embedding**, relying on the structure of symbolic music built on relative onsets and pitches. In a context of controllable style transfer, the MuseMorphose model (Wu and Yang, 2023b) includes an **in-attention conditioning** that takes into account constraints in the self-attention computation. The **structure-enhanced self-attention** from BandControlNet (Luo et al., 2024) incorporates a similarity score between bars in the attention computation in order to enhance the structure consistency of tracks. For lead sheet data, a melody/rhythm cross attention is implemented in MRBERT (Li and Sung, 2023b), in which these two features are merged and simultaneously processed through attention.

Finally, *training strategies* with musical specificities have also been developed. Based on a GAN architecture (Goodfellow et al., 2014), a **local prediction map** (Neves et al., 2022) is proposed so that the discriminator also specifies which parts of the generated sequence is real or generated, instead of the whole sequence. The Nested Music Transformer (Ryu et al., 2024) improves the training of generative models based on Compound Words (Hsiao et al., 2021) by generating the inner components of each word auto-regressively, rather than producing full tokens at once. This approach enhances both memory efficiency and generation quality. Pre-trained models, in particular masked language models such as BERT, are usually pre-trained on a token prediction task from a masked sequence and a next sentence prediction task (Devlin et al., 2019). For symbolic music, MusicBERT (Zeng et al., 2021) is pre-trained with a **bar-level masking**: instead of masking a single token and leveraging its Octuple representation (Figure 4.7a), the pre-training process masks a type of feature for all the tokens within a bar. This masking is improved with **quad-attribute masking** (Shen et al., 2023). Going further, PianoBART (Liang et al., 2024), which also uses an Octuple representation, implements a multi-level object masking strategy, where the masked token can be at the level of an Octuple-element, the whole Octuple, or ranging over multiple bars. These strategies avoid information leakage between tokens, as some musical features can be easily inferred from adjacent tokens. Taking inspiration from the multi-task pre-training approach of the original

BERT model, [Shen et al. \(2023\)](#) also propose an analogous pre-training task with next sentence prediction with **key prediction**. Finally, MuseBarControl ([Shu et al., 2024](#)), a Linear Transformer for controllable generation, implements a pre-training task aiming at directly incorporating control signals during the pre-training step to improve the resulting bar-level controllability.

This chapter has reviewed *models* adapted from NLP for symbolic music processing. In particular, we focus on Transformer-based models, on which most of state-of-the-art models for music analysis and generation are based on. These models have demonstrated versatility across various tasks and input representations, offering multiple architectural and training paradigm alternatives that can be well-adapted for symbolic music processing. Though, because of their complexity and their size, they remain black boxes. Yet, understanding the musical knowledge they capture remains valuable and is the focus of a technical contribution presented in [Chapter 7](#).

Part II

Technical Contributions

The structured overview of NLP methods for symbolic music processing presented in the first part of this thesis has led to the identification of multiple possible contributions in the three axis, *representations*, *models* and *tasks*. In this part, we present technical contributions developed by taking inspiration from existing NLP methods to adapt them for symbolic music processing.

Therefore, this part is organized following the organization of NLP methods for MIR developed in the previous part:

- We examine further **sequential representations** of symbolic music in the context of event-based tokenizations ([Chapter 6](#)). In particular, we explore how the expressiveness of such tokenizations can be improved. This tokenization expressiveness can be related to the choice of the initial *alphabet*, for which we propose a tokenization strategy based on intervals instead of absolute pitches. This expressiveness is then studied through the analysis of *grouping* strategies, for which we specifically focus on Byte-Pair Encoding (BPE) which aims at building more informative tokens.
- We then analyze **models** through the lens of model explainability, focusing on the attention mechanism ([Chapter 7](#)). We focus on a task functional harmony analysis performed by an encoder-only model and we propose a framework to analyze the inner attention mechanisms of such model. In particular, we focus on two behaviors derived from the attention mechanism, with a first focus on attention spans. We then evaluate the role of individual attention heads in the sub-task of local key detection that we quantify using Layer-wise Relevance Propagation (LRP).
- Finally, as an application of NLP methods implemented for a MIR **task**, we present METEOR, a Transformer-based model for multi-track music generation ([Chapter 8](#)). More precisely, the model is trained for a task of automatic re-orchestration by adapting an existing model trained for piano style transfer which implements a custom attention mechanism with token constraints.

Chapter 6

Improving music sequential representations' expressiveness

6.1	Interval-based tokenization for pitch representation	80
6.1.1	Intervalization	81
6.1.2	Evaluating intervalization on downstream tasks	83
6.1.2.1	Downstream tasks	84
6.1.2.2	Model & pre-training	85
6.1.2.3	Results	86
6.1.3	Towards improving token alphabet expressiveness	90
6.2	Analyzing byte-pair encoding for monophonic and polyphonic music	92
6.2.1	Analyzing music byte-pair encoding	92
6.2.1.1	Comparing text and music BPEs	93
6.2.1.2	Musical content carried by supertokens	95
6.2.2	Evaluating BPE on musical phrase segmentation	97
6.2.2.1	Task & data	97
6.2.2.2	Results	98
6.2.3	Towards improving token grouping expressiveness	100
6.3	Tokenization expressiveness or model power?	101

In [Chapter 4](#), we proposed a formalization of event-based tokenizations seen as a chosen *alphabet* with possible *grouping* strategies. The alphabet describes which musical features are encoded, and the grouping strategy aims at shortening sequences by creating longer tokens. However, MIR studies tend to overcome the choice of the tokenization strategy by relying on existing tokenization that often encode local characteristics (pitch, velocity, . . .). Instead, these two aspects may be further explored to build more musically *expressive* tokens.

The current chapter studies two attempts towards improve music sequential

representations' expressiveness with regards of the *alphabet* and the *grouping* strategy.

In a first approach, we explore how the choice of the initial *alphabet* can be more expressive, so that the atomic elements directly reflect musical attributes. In this way, we explore how choosing interval-based tokens instead of absolute pitch tokens in the initial alphabet can later impact a model trained on MIR analysis tasks (Section 6.1).

A second approach is to improve the expressiveness of a tokenization strategy through a *grouping* process, which builds more musically informed tokens. In particular, we study Byte-Pair Encoding (BPE), an algorithm typically used to group tokens to build words or subwords and its impact on analysis tasks, depending on the polyphony level of the music (Section 6.2).

6.1 Interval-based tokenization for pitch representation

This section is based on a work published at the workshop *AI for Music* co-located with AAAI 2025 (Association for the Advancement of Artificial Intelligence) (Le et al., 2025b).

A key distinction between text and music lies in the presence of *pitch* – a fundamental dimension of music that has no direct counterpart in written natural language. In event-based symbolic music tokenization alphabets (Chapter 4), the information of pitch is typically represented through dedicated tokens. Traditional strategies mostly employ MIDI numbers to encode pitch information: in other words, they most often rely on *absolute* pitch encoding. While effective this approach may overlook relational or contextual aspects between notes.

Instead, one may often memorize music by its melodic contour, considered as a sequence of intervals unchanged by transposition to different keys, rather than by their absolute pitches (Dowling and Fujitani, 1971). Similarly, in the context of tonal music, harmony is based on the relation between the notes constituting a chord and a tonal center more than their absolute pitches. Musical *intervals* capture the relative distances between pitches, emphasizing relationships over fixed pitches, which aligns more closely with human musical perception. Applied to computational representations of music, interval-based tokenization may provide an alternative and more expressive approach to pitch encoding.

In this work, we present tokenization strategies based on intervals that can be used jointly with absolute pitch encodings. Such tokenizations are shown to improve model performances in analysis tasks. This work's contribution is two-fold:

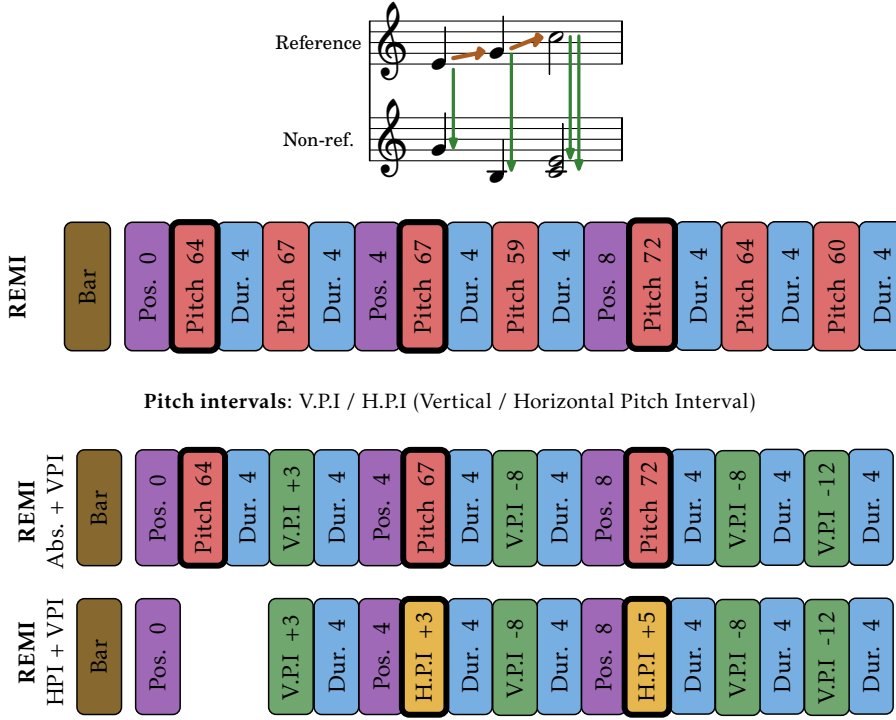


Figure 6.1: Representations of the sheet music based on absolute and different variants of intervalization of the REMI tokenization. (Abs.: *Absolute pitch encoding*)

- We first introduce a general framework to build interval-based tokenization strategies. This framework is built around representing a chosen reference (*e.g.* the melody) with using either absolute or interval-based encoding, while encoding the remaining content in relation to this reference. We formalize this approach to ensure it remains flexible and extensible for future potential adaptations.
- We then show that interval-based tokenization can improve model performances on three downstream tasks. Moreover, we show that among the studied interval-based encoding strategies, optimal tokenization settings depend on the downstream task and can result in musically meaningful interpretations.

6.1.1 Intervalization

In this section, we propose a formal description of the *intervalization* process which aims at transforming a tokenization strategy originally based on absolute tokens only into a interval-based variant tokenization strategy. Let \mathbf{x} be a sequence of note events:

$$\mathbf{x} = \{e_1, \dots, e_T\}$$

Each note event can be written as $e_k = (p_k, t_k)$ where $p_k \in \{1, \dots, 128\}$ denotes an absolute pitch element and t_k the onset time element associated to the note.

Let $\mathbf{x}_{\text{ref}} \subset \mathbf{x}$ be a sub-sequence of notes, chosen as *reference*. \mathbf{x}_{ref} can correspond for instance to the *bottom-line* of the musical content, the *skyline*, or the melody if available in the data:

$$\begin{aligned}\mathbf{x}_{\text{ref}} &= \{e_{\text{ref}_1}, \dots, e_{\text{ref}_\tau}\} \\ &= \{(p_{\text{ref}_1}, t_{\text{ref}_1}), \dots, (p_{\text{ref}_\tau}, t_{\text{ref}_\tau})\}\end{aligned}$$

\mathbf{x}_{ref} is chosen to be a monophonic sequence (*i.e.* without simultaneous events): $t_{\text{ref}_j} \neq t_{\text{ref}_{j'}}$, for $j, j' \in \{1, \dots, \tau\}$.

The choice of \mathbf{x}_{ref} induces a partition of \mathbf{x} into τ subsets of events:

$$\mathbf{x} = S_1, \dots, S_\tau$$

where S_j is defined as the set of notes occurring between e_{ref_j} and $e_{\text{ref}_{j+1}}$:

$$S_j = \{e_{\text{ref}_j}\} \cup \left\{ (p, t) \in \mathbf{x} \left| \begin{array}{l} t_{\text{ref}_j} \leq t < t_{\text{ref}_{j+1}} \\ (p, t) \neq e_{\text{ref}_j} \end{array} \right. \right\}$$

We call *intervalization* I the process of converting an absolute pitch element into a pitch interval element. The sequence \mathbf{x} is thus transformed into the sequence $\mathbf{x}_{\text{relative}}$:

$$\mathbf{x}_{\text{relative}} = \{I(e_1, \mathbf{x}_{\text{ref}}), \dots, I(e_T, \mathbf{x}_{\text{ref}})\}$$

$I(e, \mathbf{x}_{\text{ref}})$ is defined for $e = (p, t) \in S_j$ as:

$$I(e, \mathbf{x}_{\text{ref}}) = \begin{cases} (I_{\text{ref}}(p_{\text{ref}_j}, p_{\text{ref}_{j-1}}), t) & \text{if } e = e_{\text{ref}_j} \\ (I_{\text{non-ref}}(p, p_{\text{ref}_j}), t) & \text{otherwise} \end{cases}$$

where I_{ref} represents a method specifying the encoding method for reference pitch tokens, while $I_{\text{non-ref}}$ represents the encoding method for non-reference tokens.

I_{ref} can be chosen as being an encoding using absolute pitches:

$$I_{\text{ref}}(p_{\text{ref}_j}, p_{\text{ref}_{j-1}}) = p_{\text{ref}_j}$$

or horizontal pitch intervals (Kermarec et al., 2022) (*i.e.* where each pitch is encoded as a *horizontal* interval with the previous pitch within the reference sequence):

$$I_{\text{ref}}(p_{\text{ref}_j}, p_{\text{ref}_{j-1}}) = p_{\text{ref}_j} - p_{\text{ref}_{j-1}}$$

In the latter case, the first event is dropped.

Tokenization	\mathbf{x}_{ref}	I_{ref}	$I_{\text{non-ref}}$
REMI-abs	–	Absolute	Absolute
REMI-abs+VPI			
ref-melody	Melody	Absolute	Vertical Pitch Interval
ref-skyline	Skyline	Absolute	Vertical Pitch Interval
ref-bottom-line	Bottom-line	Absolute	Vertical Pitch Interval
REMI-HPI+VPI			
ref-melody	Melody	Horizontal Pitch Interval	Vertical Pitch Interval
ref-skyline	Skyline	Horizontal Pitch Interval	Vertical Pitch Interval
ref-bottom-line	Bottom-line	Horizontal Pitch Interval	Vertical Pitch Interval

Table 6.1: Tokenizations studied in this chapter, including the original REMI tokenization (REMI-absolute), and interval-based tokenizations based on REMI. (Abs: *Absolute pitch encoding* ; VPI: *Vertical Pitch Interval* ; HPI: *Horizontal Pitch Interval*)

Similarly, $I_{\text{non-ref}}$ can be chosen as being an encoding using absolute pitches:

$$I_{\text{non-ref}}(p, p_{\text{ref}_j}) = p$$

or vertical pitch intervals (*i.e.* where each pitch is encoded as a *vertical* interval in relation to the simultaneous pitch of the reference sequence):

$$I_{\text{non-ref}}(p, p_{\text{ref}_j}) = p - p_{\text{ref}_j}$$

We give visual examples of these intervalization strategies derived from the REMI tokenization (Huang and Yang, 2020) in Figure 6.1.

Although the choice of \mathbf{x}_{ref} , I_{ref} and $I_{\text{non-ref}}$ can be much larger as described further, we limit this study to the six intervalization strategies listed in Table 6.1 applied to the REMI tokenization using the MidiTok package (Fradet et al., 2021). Though, we altered the original REMI tokenization by dropping <Velocity> tokens. Since the datasets are not derived from performance data, the velocity values of these tokens are only set arbitrarily. Moreover, as our tasks are not concerned with musical interpretation, we assume that velocity has minimal influence on model performance.

6.1.2 Evaluating intervalization on downstream tasks

In this section, we present an experimental framework that aims to evaluate the impact of intervalization on various MIR tasks. More precisely, we consider the following hypotheses:

- (H1) The choice of an intervalized tokenization has a positive impact over an absolute tokenization on the performance of a model on a task.

(H2) Given a task, not all intervalized tokenization strategies are equivalent. In particular, the choice of one specific \mathbf{x}_{ref} can be more judicious for this task.

Our experiments involve three downstream tasks – era classification, start-of-phrase detection, and chord inversion identification – further described in the next paragraph, on which pre-trained and end-to-end BERT models (Devlin et al., 2019) are evaluated on different interval tokenizations.

6.1.2.1 Downstream tasks

We evaluate the impact of intervalization on three supervised downstream tasks associated with different datasets. We chose to evaluate the tokenization strategies on the two types of tasks presented in Section 3.1: a *sequence classification task* and *token classification tasks* (which we also refer to as sequence tagging tasks).

Because we propose to evaluate the impact of intervalization when the reference is the melody, the datasets are composed of music with a homophonic texture, characterized by a polyphonic music which include a single melody supported by an accompaniment (Benward, 2018). In particular, we chose these datasets so that the melody is played by a single track during each piece. A quantitative description of the datasets in terms of token count and musical pieces is given in Table 6.2 (bottom). We focus on the following three tasks:

- **Start-of-phrase detection.** We first introduce a start-of-phrase detection as a *sequence tagging task*. The model is trained to classify each token of a sequence as being a start-of-phrase or not. We build a synthesized dataset of folk tunes with generated piano accompaniment which includes start-of-phrase annotations. In particular, we consider the ESSEN dataset (Schaffrath, 1995) which includes 7k folk melodies from 47 countries with phrase annotations, which are arranged for solo piano using a lead sheet piano arrangement model (Zhao and Xia, 2021). Each tune was composed of 5.6 phrases on average. The construction of this dataset follows a similar approach to that of the MTC-Piano dataset, which is described in the following Section 6.1.2.2.
- **Chord inversion identification.** Inspired by the task of figured bass identification (Ju et al., 2020), we implement a chord inversion identification task as a *sequence tagging task*. The model is trained to classify each token as being part of a root position, first, second, or third inversion chord¹. We consider the When-in-Rome dataset which includes roman numeral labels (Gotham et al., 2023a) from which only the chord inversion characteristic is extracted. From this dataset, we only kept Bach chorales, from which we assume the melody to be the soprano voice. While the dataset does include much more data than

¹The theory of functional harmony is further described in Section 7.2.1.

chorales, other instrumentations, such as piano solo or orchestral pieces, do not clearly involve a melodic line or a single instrument playing the melody throughout the whole piece, which can be used as a reference in our evaluation.

- **Era classification.** This task is a binary *sequence classification task*. We consider the OpenScore Lieder dataset (Gotham and Jonas, 2022), which includes voice and piano pieces by 107 composers from 1730 to 1949. In particular, we characterize each composer by an average year derived from their birth and death year. We selected the discriminative year, 1865, for this binary classification task based on the distribution of composition year within the dataset so that the dataset is balanced between the two classes.

Dataset	Task	# tokens (# pieces)
POP909	Pre-training	12.1M (2897)
MTC-Piano	Pre-training	12.4M (18.1k)
String quartets	Pre-training	3.2M (121)
<i>Total</i>	<i>Pre-training</i>	<i>27.8M (21.1k)</i>
Lieder	Era classification	2.7M (1356)
ESSEN-Piano	Phrase detection	3.4M (6926)
Bach chorales	Chord inversion identification	204k (371)

Table 6.2: Description of the datasets used for pre-training and downstream tasks, namely era classification, start-of-phrase detection and chord inversion identification. The count of tokens is given in terms of REMI-absolute tokens.

6.1.2.2 Model & pre-training

For performing these tasks, we chose to implement a Transformer encoder-only model on which a classification layer is plugged. We consider a pre-training + fine-tuning strategy, following the model MidiBERT-Piano (Chou et al., 2024). We gather three datasets for pre-training for which quantitative descriptions are given in Table 6.2 (top):

- POP909 (Wang et al., 2020a) includes Chinese pop songs with tracks annotated as “melody”, “lead”, and “piano”. The melody is considered as being the “melody” and “lead” tracks merged together.
- We introduce MTC-Piano, a dataset that compiles piano arrangements of Dutch folk melodies sourced from the Meertens Tune Collections (MTC) (Van Kraenburg et al., 2014). More precisely, the MTC dataset contains monophonic folk tunes with several annotations, including chords. Therefore, by considering each tune as a lead sheet, we generate a piano accompaniment with the

AccoMontage model (Zhao and Xia, 2021). Beyond serving as a large dataset of polyphonic music well suited for pre-training in this current work, this dataset has also been compiled to make use of the musical phrase annotations in the work related to BPE presented later in this chapter (Section 6.2).

- The OpenScore String quartets collection (Gotham et al., 2023b) features string quartets from European composers spanning the classical to late-romantic periods. The melody is approximated as the part played by the first violin.

Using the union of these corpora as a pre-training dataset, we consider a Transformer-based encoder-only architecture on an unsupervised masked language model pre-training task (Devlin et al., 2019). Using this common architecture, we pre-train seven models, one for each tokenization strategy (Table 6.1). For the fine-tuning process, we use the datasets described in Section 6.1.2.1 to train the model on their corresponding downstream task.

The implementation of the model is based on the MidiBERT-Piano model (Chou et al., 2024). However, while the latter consists of 12 layers with 12 heads each, we use a smaller model with 3 layers and 8 heads per layer. The fine-tuned models then adds a classification head on top on the pre-trained model, with its nature depending on the downstream task. This classification is a self-attention layer with a linear layer for the full sequence classification task and a pair of linear layers for the two sequence tagging tasks. In particular, the pre-trained model weights are not frozen during the fine-tuning process. For each downstream task, we evaluate the three intervalization strategies on both an end-to-end model (*i.e.* the full model is initialized randomly and trained from scratch) and a pre-trained + fine-tuned model.

This configuration results in a pre-trained model with 14M parameters, which is eight times lighter than MidiBERT-Piano. Thus, using our training hyperparameters, two models can fit into a single 12GB Tesla P100 GPU. The models are pre-trained until an early stopping on the validation accuracy of 10 epochs, resulting in approximately one week of pre-training for all the seven models on our hardware. The fine-tune process is stopped after an early stopping of 3 epochs.

6.1.2.3 Results

We evaluate models on the above downstream tasks following the two hypotheses (H1) and (H2) presented above. To this end, we consider various settings regarding intervalization strategies presented in Table 6.1, namely, an absolute tokenization and two intervalized tokenization with 3 references each. For each downstream task, we evaluate each tokenization strategy on both an end-to-end and a pre-trained model. Each model is trained and evaluated on three seeds of the dataset splits. Therefore, in total, 42 trainings have been performed on each downstream tasks by

allowing all the possible combinations:

$$\begin{aligned}
 & ((1 \text{ REMI-abs}) + (1 \text{ REMI-abs+VPI} + 1 \text{ REMI-HPI+VPI}) \times (3 \text{ references})) \\
 & \times (1 \text{ pre-trained} + 1 \text{ end-to-end}) \\
 & \times (3 \text{ seeds})
 \end{aligned}$$

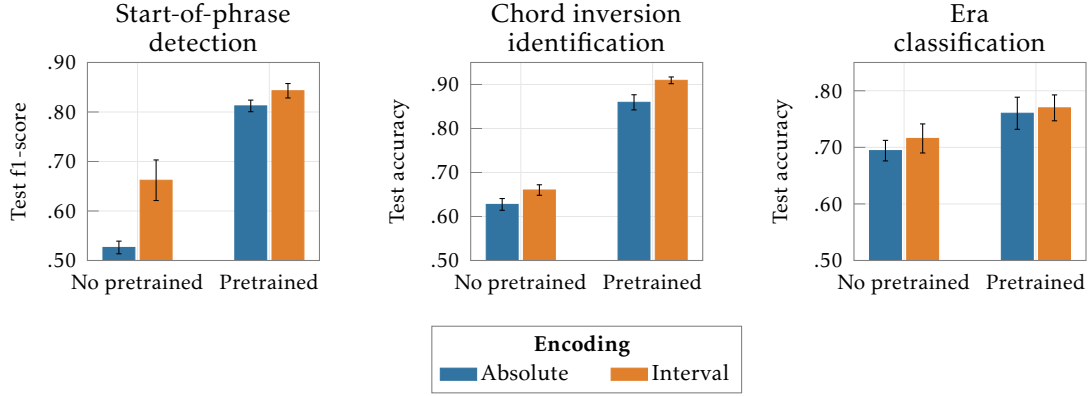


Figure 6.2: Performance comparison between absolute and intervalized tokenization strategies on the three downstream tasks with non pre-trained and pre-trained models. The intervalized model is based on the reference resulting in the best performance.

Impact of intervalization (H1) – First of all, for the hypothesis (H1), we focus on comparing the performance of models using an absolute tokenization in comparison with intervalized ones. In Figure 6.2, we compare the absolute encoding model with three interval-based models using different references, and report the performance of the best-performing intervalized model.

Our results show that, for all tasks, there exists an interval-based model that outperforms its absolute-encoding counterpart, both in pre-trained and end-to-end settings. However, such improvements range from a marginal 1.2% performance increase in the case of a pre-trained model on era classification (*i.e.* the sequence classification task) to a significant 6% increase with an end-to-end model trained on start-of-phrase detection. Moreover, our results show that pre-training models systematically outperform end-to-end models. On the three tasks, pre-trained model performances are on average 1.2 times better than their end-to-end counterparts, with variations depending on the task.

Therefore, for the hypothesis (H1), we can state that there is a positive impact of intervalized tokenization over absolute ones, more particularly for token classification tasks.

Impact of intervalization references (H2) – Beyond the raw contribution of intervalization for the downstream tasks, we then analyze the impact of the chosen reference on performance to evaluate hypothesis (H2). More specifically, we evaluate whether certain references are better suited to specific tasks. To this end, we compare models trained using different references against one another.

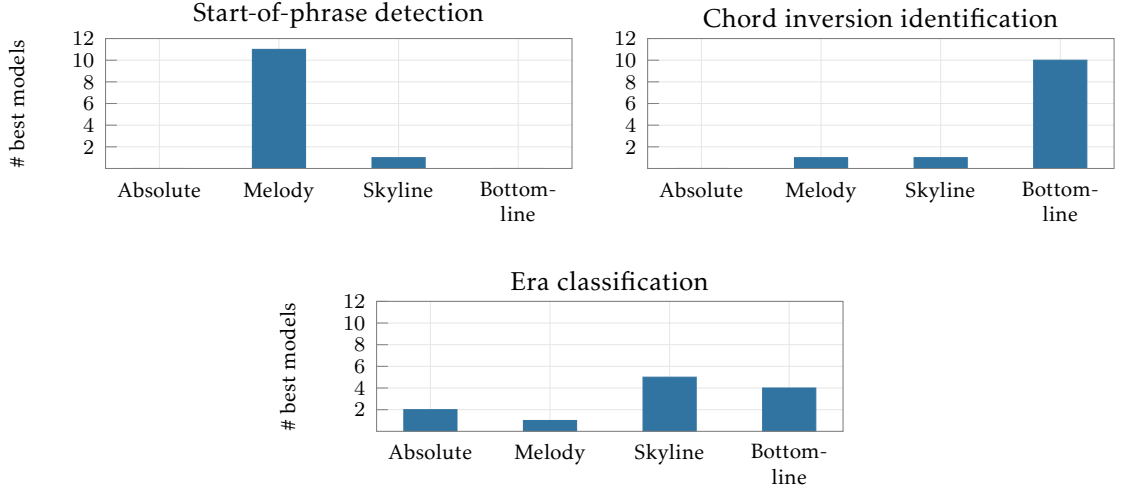


Figure 6.3: Count of best intervalization references when comparing intervalized models with various references and the absolute. For each task, we consider two pre-trained and two end-to-end models trained on the tokenizations shown in Table 6.1. This results in 12 comparisons by task, where each comparison involves three intervalization references tested against an absolute tokenization.

For each task, for each split of the dataset among the three considered seeds, we perform two comparisons – one for the end-to-end setting and one for the fine-tuned setting – between the model trained without intervalization and the three models trained with intervalized tokenizations, each based on one of the three reference types, for a given setting of $I_{\text{non-ref}}$ and I_{ref} . We then count the number of times that the choice of a particular reference leads to the best result among these four models. In total, we therefore proceed to 12 comparisons per task: (3 seeds \times (2 pre-trained models + 2 end-to-end models)). Each comparison involves 4 models (3 intervalized + 1 absolute). The counts of the best models associated with their reference are shown in Figure 6.3.

On the whole, for the hypothesis (H2), we show that there is a preferred reference \mathbf{x}_{ref} which depends on the considered task.

Models trained with a melodic reference achieve the best performance in 11 of 12 comparisons for the *start-of-phrase detection* task in contrast with the skyline or bottom-line references. This confirms intuitions from music theory regarding the role of melody in phrasing highlighted, among others by Schoenberg (Schoenberg et al., 1999, p. 3):

Phrase endings may be marked by a [...] melodic relaxation through a drop in pitch, the use of smaller intervals and fewer notes;

For the *era classification* task, the choice of the intervalization reference does not show a significant impact on the model performance. Unlike the other tasks, which

involve local token tagging, this task focuses on classifying entire sequences into more abstract classes. Such a higher-level task may explain why tokenization plays a less critical role in this context. Further experiments could be considered with more consistent era labelling, for instance considering more discriminative era labels, such as baroque and romantic era pieces. Finally, for the task of *chord inversion identification*, the bottom-line reference leads to the best models in 10 of the 12 comparisons. This result may not be surprising given that chord inversion are most often defined by the bass note constituting the chord. Though, we further investigate potential explanations for the effectiveness of this intervalization reference compared to the others in the next paragraph.

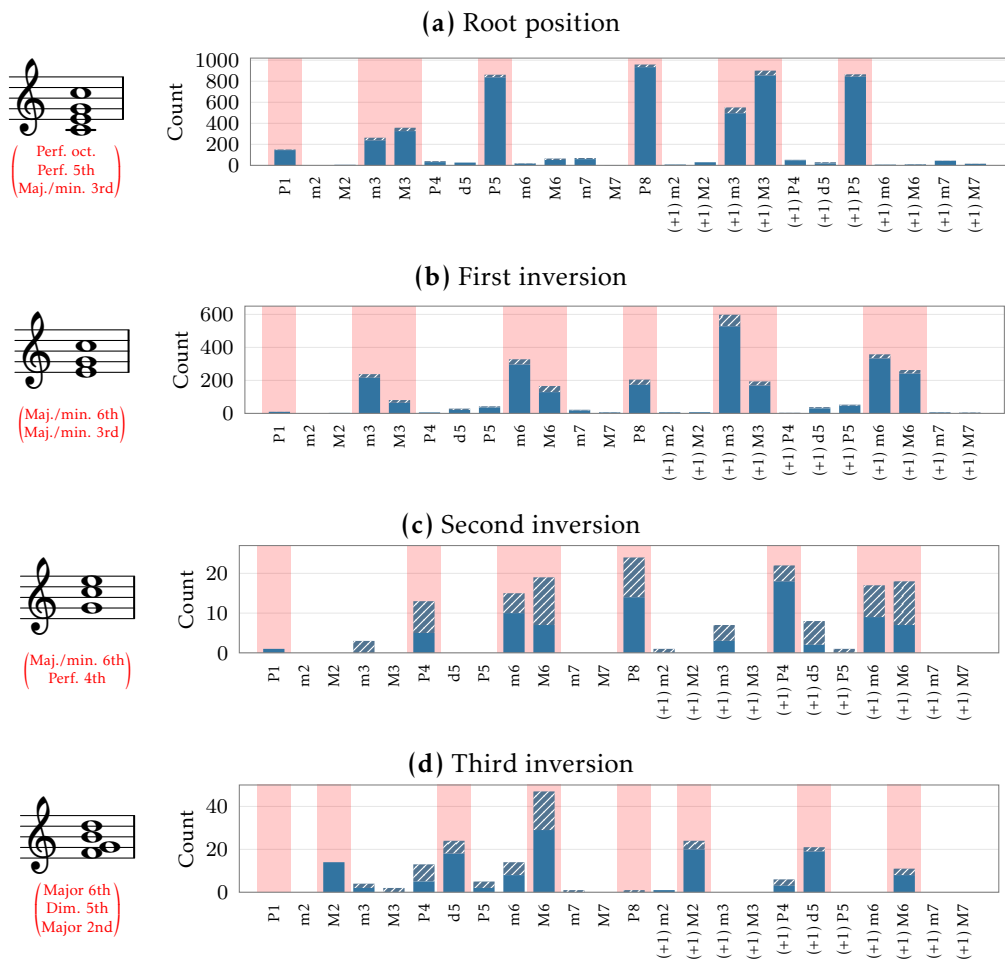


Figure 6.4: Histograms of vertical pitch interval tokens predicted as root position, first, second or third inversion. The hatched part of a bar represents the proportion of false positives. Red highlights indicate the intervals that occur in each chord inversion. The notation (+1) indicates an additional octave.

Musical attributes reflected by the intervalization – We focus on the task of chord inversion identification and we analyze how a tokenizer with a bottom-line reference classifies each token. We study the frequency of vertical pitch tokens classified by

the model as root position, first, second, or third inversion (Figure 6.4).

This shows that particular sets of vertical pitch intervals are more prominent within specific inversions. In particular, these more common interval values match the musical definitions of chord inversions. For example, a major (resp. minor) third in combination with a fifth in relation to the bass note defines a root position major chord (resp. minor chord) (Figure 6.4a). The presence of occurrences outside these musical definitions of major/minor/dominant chords can reflect the presence of chord extensions or note embellishments such as passing notes. Moreover, analyzing the proportions of false positives among the predictions can explain some of the model's errors. For example, for first inversions (Figure 6.4b), the largest number of false positives occur with thirds and sixths, which are intervals that also compose chords in root position and second inversion respectively.

Going further, various four-part writing principles (Peters, 2016; Benward, 2018) can be inferred from these frequency distributions. For example, Figure 6.4a and 6.4b show that third intervals (m3 and M3) occur more often with an additional octave than within the same octave. This aligns with voice spacing rules, which typically recommend that bass and tenor voices are not too close. Similarly, Figure 6.4d shows that the bass is not doubled at the octave (P8) in the case of a third inversion. This is consistent with a four-part harmony rule stating that the seventh of a dominant seventh chord is typically not doubled.

These experiments show that a more *expressive* alphabet based on interval tokens to encode pitch information, can potentially improve the performance of models initially trained with basic absolute pitch tokens. Moreover, this increased expressiveness can also capture established musicological principles. Besides these quantitative results, the primary contribution of this section is a formal description of interval-based tokenizations, that may serve as a foundation for developing further expressive alphabets.

6.1.3 Towards improving token alphabet expressiveness

This section proposes possible future directions to experiment with interval-based tokenizations, based on the formalism presented above.

The previous experiments limited intervalization strategies to three types of \mathbf{x}_{ref} . In particular, we have assumed $\mathbf{x}_{\text{ref}} \subset \mathbf{x}$. By releasing this, \mathbf{x}_{ref} can be chosen as a musically meaningful reference such as a reference sequence composed only of the tonal center of a piece, which may help in tasks such as harmonic analysis.

In addition, we have implemented only two types of interval encodings, namely horizontal and vertical pitch intervals. However, similarly to pitches which can be encoded as `<pitch-class>` and `<octave>` (Li et al., 2023c), an interval can also be considered as `<octave-interval>` and `<interval-class>` as presented in Figure 6.5.

Such an intervalization strategy may allow to directly disentangle octave relations between the notes and interval classes.

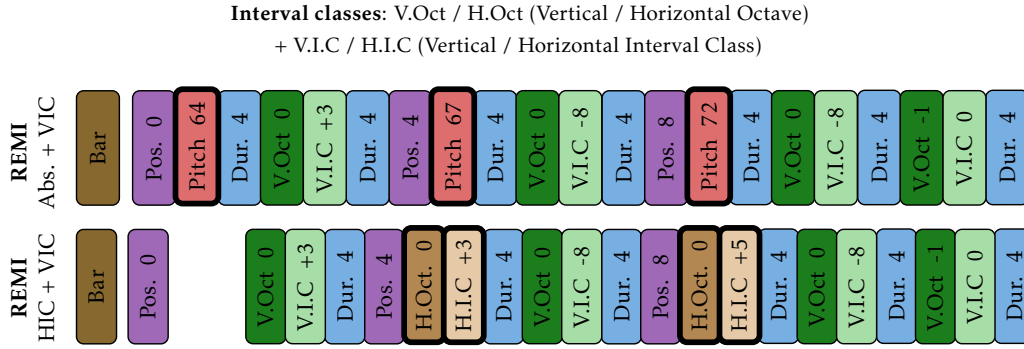


Figure 6.5: Examples of intervalized tokenizations based on interval classes instead of pitch intervals. (Abs.: *Absolute pitch encoding*)

Moreover, while interval-based tokenizations can be used for analysis tasks, they cannot all be used for generation purposes, in particular for tokenizations where I_{ref} are horizontal pitch intervals because the generated sequence will not refer to a unique musical sentence. Consequently, tokenizations can be considered where absolute pitches are periodically indicated (for example, at the beginning of each bar), and with intervals representing horizontal/melodic and vertical/harmonic relationships relative to this periodic absolute reference.

Finally, we have only considered tokenizations based on REMI. However, the intervalization process can be applied to any tokenization strategy in which pitches are encoded as absolute values. Time-related tokens (*i.e.* <Bar> and <Position> for score-based tokenization, or <Time-Shift> for performance-based tokenization (Oore et al., 2018)) are not affected by pitch encodings. Therefore, a study comparing both time and pitch representations may show possible combinations of encodings resulting in a better modeling of symbolic music.

However, beyond interval-based tokenization, taking action on the alphabet – the very first component of most MIR study pipelines – might be heavily mitigated by the choice of the following elements of the pipeline, such as token grouping or going further beyond the representation aspects, the model architecture and mechanisms. In other words, although enhancing the alphabet's expressiveness leads to some improvements on our selected tasks and models, it may remain too minor a factor to have a meaningful impact for more general tasks, such as music generation. In the following section, we take a step further by examining the expressiveness of *grouping* strategies applied to elements of the alphabet.

6.2 Analyzing byte-pair encoding for monophonic and polyphonic music

This section is based on a work published at the workshop *NLP for Music and Spoken Audio (NLP4MusA)* co-located with ISMIR 2024 (International Society of Music Information Retrieval) (Le et al., 2024).

As described in Chapter 2, text disposes of natural ways to group characters into words, such as using whitespaces to separate groups of characters in several languages. In contrast, music lacks such explicit mechanisms for grouping elements of the alphabet into coherent “musical words”.

In NLP, words learned using a data-driven way often rely on subword tokenization algorithms such as **Byte-Pair Encoding (BPE)**. The algorithm relies on creating new subword tokens by iteratively merging the most recurring pairs of successive tokens in a corpus until a chosen vocabulary size is reached. In the following, we call *atomic elements* the tokens from the initial vocabulary and *supertokens* the tokens added to the vocabulary through BPE (Figure 6.6).

BPE has been applied to symbolic music tasks, as described in Section 4.1.2.1. It has been typically implemented to shorten token sequences (Liu et al., 2022), or evaluated empirically as a pre-processing step for music generation (Fradet et al., 2023a) or analysis tasks (Zhang et al., 2023). However, the resulting vocabulary and the impact of the type of music data given to the algorithm have not been thoroughly studied. Therefore, this work explores two aspects of BPE applied to symbolic music, seen as a mean of creating more *expressive* musical tokens:

- We first propose a statistical analysis of the vocabulary of tokens obtained from symbolic music. We examine its behavior in comparison to BPE applied to textual data, and we apply BPE on diverse types of music with various degrees of polyphony.
- Some musical properties captured by BPE reflected through this first analysis then prompts a quantitative analysis of the effects of BPE on a model trained on a task of musical phrase segmentation. In particular, we study its impact on monophonic and polyphonic piano music.

6.2.1 Analyzing music byte-pair encoding

In this section, we present analyses of the vocabulary produced by BPE when applied to text and music. We first analyze supertokens resulting from text-BPE and

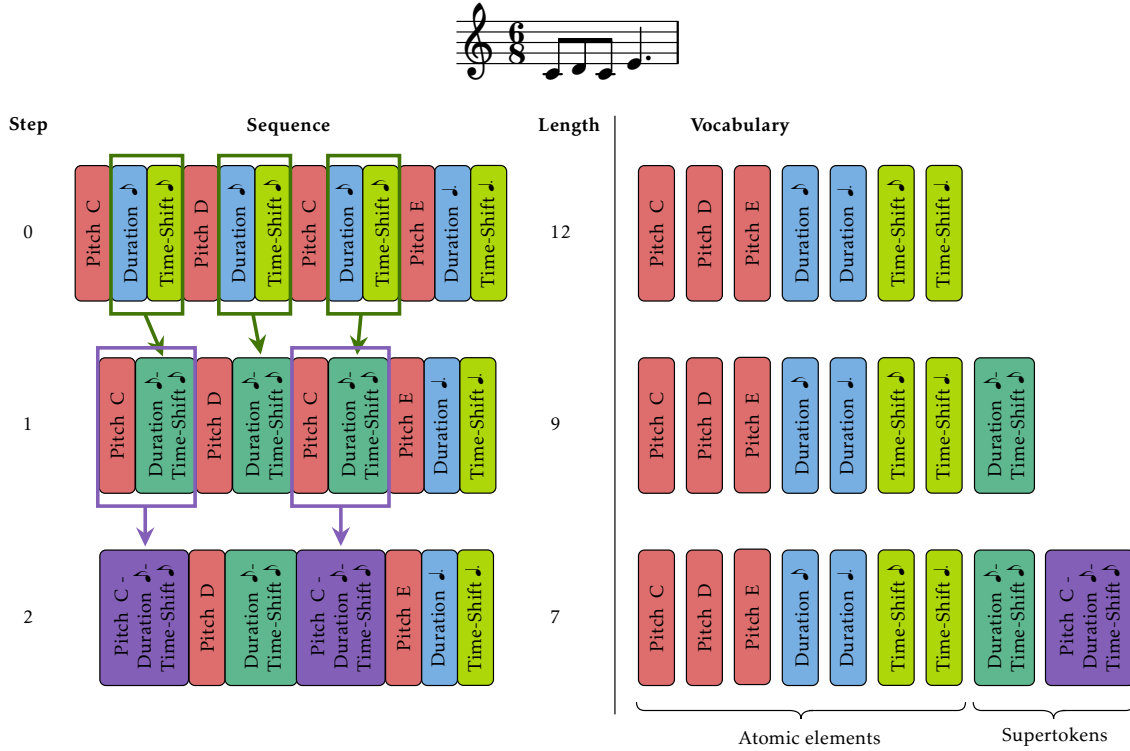


Figure 6.6: Byte-Pair Encoding (BPE) algorithm applied to music tokens using a TSD-like tokenization (Fradet et al., 2023a). The most recurring pair of tokens is merged iteratively to create new *supertokens*, as a combinations of atomic elements with possible already created supertokens. BPE jointly results in decreasing the length of sequences while increasing the size of the vocabulary.

music-BPE with various types of instrumentation. We qualitatively observe that supertokens can convey high-level musical content for phrase segmentation, which in turn motivates a quantitative analysis of the impact of BPE on this task.

6.2.1.1 Comparing text and music BPEs

Musical notes are often compared to text at the level of characters (Hirata et al., 2022). Deep learning models have been shown to be more efficient when dealing with characters grouped into (sub)words (Shapiro and Duh, 2018; Tay et al., 2022). Therefore, we study the BPE results when processed, on text and music, in order to observe common or distinctive operating regime on such data with various languages and instrumentations. Text data includes alphabetic² languages from various regions, extracted from the XLNI dataset (Conneau et al., 2018b) on which we run BPE on 100k premises. For music, we compare monophonic folk tunes, classical piano, string

²Experiments have also been conducted on syllabic (Japanese) and logographic (Chinese, Korean) languages, that show major differences due to the different nature of the atomic elements of their initial vocabulary.

quartet, and orchestral corpora with similar sizes and tokenize these datasets using REMI (Huang and Yang, 2020) from which `<Velocity>` tokens are removed.

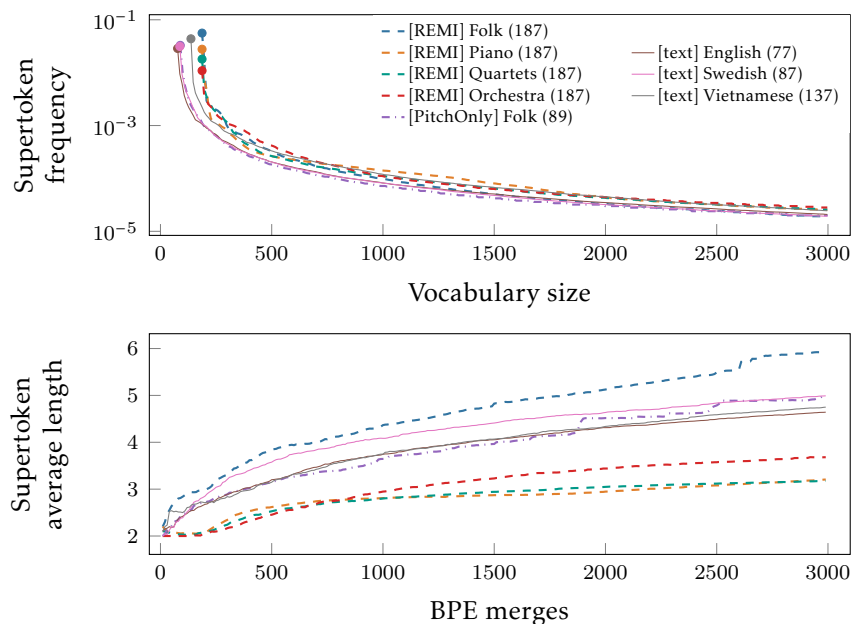


Figure 6.7: (*Top*) Frequency of the created supertokens through the vocab size increasing with the BPE steps, for different styles of music and multilingual text data. (*Bottom*) Average length of already created supertokens through BPE iterations for musical and text data. The initial vocabulary size of each tokenization is indicated.

We first study the occurrence frequency of the newly created supertoken within the corpus, at each step of the training (Figure 6.7, top). To make the corpora and vocabularies comparable, supertoken frequencies are normalized by the initial corpus length, and the BPE iterations are aligned with the resulting vocabulary size. Interestingly, the vocabularies obtained on music or text through BPE do not show major differences with respect to the decay rate or the order of magnitude of the frequencies.

We also compute the mean length of the supertokens through the BPE steps (Figure 6.7, bottom). The evolution of supertoken length differs between text and music, depending on the instrumentation. While monophonic supertokens are generally longer than polyphonic ones, orchestra supertokens surprisingly appear to be longer than piano or string quartet ones. An in-depth study of the constructed vocabulary shows that the orchestral vocabulary predominantly consists of “harmonic” supertokens formed of simultaneous notes. In contrast, piano and string quartet vocabularies include both simultaneous and consecutive notes. This difference causes BPE to struggle to build long piano or string quartet supertokens. On a separated experiment, we observed that it takes over 10 times more steps on a piano corpus to get an average length comparable to that of the vocabulary obtained on the monophonic corpus. This assumption could be quantified through further

experiments by comparing supertoken lengths in terms of “musical length” (*i.e.* the number of sub-beats included in the supertoken) instead of raw counts of aggregated atomic elements.

Moreover, when considering an alphabet which only keeps `<Pitch>` tokens, we show that monophonic supertoken lengths have a regime closer to that of text for this range of BPE merges (Figure 6.7, “PitchOnly” curve), while polyphonic curves still stand out. We can thus posit that the differences between the music and text curves might be due to simultaneity and timing information, which are inherent to music.

6.2.1.2 Musical content carried by supertokens

So far, we have drawn a broad characterization of the BPE vocabularies, let us now zoom in and try to delineate which supertokens are present in a specific context. Borrowed from text, the terms “musical phrase” or “musical sentence” (Nattiez, 1990) denote a part of the music which can give the impression of a complete statement by its own. The TAVERN dataset (Devaney et al., 2015) include such phrase annotations for theme and variations for piano by Mozart and Beethoven. Similarly to text words or expressions that may have particular places within a sentence (*e.g.* link words such as “On the one hand, ...” at the beginning of sentences, or tag questions such as “..., isn’t it?” at the end), we analyze the relation between musical supertokens built via BPE and musical phrases.


In addition to REMI, we perform this analysis using a Structured (Hadjeres and Crestel, 2021) tokenization with pitches encoded as intervals (Section 6.1) with x_{ref} being the skyline notes. In contrast with the REMI tokenization used in the last and following sections, we use the Structured tokenization in order to take advantage of both Structured’s relative encoding of rhythm with `<Time-Shift>` tokens and the relative encoding of pitches through `<Horizontal_PitchInterval>` and `<Vertical_PitchInterval>`. We considered a 1024-merge BPE.

Tokenizer	Overlapping supertokens (↓)	
	Random split	Phrases
Absolute REMI	58%	47%
Absolute Structured	69%	1.3%
Interval REMI	60%	32%
Interval Structured	71%	4.2%

Table 6.3: Ratio of overlapping supertokens between two phrases in the TAVERN dataset for multiple tokenization strategies. This ratio is compared with the ratio of overlapping supertokens when pieces are split randomly.

Supertokens and phrase boundaries – A first observation is that supertokens are not likely to overlap musical phrases (Table 6.3). We consider the ratio of supertokens built through BPE that overlap two musical phrases, in comparison with this ratio if the piece was split randomly with the same number of chunks as the BPE segmentation. Using the Structured tokenization, both with absolute- or interval-based pitch encodings, less than 5% of the supertokens among the tokens of the sequences do overlap phrases. In contrast, randomly splitting the piece results in a 69% to 71% overlap ratio, indicating that supertokens are unlikely to span across phrase boundaries. This phenomenon is less marked with REMI but the segmentation in musical phrases still decreases the ratio of overlapping supertokens compared to a random segmentation.

< Duration(4), TShift(4), Horizontal_PitchInterval(+5) >



< Duration(1), TShift(1), Horizontal_PitchInterval(-5),
Duration(1), TShift(1), Horizontal_PitchInterval(-3),
Duration(1), TShift(1), Horizontal_PitchInterval(-4) >

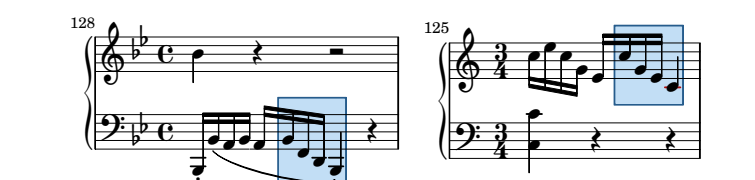


Figure 6.8: (Top) First most common start-of-phrase supertoken from Mozart’s 7 *Variations on “Willem von Nassau”* (K 25), Variation VII and Beethoven’s 12 *Variations on “Menuet a la Vigano”* (WoO 68), Variation VII.

(Bottom) 9-long common ending supertoken (10th most common) from Beethoven’s 10 *Variations on “La stessa, la stessissima”* (WoO 73), Variation VII and Mozart’s 12 *Variations on a Minuet by Fischer* (K 179), Variation IV. The tokenization is Structured + intervals.

Start & end-of-phrase supertokens – We then analyzed the supertokens occurring at the beginning and end of musical phrases within the TAVERN dataset. In particular, choosing the Structured tokenization allows this analysis to be key signature-independent and bar position-independent. The most recurrent start-of-phrase supertoken appears to be a melodic rising perfect fourth (Figure 6.8, top), which follows musicology studies (Meyer, 1973, p.145):

An upbeat interval of a perfect fourth, moving to the tonic [...] may be understood as a rhythmic-harmonic event emphasizing the tonic on which the melody proper begins.

Most represented end-of-phrase supertokens include descending arpeggio patterns on the tonic chord (Figure 6.8, bottom). This also verifies some musicological

observations (Huron et al., 1996):

Melodic passages tend to exhibit an arch shape where the overall pitch contour rises and then falls over the course of a phrase or an entire melody.

Similar to how BPE can capture syntactic rules in text, this qualitative analysis shows that musical supertokens also convey high-level musical information when considering their relation with musical phrases.

The initial analysis of BPEs across different instrumentations suggests that vocabularies derived from monophonic and polyphonic corpora differ (Section 6.2.1.1). Additionally, the second qualitative analysis indicates that BPE may capture musically relevant information for phrase segmentation (Section 6.2.1.2). These preliminary findings therefore motivate a more quantitative evaluation of the impact of BPE on monophonic and polyphonic data in the context of a musical phrase segmentation task.

6.2.2 Evaluating BPE on musical phrase segmentation

In the literature, BPE applied to MIDI-derived tokenization has been mainly evaluated on music generation (Fradet et al., 2023a) or sequence classification, in particular composer classification, on a general multi-track dataset (Fradet et al., 2023a) or specifically piano music (Zhang et al., 2023). Given our preliminary results from the previous sections, we aim to quantitatively evaluate BPE specifically on a task of musical phrase segmentation for monophonic and polyphonic datasets.

From a technical standpoint, our experiments rely on the MidiTok package (Fradet et al., 2021) to handle the tokenization of MIDI files, and use Transformer model implementations provided by the HuggingFace library (Wolf et al., 2020).

6.2.2.1 Task & data

We consider the *musical phrase segmentation task* presented in Section 6.1.2.1, where a model is trained to tag each token of a sequence as being a start-of-phrase or not. For BPE sequences, if a start-of-phrase occurs within a supertoken, the whole supertoken is annotated as being a start-of-phrase.

We perform this task on two types of music: monophonic and polyphonic piano music. For this purpose, we first performed this task on the MTC dataset (Van Kranenburg et al., 2014) composed of monophonic Dutch folk tunes and including phrase annotations. For the choice of a polyphonic dataset, while the TAVERN dataset (Devaney et al., 2015) does include phrase annotations, the comparison would be less fair given its limited size: the MTC dataset includes 101 times more phrases than

TAVERN. Moreover, the nature of classical-style musical phrases, generally based on cadences (Spencer and Temko, 1994), may differ from folk music phrases, based on melodic contours (Huron et al., 1996). Therefore, for a fairer comparison, we discard TAVERN as our polyphonic dataset and we rely on the MTC-Piano dataset presented in Section 6.1.2.2. It consists in a synthetic dataset of folk music piano arrangements from the MTC dataset generated by the AccoMontage model (Zhao and Xia, 2021) aligned with the original phrase annotations, which gives a dataset of polyphonic piano music with folk music phrases which is 55 times larger than TAVERN. We tokenize both datasets using REMI (Huang and Yang, 2020) and remove the Velocity tokens, wishing to focus on the impact of BPE on pitch and rhythm information.

Note that the non-BPE dataset is by design more unbalanced than the BPE one. In the polyphonic setting, the proportion of start-of-phrases increases from 1.2% in the whole dataset to 3.3% after 128k BPE merges. In contrast, for the monophonic dataset, only 2% of tokens are annotated as start-of-phrases without BPE while 27% of (super)-tokens are considered as start-of-phrases after the same number of BPE merges.

6.2.2.2 Results

We trained a 2-layer Transformer encoder-only model with 8 heads per layer and a common embedding size between BPE and non-BPE vocabularies on each dataset. In contrast to Section 6.1, where we use a 3-layer model, we reduce the size of the part dedicated to Transformer encoders to compensate for the increased embedding size resulting from the larger vocabulary. We evaluate each model on 3 different splits of the datasets, using the F1-score of the start-of-phrase label prediction. As our experiments focus on representation impact, we chose to have light models rather than ones achieving optimal performance.

Performance on the task – The polyphonic setting of our experiment seems to indicate that BPE can have an impact on performance. Indeed, unlike Zhang et al. (2023) also focusing on piano music, who demonstrated on a sequence global classification task that a BPE (with the initial vocabulary size $\times 4$) does not result in significant improvements, we see on this token classification task that the performance increases with the number of merges (Figure 6.9, top).

Our results on the monophonic dataset show even that BPE with too few number of merges can degrade the performance (Figure 6.9, bottom). This surprising behavior also occurs in NLP tasks, where character-based models can outperform subword-based models (Chung et al., 2016).

Content of the supertokens – Figure 6.10 describes the “melodic” content of the supertokens created along BPE steps, *i.e.* the proportion of supertokens having $n < \text{Pitch}$ atomic elements. An analysis of supertokens reveals that early merges tend to produce *structural* supertokens, such as combinations of $< \text{Bar} >$ and $< \text{Beat} >$

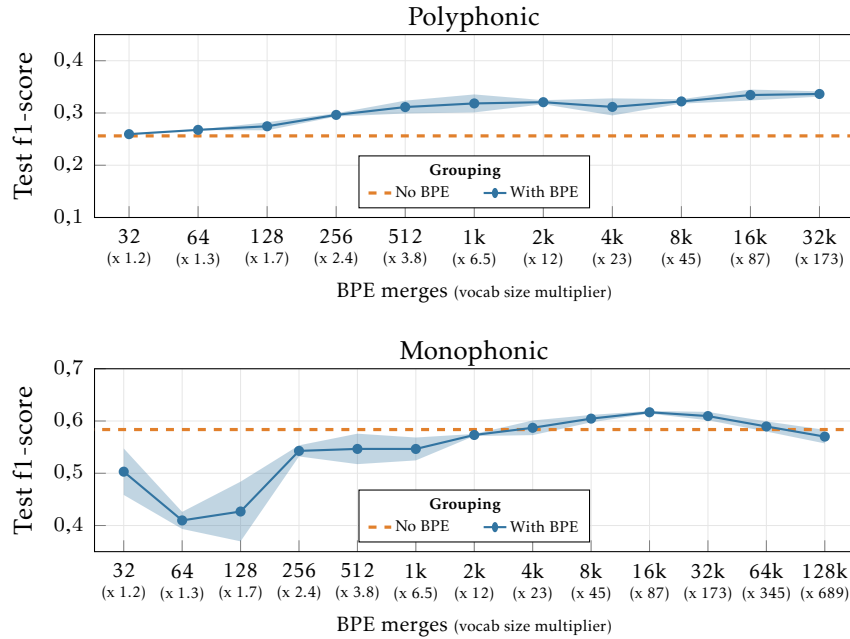


Figure 6.9: Performance of the models improved with BPE on the task of start-of-phrase detection with polyphonic and monophonic datasets. f1-score for start-of-phrase classification on the polyphonic dataset (*top*) and on the monophonic dataset (*bottom*) averaged over the 3 splits.

(Figure 6.10 gray area: proportion of created supertokens with 0 <Pitch> atomic element), while melodic patterns emerge later, but at different rates for monophonic and polyphonic datasets.

At 128 merges (Figure 6.10, dashed line), 26% of monophonic supertokens do not include any <Pitch> atomic element (gray area). In contrast, this ratio is only 9% for polyphonic, most of the supertokens are composed of 1 <Pitch> atomic element (yellow area) and 7% already contain 2 <Pitch> atomic element (green area). Fewer melodic patterns, which are more likely to indicate phrase boundaries in monophonic tunes (Huron et al., 1996), may explain why the BPE model performs better only after a certain number of merges.

In the monophonic dataset we also see that, after too many merges, the model performance drops (Figure 6.9, bottom). An analysis of the supertoken length shows that, after 128k merges (not shown), monophonic supertokens are on average 38.6-long (compared to 8.4 for polyphonic ones). Indeed, the smaller size of the monophonic dataset (3 times smaller than the polyphonic one) leads late steps supertokens to capture long but rare patterns that might be less relevant for this task of phrase segmentation.

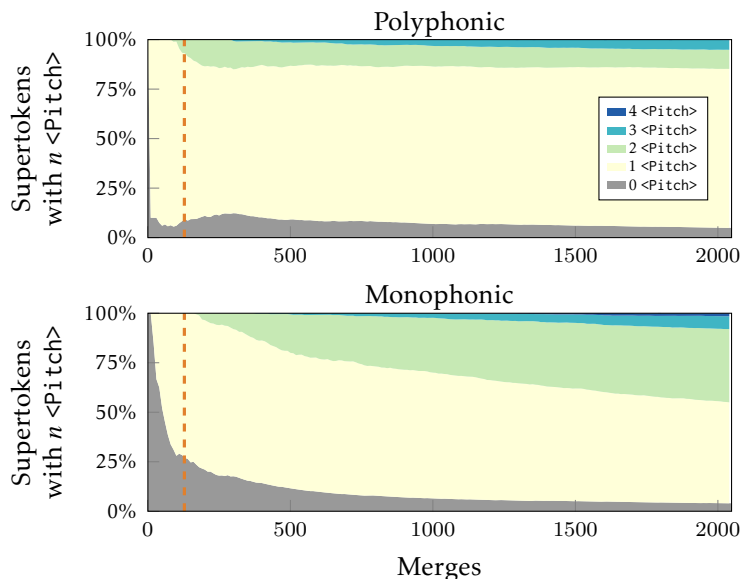


Figure 6.10: Ratio of supertokens containing n `<Pitch>` atomic elements in the vocabulary for each number of BPE merges.

6.2.3 Towards improving token grouping expressiveness

In this last section, we show that BPE behaves differently depending on the type of music it is trained on, both with a descriptive and a quantitative approach through a downstream task. More precisely, on this task, we confirm the impact of instrumentation on the model performance and show that the number of BPE merges should be chosen carefully. In particular, this hyper-parameter can be an obstacle, as certain cases may require a large number of BPE merges, leading to additional tokenization training time. While our study outlines general trends in BPE behavior, identifying a rule of thumb for the number of merges based on data or task would be a major improvement towards the practical use of subword tokenization for symbolic music.

Moreover, while we focus on REMI tokenization, the initial tokenization itself may significantly influence the effects of BPE or other subword strategies. Unlike text, musical supertokens can vary greatly depending on both the tokenization method and the underlying data. Moreover, our results show that BPE primarily aggregates temporal structural information, while tokenization strategies represent time in different ways (Section 4.1.2), such as through absolute positions using combinations of `<Bar>` and `<Position>` in REMI, or a single `<Time-Shift>` in MIDI-like tokenization. Thus, beyond the number of merges, the choice of initial tokenization is also a key factor to consider for the application of BPE.

6.3 Tokenization expressiveness or model power?

In this chapter, we explored how improving the expressiveness of event-based tokenization may improve a model performance on downstream analysis tasks. This can be done through a choice of musically relevant atomic elements of the *alphabet* such as intervals or by implementing *grouping* strategy such as BPE.

However, in these works, we fixed the model architecture so that only the tokenization is the impactful factor. A thorough investigation into the relationship between music representation and model architecture and size remains essential. Indeed, while choosing a judicious tokenization can possibly lead to better performance, a larger model or equipped with particular mechanisms may result in similar performances with “weaker” tokenization. In the same way that sufficiently large LLMs can learn meaningful linguistic features through generic pre-training tasks (Clark et al., 2019), a large enough model processing symbolic music might also be capable of learning by itself musically relevant token groupings or musical features that could otherwise have been defined *a priori* in the alphabet. We believe that a quantitative evaluation of the performance gained thanks to the tokenization choice or the model size may be a promising avenue to explore for future works.

Chapter 7

Exploring attention-based model inner mechanisms

7.1	Model explainability through mechanistic interpretability	104
7.2	Models for functional harmony analysis	105
7.2.1	Functional harmony analysis	105
7.2.2	An end-to-end model for roman numeral analysis	107
7.2.3	Comparing hidden states from pre-trained and end-to-end models for RNA	109
7.3	Analyzing attention behavior	111
7.3.1	Attention span	112
7.3.2	Attention span & harmonic diversity	114
7.4	Understanding attention heads' relevance	115
7.4.1	Layer-wise relevance propagation	116
7.4.2	Attention heads' relevance and musical tonality	119
7.4.2.1	Hypotheses between LRP maps and local tonalities	120
7.4.2.2	Evaluating the hypotheses between LRP maps and local tonalities	123
7.4.3	Is tonality learnt during pre-training? Applying LRP to a MLM	129

Attention-based models have demonstrated state-of-the-art performance in several analysis and generative tasks in symbolic MIR. However, they are often considered as black boxes. This aspect may limit one's ability to understand, validate and trust a model's decision. For instance, while human-AI co-creative tools are increasingly common, musicians are more likely to trust and engage with models that provide interpretable feedback aligned with established musical theory or practice.

In this chapter, we explore models under the realm of *explainability* (Section 7.1). To this end, we present an end-to-end model for functional harmony analysis (Sec-

tion 7.2) in which we aim to perform model explainability, in particular through *mechanistic interpretability*. More precisely, we focus on the attention mechanism, by analyzing the patterns within attention matrices (Section 7.3) and the relevance of attention heads in particular in the context of local key detection (Section 7.4). The work presented in this chapter is still ongoing and includes incomplete, unfinished, or currently unexplored research directions.

7.1 Model explainability through mechanistic interpretability

An important domain in artificial intelligence research focuses on *model explainability* or *interpretability* and aims at understanding complex models. They are often considered as “black boxes” because their training often results in too much complex states to be easily interpretable. From a technical standpoint, such interpretability can be performed through various ways (Bereska and Gavves, 2024):

- **Behavioral interpretability** focuses on input-output relationships, independent of the model’s internal structure.
- **Attributional interpretability** links output decisions to specific features characterizing the input (*e.g.* specific dimensions of a multi-dimensional input).
- **Concept-based interpretability** uses probing tasks different from the original task to assess whether the model has captured high-level concepts
- **Mechanistic interpretability** analyzes a model through its inner mechanisms, such as connections between layers, hidden states, or self-attention for Transformer-based models.

In NLP, several studies have focused on model explainability (Zhao et al., 2024a). In particular, *mechanistic interpretability* studies have focused on the attention mechanism. Studies have shown that attention heads play different roles in a task of machine translation and specifically focus on particular information such as token positions, syntactic structures of rare tokens (Voita et al., 2019). For BERT models, these heads can attend to specific delimiter tokens such as punctuation or syntactic objects such as verbs or nouns (Clark et al., 2019). The importance of these different attention matrices is also task-dependent, with their contributions varying across downstream applications (Kovaleva et al., 2019).

Models’ inner mechanisms can be explored following this global framework:

- (a) Select a feature derived from an internal mechanism of the model to analyze.

- (b) Choose a dataset and segment it according to discriminative characteristics. In our case, we consider musically relevant characteristics.
- (c) Compare the behavior of the selected feature across the different classes or types of data defined by those characteristics.

In the following, for (a), we consider two features: attention span extracted from the attention matrices (Section 7.3) and attention head relevance (Section 7.4).

In-depth analysis of model interpretability remains under-explored in MIR (Section 9.1.3). It is typically treated as a secondary finding rather than a central research objective (Dong et al., 2023). In our work, for the item (b) of our presented framework, we specifically focus on a model’s self-attention mechanism interpretability in the context of *functional harmony analysis*. This task suits well a framework of attention analysis as it encompasses the analysis of long-term attributes such as local key detection or chord segmentation, and short-term attributes such as chord degree, quality or inversion identification.

7.2 Models for functional harmony analysis

We first present models considered in this chapter trained for functional harmony analysis. This task aims at labeling chords through their *function* within a tonal context. We then present an end-to-end model trained on this task and we compare its behavior with a pre-trained model.

Our model does not achieve state-of-the-art performance, but its results remain comparable and within a competitive range. We then use this model in Sections 7.3 and 7.4 to illustrate the core contributions of this chapter, namely tools for Transformer-based model explainability through the prism of the attention mechanism.

7.2.1 Functional harmony analysis

In the context of Western tonal music, functional harmony is a system of rules designed to give direction to chord progressions by assigning *functions* to these chords in relation to a key (Caplin, 1998). This creates effects such as harmonic tension or release (Lerdahl and Jackendoff, 1996) with particular patterns such as cadences. This theory plays a central role in tonal music, as it is often compared to grammatical structures in language, as previously explained in Section 2.2.2.

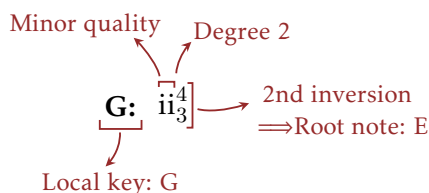
More formally, a chord function can be characterized by multiple features, including these main characteristics:

- The **local key** or local tonality, for example C major. It can be different from the global key of the piece and can change throughout a piece due to modulations. In the following, “key” and “tonality” will be used interchangeably.
- The chord **degree** characterizes the position of the chord in relation to the local key. For example, the fifth degree in a C major key is based on the note G.
- The **quality** of the chord which specifies whether the chord is major, minor, dominant,
- The chord **inversion** specifies the vertical order of the notes of the chord. For example, a second inversion of a C major chord is defined with a G as root note.
- The **root note** of the chord designates the lowest note of the chord. It can differ from the root note due to the inversion, for example when a note is held as a pedal.

From a computational standpoint, other features such as harmonic rhythm, pitch class set or 4-part voices identification, can be added for better performances (Chen and Su, 2018; Nápoles López et al., 2021).

Chord roles are often annotated with roman numeral labels (Figure 7.1a), so that functional harmony analysis is often referred to as Roman Numeral Analysis (RNA). The roman numeral reflects the characteristics of the chord, as shown in Figure 7.1b. From these annotations, common patterns can be defined. For example, a perfect cadence depicts a sequence of V to I chords, both in root position. In particular, such patterns are defined *regardless of the absolute key signature* because chords are defined by their degree *relative* to the local key signature. For further insights into roman numeral analysis, refer to (Benward, 2018).

(a) Reduction of the beginning of the Bach's Prelude in C Major (BWV 846). The RNA is written at the bottom of the staff, with their equivalent analysis using absolute pitch annotations at the top.



(b) Example of a roman numeral annotation describing the features of a chord.

Figure 7.1: Roman numeral annotations used for functional harmony analysis.

These annotations are however prone to annotator disagreement (*e.g.* a same set of notes can be interpreted differently based on the context) with various levels of discrepancy, from chord function to the actual roman numeral label (Devaney et al., 2015; Koops et al., 2019).

Functional harmony analysis in MIR – Multiple machine learning models have been developed to perform automatic functional harmonic analysis. To this end, most studies have considered functional harmonic analysis as a multi-objective task. Such sub-task describes a particular feature of the chord, which are then put in relation in order to derive the roman numeral label. This multi-task framework has been then widely used to perform functional harmonic analysis with models such as recurrent networks (Chen and Su, 2018; Micchi et al., 2020, 2021), convolutional networks (Nápoles López et al., 2021; Nápoles López, 2022), graph neural networks (Karystinaios and Widmer, 2023) or with audio data (Fricke et al., 2024).

Transformers have also been used to perform this task. Harmony Transformer (Chen and Su, 2019, 2021) is a model based on Transformer-based encoder-decoder. The encoder aims at dividing the score into chord regions, from which the decoder predicts the corresponding roman numeral label.

RNBert (Sailor, 2024) is a Transformer-based encoder-only model which relies on a pre-trained MusicBERT (Zeng et al., 2021). It is fine-tuned on RNA on 8 sub-tasks and comes in two versions: an unconditioned one, and a key-conditioned version – either using teacher forcing or an initial key detection step – followed by post-processing steps (*e.g.* a Viterbi algorithm to avoid too brief key changes). It is trained on a dataset of 1.4k scores with RNA annotations. Regarding representations, RNBert includes a conversion between the event-based tokenization of MusicBERT based on Octuple and a time-slice-based tokenization used for RNA. Using a time-sliced-based tokenization (Section 4.1.1) is necessary for RNA because a note can change its tonal function through time, requiring it to be annotated with different labels at different moments.

7.2.2 An end-to-end model for roman numeral analysis

In this section, we present the technical details of the model performing RNA, based on a BERT architecture, as well as the performances reached on this task.

Representation – While BERT models for symbolic music are usually trained using an event-based tokenization like MIDI or REMI, such a tokenization is not suited for harmonic analysis. A single event, such as a single note, can span over multiple chord changes, so that such event must be labeled differently through time. Therefore, we chose to use the representation of AugmentedNet (Nápoles López, 2022, §5.1.4) based on regular time slices quantized at the thirty-second note. This representation includes three features:

- *Chromagram*. The chroma information is represented as a slice of dimension 19, with 12 dimensions for the pitch class and 7 for the note letter (C, D, ..., B). The note letter corresponds to the pitch spelling to disambiguate enharmonics. In particular, the pitch class and note letter parts can be multi-hot when simultaneous notes occur.
- *Lowest sounding note*. The encoding of the lowest sounding note relies on the same representation as the chromagram, but by definition, the pitch class and note letter parts are both one-hot vectors.
- *Measure and note onsets*. The onsets are split into two parts: two 7-dimensional vectors, one for the onset of measures and one for the onsets of notes. One of the dimension represents an onset (of a note, or a measure), and the six other dimensions represent the time elapsed since the onset, measured in note durations (from ♩ to ♩).

These three features are then embedded independently and concatenated to give the embedding of the time-slice-based token given as input to the Transformer encoders. Using this representation, the chosen sequence length of 512 tokens represents a 16-bar long music in $\frac{4}{4}$.

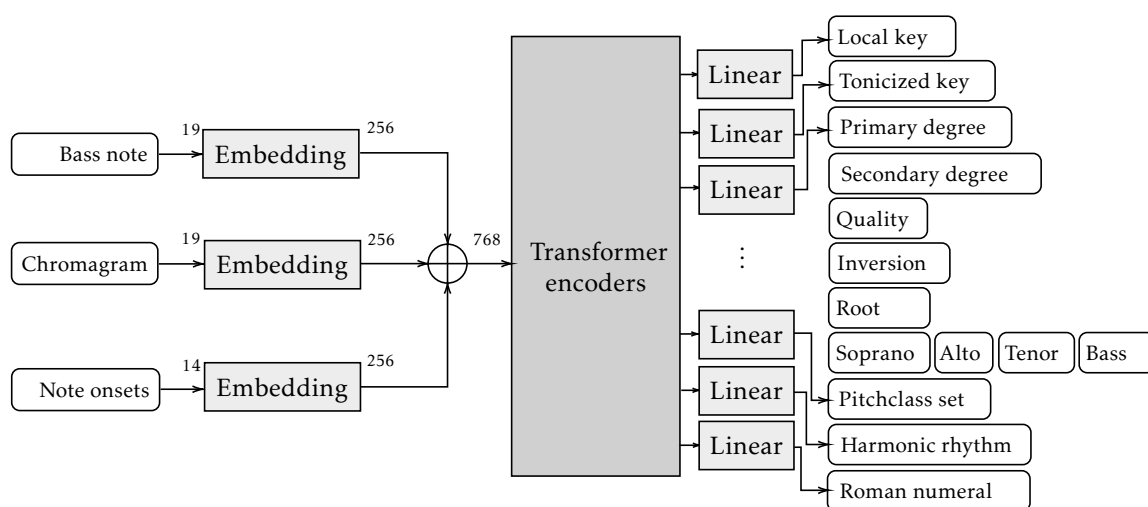


Figure 7.2: Model architecture for roman numeral analysis based on Transformer encoders, trained as an end-to-end model. “Bass note”, “Chromagram” and “Note onsets” are multi-hot vectors, as presented in [Nápoles López \(2022, §5.1.4\)](#).

Architecture – The architecture of our model is based on a stack of Transformer encoders, following the BERT architecture. The model is followed by a classification module for the multi-classification task. We keep 14 sub-tasks as represented in [Figure 7.2](#). Our model for harmonic analysis is a stack of 12 layers of encoders, each one being composed of 12 attention heads (*i.e.* 144 heads in total) and processes 512-long sequences, following the architecture of the original BERT model ([Devlin](#)

et al., 2019). Its implementation is based on the the HuggingFace library’s (Wolf et al., 2020) BERT model from which the embedding layer specific to text is removed.

Training – We use the dataset and train/valid/test splits from (Nápoles López, 2022) which compiles multiple datasets with roman numeral annotations. From a musical viewpoint, it spans from early works such as Bach chorales and Well-Tampered Clavier (Gotham et al., 2019) to early romantic pieces with Schubert lieder (Gotham and Jonas, 2022). Regarding instrumentations, it is mostly composed of piano solo (notably Beethoven and Mozart piano sonatas) but is limited to at most four voices with string quartets (Haydn string quartets) or chorales. It is composed of 596 pieces for a total of 2.4M tokens in the full dataset. We perform data augmentation by transposing each piece in multiple keys, resulting in 28.3M tokens. Too short sequences are padded with a special token and an attention mask is applied on these <PAD> tokens.

Following Liebel and Körner (2018), we implement a multi-task loss which is defined as a weighted sum of individual classification losses from each task. These weights associated with the loss are learned simultaneously with the model training. For the following model explainability experiments, we used the end-to-end model as justified in Section 7.2.3. We further explore a pre-trained model in Section 7.4.3.

Performances – A comparison between the performance of multiple multi-task models for functional harmonic analysis is presented in Table 7.1. Comparisons with previous models (Chen and Su, 2018; Micchi et al., 2020; Chen and Su, 2021) are not shown, as these models were evaluated on different test sets. In particular, as depicted by (Chen and Su, 2018), our experiments confirm that a multi-task learning framework results in better performances than mono-task models in the context of a Transformer-based model.

Our model outperforms those of previous studies at the time of this work (*i.e.* without RNBert (Sailor, 2024)). It does so however by stacking 12 layers of 12 heads Transformers, which is about 900 times larger than AugmentedNet which is based on a convolutional network. Therefore, we also evaluate a Transformer-based model composed of 3 layers of 3 attention heads which has a similar number of parameters as ChordGNN in order to compare performances between the two types of architectures. Nevertheless, the focus of our current study is not on model lightness versus performance balance but on *explainability* for which this high performing model is well suited. The experiments described in the following sections use our larger model, which seems better suited to analyze the behavior of attention mechanism.

7.2.3 Comparing hidden states from pre-trained and end-to-end models for RNA

We first aim to assess whether the decision-making process for a functional harmony label is distributed across all layers of the model or primarily concentrated in the

Model		Key (38)	Degree (22)	Quality (11)	Inversion (4)	Root (35)	Avg.*	# params
AugmentedNet v1.9.1 (Nápoles López, 2022)	(11 tasks)	82.2	67.0	79.7	78.8	83.0	–	105k
ChordGNN (Karystinaios and Widmer, 2023)	(10 tasks)	81.3	71.4	78.4	80.3	84.9	–	5.88M
RNBert (Sailor, 2024)	(8 tasks)	82.5	85.9	86.5	87.2	–	–	109M
Our model (mono-task)	(14 tasks)	79.5	81.7	90.0	89.7	92.0	87.5	92M [†]
Our model (multi-task) (light)	(14 tasks)	77.4	73.5	86.1	87.2	89.2	83.3	5.08M
Our model (multi-task)	(14 tasks)	81.7	84.3	90.9	90.3	92.8	88.6	94.9M

Table 7.1: Accuracies of multi-objective RNA models on 5 sub-tasks and averaged over all sub-tasks. For each sub-task, we indicate the number of classes corresponding to the sub-task. The models are evaluated on an identical test set (Nápoles López et al., 2021). Performances from comparative models are extracted from the original papers. (*) We report the average of the accuracies on all the tasks performed by the model. (†) The indicated size concerns one model without its classification head specific to the task. These classification heads are composed of about 200k parameters depending on the task.

final ones. This will provide insight into which parts of the model likely to be relevant for further explainability analysis. For this purpose, we implement **logit lens** of RNBert (pre-trained model) and our end-to-end model.

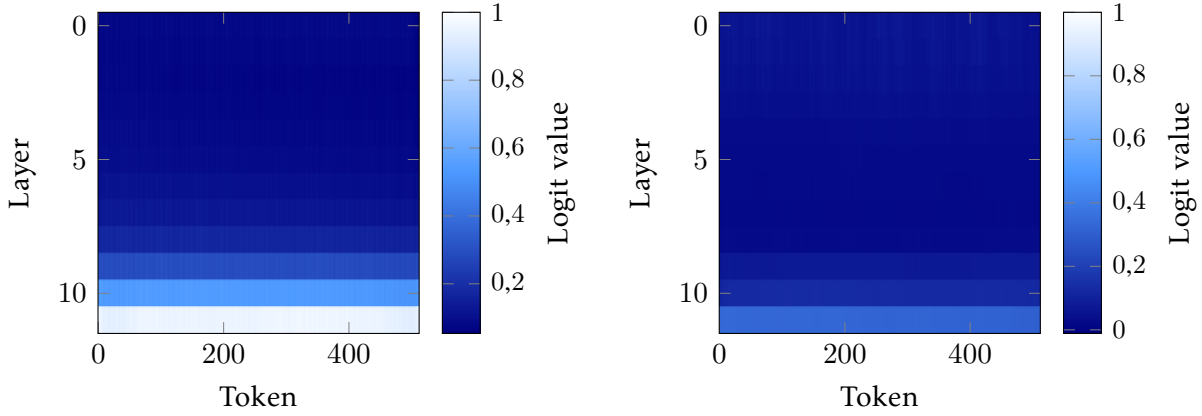
Logit lens (nostalgebraist, 2020) is a method aiming at examining the intermediate activations of hidden layers of a model by translating them into the output vocabulary using the model’s output layer as translator. Mathematically, for hidden states $h_i \in \mathbb{R}^d$ resulting from a layer i and a vocabulary of size v , we plug the final layer $W_{\text{out}} \in \mathbb{R}^{v \times d}$ in order to get logits as if they were computed at this early stage of the model.

$$\text{LogitLens}(h_i) = W_{\text{out}} \cdot h_i \in \mathbb{R}^v$$

This method can be refined, for example with TunedLens (Belrose et al., 2023), but we chose to rely on LogitLens as an initial baseline.

Figure 7.3 shows the logit values $\text{LogitLens}(h_i)$ of the predicted labels that would have been obtained for a given sequence if the lens had been applied at each layer i , for two different models: our end-to-end model (7.3a) and the pre-trained + fine-tuned RNBert (7.3b). For comparison, the logit values are normalized by the maximum logit value.

While no formal quantitative analysis have been performed yet regarding the use of LogitLens, qualitative results tend to show that only the three last layers of RNBert reflect the final label, both in terms of predicted class and logit value. Indeed, these three layers are the fine-tuned layers, while the other layers of MusicBERT were frozen (Sailor, 2024). In contrast, our end-to-end model suggests that the prediction is more evenly distributed across all layers. Therefore, although the following analyses presented in Sections 7.3 and 7.4 could have been conducted on



(a) Average logit of the predicted class at each layer of our end-to-end model.

(b) Average logit of the predicted class at each layer of RNBert.

Figure 7.3: Application of logit lens on RNBert and our end-to-end model for the sub-task of local key labeling for a piece. The color scale indicates the logit value when the lens is placed at each layer. On average, the decision-making process is more distributed across the layers for the end-to-end model, while only the three fine-tuned layers of RNBert are impacted by the decision.

the last three layers of RNBert, we choose to focus on our end-to-end model and consider all the layers.

7.3 Analyzing attention behavior

The model presented in the previous section is based on Transformers (Vaswani et al., 2017) which rely on an attention mechanism (Bahdanau et al., 2015). This mechanism computes new representations of a sequence X of size n by repeatedly passing it along the layers of the deep architecture. As described in Section 5.3, the self-attention mechanism is defined by a scaled dot-product (Equation (5.1)):

$$\text{Attention}(Q, K, V) = \underbrace{\text{Softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)}_{\text{Attention matrix} \in \mathbb{R}^{n \times n}} \times V$$

with a query $Q = XW_Q$, a key $K = XW_K$ and a value $V = XW_V$ which are involved in defining an **attention matrix**. This square matrix represents the contribution of each element of the sequence as represented by the previous hidden state to compute the next hidden state. In other words, a line of this matrix can be interpreted as the attention given from a token to all the tokens of the sequence. A column can be interpreted as the attention received by a token from all the tokens of the sequence.

In practice, in contrast with NLP, music sequences are much longer than text sentences (Huang et al., 2019), resulting in attention matrices with lower values per token. Therefore, for our sequences of size $n = 512$, we reduce each 512×512 attention matrices into 64×64 matrices by applying a sum-pooling with a kernel of size 8. This process has a musical interpretation: instead of considering the attention received by a thirty-second note, we now consider the attention received by a quarter note time interval. We consider that such time scale is adequate for a task of harmonic analysis. In the following, we will consider attention matrices as pooled matrices derived from the original attention matrices.

When performing functional harmony analysis, one may attend at longer or shorter range to annotate a chord. For example, identifying the local key may need to look at further elements (*e.g.* key signature, accidentals on a potential leading tone, ...) while labeling the inversion of a chord might only involve the chord itself. Therefore, while multiple features can be obtained from the attention matrix, we focus specifically on the relation between functional harmony and the *attention span*.

7.3.1 Attention span

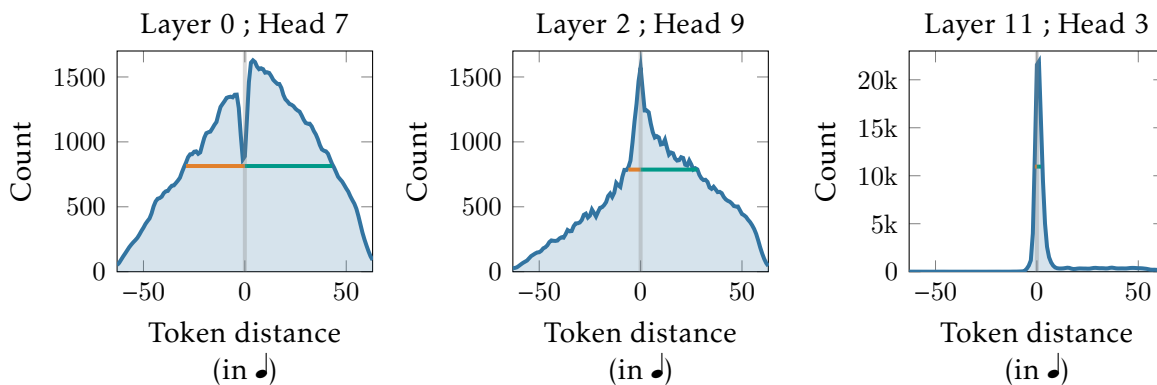


Figure 7.4: Attention spans grouped by length. They represent the distribution of relative distances between tokens for which the attention value is important. The line at mid-height represents the positive / negative width of the attention span.

We introduce a statistic designed to assess the attention span exhibited by the attention matrix of an attention head. Let $A(s) = (a_{ij}(s))$ be the attention matrix computed from a given attention head H for sequence s . Let θ be a given threshold. We will consider that token i pays attention to token j if a_{ij} is above θ . In which case, we will consider that the *attention span* of a token i is at least of $i - j$ tokens ahead or back in the sequence (depending on the sign).

$$\text{aspan}(a_{ij}) = \begin{cases} i - j & \text{if } a_{ij} \geq \theta \\ 0 & \text{otherwise} \end{cases}$$

At the level of the whole attention matrix computed by an attention head, we can define the **attention span of an attention head** as the distribution of $\text{aspan}(a_{ij}(s))$ values. More intuitively, this represents the distribution of signed distances between pairs of tokens where the attention value is important¹. Examples of such distribution are given in Figure 7.4. We can quantitatively characterize an attention span by considering the width of the distribution at mid height, a metric we refer to as **attention span width**. In particular, this width can be split into a negative (resp. positive) part, representing tokens attending to past (resp. future) tokens.

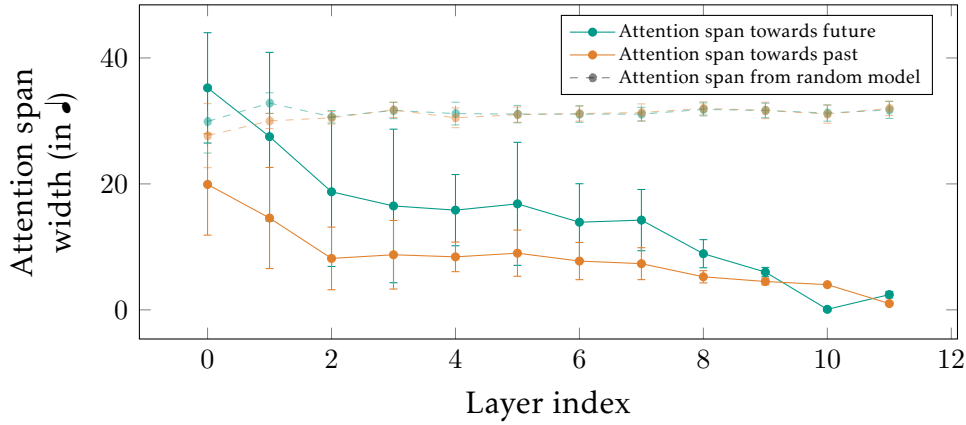


Figure 7.5: Average attention span width by layer. Attention spans represented by plain (resp. dashed) lines are extracted from the trained (resp. untrained) model. The attention towards future is more prominent than attention towards past.

Figure 7.5 shows the average attention span width by layer, with a distinction between future-oriented and past-oriented attentions (*i.e.* distinguishing positive and negative values of distances between tokens), for a trained and random model.

Attention span width – The average attention span width gets narrower across the layers of Transformer encoders. In comparison, an untrained model exhibits attention spans which remain constant across the layers. In other words, the hidden states from deeper layers of the model only need to attend to close hidden states in order to get information about much larger parts of the sequence. This shows that deeper layers manage to capture and model longer parts of the sequence within a few hidden states.

Attention towards past and future – We observe that the attention towards future is more prominent than attention towards past in the first layers. This phenomenon

¹In practice, we set $\theta = 0.8$, which is 10% of the maximum attention value on the full dataset. This value is a balance between a too strict and too weak selection of the attention values.

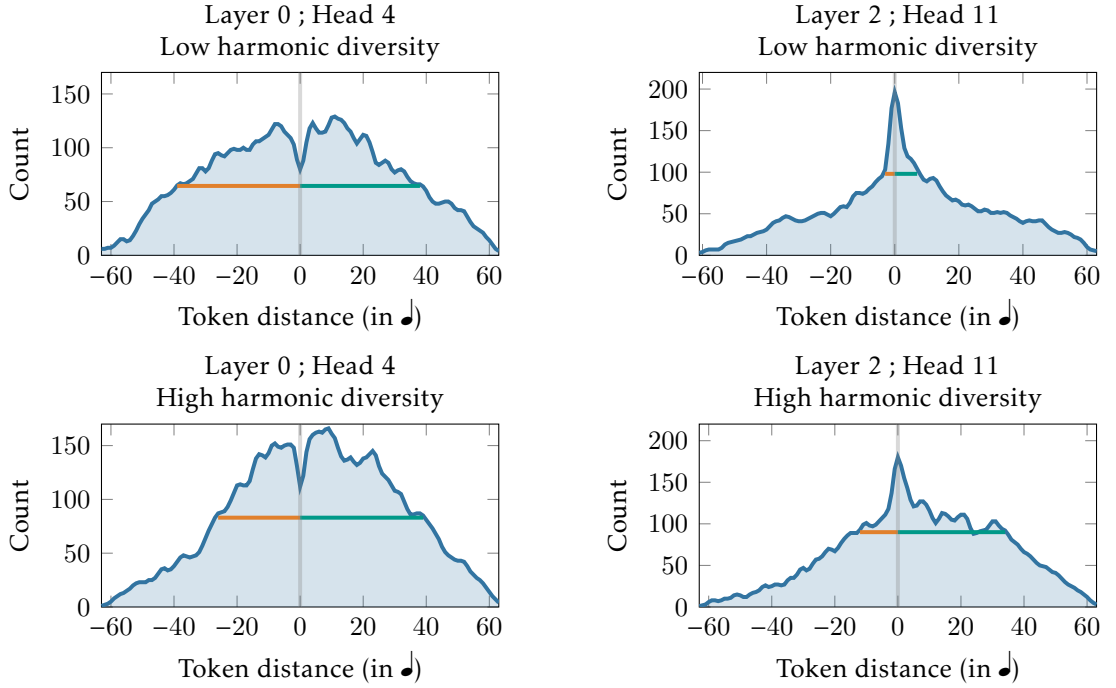


Figure 7.6: Attention spans grouped by length from selected heads impacted by a low (top) vs. high (bottom) harmonic diversity.

can be related to the manner functional harmony creates expectations in Western tonal music by building tension and release (Lerdahl and Jackendoff, 1996). Indeed, when composing music, a focus is often directed towards a final harmonic resolution in order to write a coherent harmonic discourse (Randles and Sullivan, 2013).

7.3.2 Attention span & harmonic diversity

Following the framework of an explainability study presented in Section 7.1, we consider *harmonic diversity* as a discriminative musical characteristic and we study the impact on the attention span. We define the harmonic diversity of a sequence $L = [\ell_1, \dots, \ell_T]$ the cardinality of the set of unique chord labels in this sequence:

$$\text{HarmDiv}(L) = |\{\ell_i | 1 \leq i \leq T\}|$$

Each sequence is thus characterized by a harmonic diversity value and we compare the average attention spans of the 50 most against the 50 least harmonically diverse sequences. We then compute the distribution of attention spans on these two types of sequences.

Figure 7.7 represents the average attention span width towards past and future, in the case of low and high harmonic diverse sequences. Sequences with higher harmonic diversity lead to larger attention span towards future and a shorter span

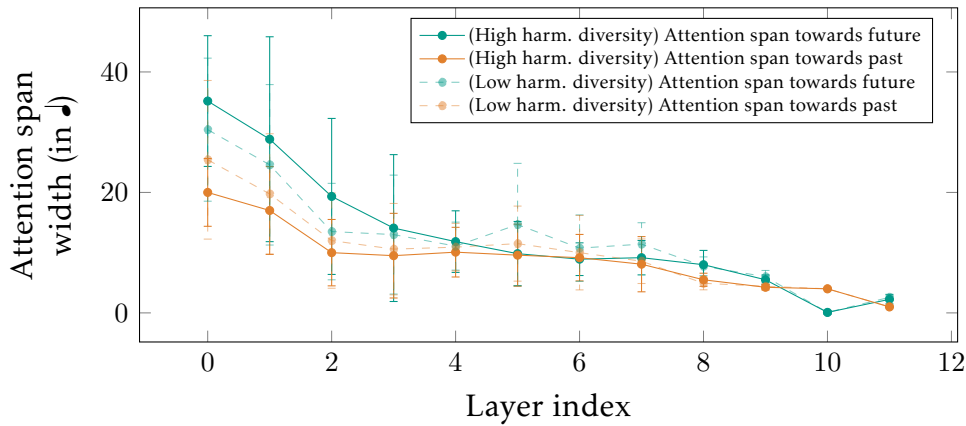


Figure 7.7: Average attention span width by layer, for low and high harmonic diversity. Higher harmonic diversity tends to shift the attention span towards the future. Attention spans from low (resp. high) harmonically diverse are represented in light / dashed (resp. plain) lines.

towards past than low harmonically diverse sequences. Examples of heads for which this phenomenon is pronounced are given in Figure 7.6. This shift towards future at the expense of the past can translate into a larger span towards future, or a reduction of the attention towards the past. Multiple musical characteristics may be considered to interpret this phenomenon. For instance, such high harmonic diversity may indicate a modulating transition (Schachter, 1987), leading to an attention mechanism more directed towards the arrival key of the modulation.

These observations regarding attention span are preliminary and lack of quantitative evaluation. In addition, considering raw attention values as an interpretable tool can however be sometimes debated (Jain and Wallace, 2019; Wiegrefe and Pinter, 2019) and may lack of faithfulness: it may not represent the internal workings and causal mechanisms of the model but instead provide plausible explanations for humans (Jacovi and Goldberg, 2020; Lyu et al., 2024). However, by introducing this new concept of attention span, we believe that further investigations could offer a deeper understanding of attention mechanism behaviors.

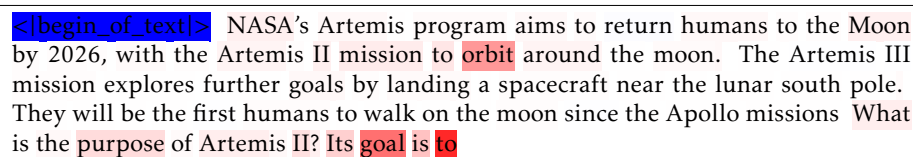
7.4 Understanding attention heads' relevance

Cognitive science studies have shown that specific regions of the brain are more “specialized” for particular tasks or stimuli, such as the auditory cortex in the temporal lobe for auditory stimulus processing or prefrontal cortex for decision-making tasks (Ward, 2019). As an analogy, functional harmony analysis is typically modeled as a multi-task framework. Therefore, when performed by an automatic analysis model, particular regions of the model – typically attention heads in the context of Transformer-based models – may possibly focus on different aspects of the harmony.

We explore the behavior of a method called *Layer-wise Relevance Propagation* (LRP) on our model. This technique is used to measure how much each artificial neuron contributes to the model’s final prediction, as explained in more detail in [Section 7.4.1](#). While this analysis can be later extended to a multi-task setting, we begin by analyzing the relevance of attention heads in a model trained specifically for local key detection ([Section 7.4.2](#)). We then attempt to explore whether similar tonal information can also be detected in a model that has only been pre-trained on a generic task, namely masked language modeling ([Section 7.4.3](#)).

7.4.1 Layer-wise relevance propagation

Layer-wise Relevance Propagation (LRP) ([Bach et al., 2015](#)) is a technique that traces the model’s prediction back through its layers to determine how much each input element contributed to the final output. This is done by iteratively distributing the output logit’s back to the input, layer by layer. In other words, LRP aims at highlighting the *contribution* of each a neuron of the inner layers – or of each input’s component – to the final prediction.



<|begin_of_text|> NASA’s Artemis program aims to return humans to the Moon by 2026, with the Artemis II mission to orbit around the moon. The Artemis III mission explores further goals by landing a spacecraft near the lunar south pole. They will be the first humans to walk on the moon since the Apollo missions. What is the purpose of Artemis II? Its goal is to

Figure 7.8: Application of LRP in NLP using AttnLRP ([Achtibat et al., 2024](#)). The model is Llama-3.2 and LRP is back-propagated from the last token <to>. Words highlighted in red (resp. blue) are the most (resp. least) relevant to predict the next token – which is the task on which the model has been trained.

This method is typically applied in image processing to identify which pixels of an image contribute most to its classification into a specific category ([Figure 7.11](#)). In NLP, LRP have been first used to highlight which words are relevant for a prediction from a convolutional neural network ([Arras et al., 2016](#)) as illustrated in [Figure 7.8](#). Later applied to Transformers, [Voita et al. \(2019\)](#) perform LRP to highlight not only the relevance of the input, but the relevance of the attention heads. This head attention relevance is defined as the sum of the relevances of its neurons. They observe that some attention heads are more relevant in this task, and some focus on specific types of words. [Achtibat et al. \(2024\)](#) then proposes AttnLRP, an extension of LRP for attention layers which ensures faithfulness. In the following, we use the implementation of AttnLRP ([Achtibat et al., 2024](#)) with the `lxt` library².

LRP is based on the idea that the value of the logit of the final prediction can be redistributed backward through the network to estimate the contribution of each neuron ([Figure 7.9](#)). The process involves two steps: a standard forward pass to

²<https://github.com/rachtibat/LRP-exPlains-Transformers>

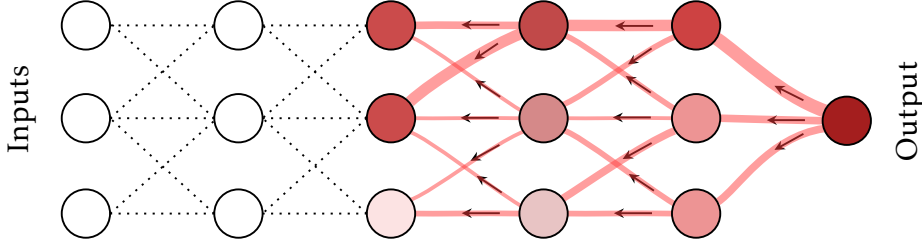


Figure 7.9: Qualitative intuition behind LRP. Given the logit of the final output, this value is back-propagated iteratively to the inner neurons to evaluate the extent to which each individual neuron contributed to the final prediction. Reproduced from (Montavon et al., 2019).

compute the output logit and the activations of each neurons of the network, followed by a backward pass (distinct from gradient back-propagation) that assigns relevance scores to individual neurons throughout the network.

More formally, it is based on propagation rules (Montavon et al., 2019). To begin with, let's first consider a simple feed-forward fully connected neural network with L layers with a single output. We focus on a sub-part of the network at layer $\ell < L$ as represented in Figure 7.10. In the following, we call a neuron the combination of the aggregation step and the activation function σ . We note the output of the i^{th} neuron of the layer ℓ computed during the forward pass as:

$$a_i^\ell = \sigma \left(\sum_j a_j^{\ell-1} \omega_{ji}^{(\ell-1, \ell)} \right)$$

where $\omega_{ji}^{(\ell-1, \ell)}$ is the weight between the neuron j of the layer $\ell - 1$ and the neuron i of the layer ℓ .

Let R_i^ℓ denote the **relevance** of the i^{th} neuron of the layer ℓ . It is computed in a backward pass once the forward pass is done. The LRP-0 rule (Montavon et al., 2019) defines the value of R_i^ℓ by recurrence following the expression:

$$R_i^\ell = \sum_k \frac{a_i^\ell \omega_{ik}^{(\ell, \ell+1)}}{\sum_j a_j^\ell \omega_{jk}^{(\ell, \ell+1)}} R_k^{\ell+1} \quad (7.1)$$

where the base case is³

$$R^L = a_1^L \quad (7.2)$$

In essence, a relevance at a particular layer can be interpreted as a weighted sum of the relevances of the next layer of the network.

³ a_1^L is typically the logit of the predicted class.

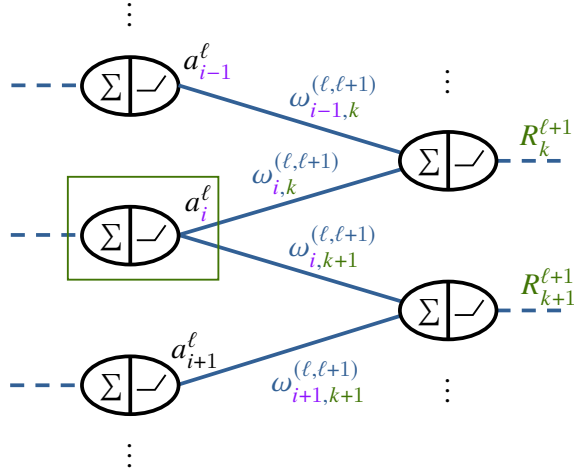
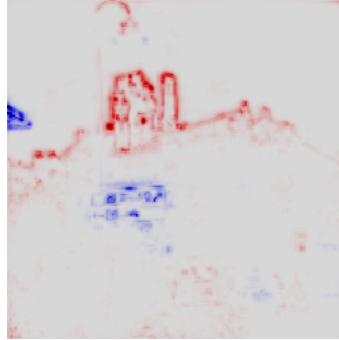


Figure 7.10: Elements involved in the LRP back-propagation. Equation (7.1) derives R_i^ℓ the relevance of the neuron in the green box. a_i^ℓ are the activations, $\omega_{i,k}^{(\ell,\ell+1)}$ the weights between neurons of layer ℓ and $\ell + 1$, and $R_k^{\ell+1}$ the already computed relevances when arriving on the layer ℓ .

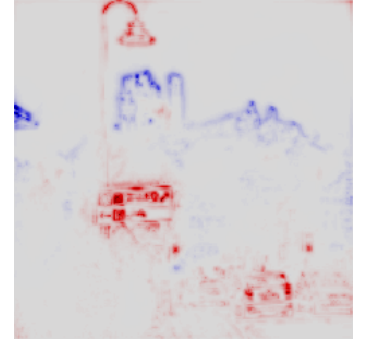
In particular, in the case of a multi-output model, note that R^L does not have to correspond necessarily to the logit of the predicted class. As illustrated in Figure 7.11, it can instead be the logit of any class, allowing us to back-propagate and analyze which components may have contributed to the decision on a given class, even if it was not the one predicted by the model.



(a) Original image.



(b) LRP back-propagation from the predicted label castle.



(c) LRP back-propagation from the 7th label street_sign.

Figure 7.11: Pixel relevances obtained via LRP back-propagation for different target labels. The classification model is a VGG-16 trained on ImageNet-1K. The label predicted by the model is castle. When ranked by probabilities, the model may have predicted church, monastery, ..., street_sign, We can back-propagate from the logit corresponding to any class which highlights the relevance of the pixel for this class.

Multiple enhancements such as LRP- ϵ and LRP- γ rules have been implemented to reduce noise or to improve the contrast between positive contributions over negative contributions. LRP can be reformulated using gradient values, and therefore, it

can be implemented leveraging automatic differentiation libraries ([Montavon et al., 2019](#))

In the case of case of Transformers, we want to quantify the relevance of a full attention head, more than single neurons. Therefore, we consider the relevance of an attention head to be the sum of the relevance of all the neurons of this head. Therefore, in the following, given a Transformer-based model with l layers and h heads per layer, we call **LRP map** the vector of size $l \times h$ ($= 12 \times 12 = 144$, for our considered model). It represents the relevance of each attention head for a given prediction. An example of LRP map is represented in [Figure 7.12](#).

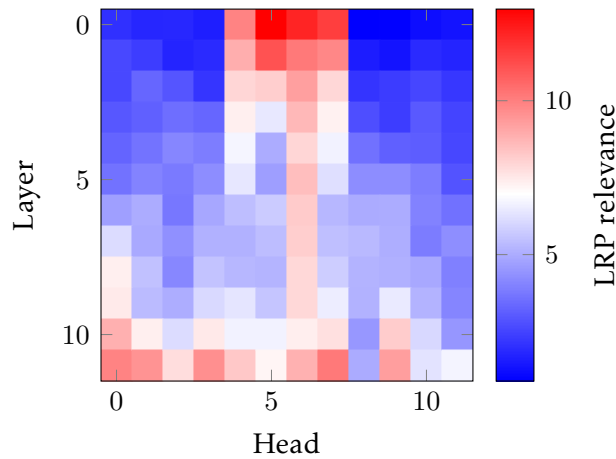


Figure 7.12: Example of an LRP map. Each cell represents the relevance of an attention head for a prediction. We represent as a $l \times h$ matrix as an intuitive visualization, but the employed metrics are standard vector distances.

7.4.2 Attention heads' relevance and musical tonality

When playing music, musicians may not have the same perception of a certain note if it is played inside sequences of different tonalities: the information conveyed by a certain note arguably varies depending on the tonality context in which it is played. For example, in the context of a choir, singing an F# may be less difficult when the surrounding key is B major because it represents the dominant of the scale rather than a key of C major, where it is the tritone of the root. This effect is even more pronounced when occurring within a piece with modulations, where a same pitch may be thought differently as it may not play the same role. If the inner mechanisms of the models are somewhat comparable to the cognitive process of the musician, one can wonder if different contexts (tonal, rhythmic, etc.), which are not explicitly encoded as such in the input data, can produce distinct inner states detectable through LRP maps.

An experimental protocol relying on LRP is developed to prob the inner mechanisms when the model is trained to distinguish tonal contexts. We show that the set

of relevant attention heads is more similar when they are built from tokens sharing the same tonality labels, compared to different tonalities. Going further, we also prob a model trained as a masked language model, in order to evaluate if functional harmony information is encoded in these inner mechanisms.

7.4.2.1 Hypotheses between LRP maps and local tonalities

In the following, we use these notations:

- Let X be a sequence of time-sliced tokens of length T .
- Let Φ_{ton} a Transformer-based model trained *only* on the task of tonality labeling composed of n layers with h heads per layer.
- Let $L = \ell_1, \dots, \ell_T$ a sequence of labels annotating the tokens X . These labels are typically the labels of the ground truth, but could also be those predicted by Φ_{ton} . Going further, we also explore cases where these labels relate to other features such as texture or other functional harmony-related features.
- Let $\text{LRP}(\Phi_{\text{ton}}, X, \ell_i) \in \mathbb{R}^{n \cdot h}$ the LRP map of Φ_{ton} for the prediction of ℓ_i from the sequence X . (*i.e.* the relevance of all attention heads of Φ_{ton} for the prediction of ℓ_i).
- Let $d(v_1, v_2)$ the euclidean distance between two vectors v_1 and v_2 with the same size.

We begin by considering *intra-sequence* hypotheses to assess whether the set of attention heads that activate for the same tonality is more similar compared to different tonalities within a same sequence. In particular, the segmentation of the sequence in terms of tonality labels can be performed at multiple levels of granularity, including token-wise and bar-wise.

Hypothesis 1a (token-wise) – We want to verify the following hypothesis:

In a sequence, given two *tokens* of the same label, and two *tokens* of different labels, we hypothesize that the activated heads for the pair of *tokens* with the same label are more similar than for those the pair of *tokens* with different labels.

More formally, let

$$\mathcal{R}_k = \text{LRP}(\Phi_{\text{ton}}, X, \ell_k)$$

the map of head relevances computed from ℓ_k through LRP. We want to verify the following hypothesis.

Given i, j, k such that $\ell_i = \ell_j$ and $\ell_i \neq \ell_k$, is the following relation true?

$$d(\mathcal{R}_i, \mathcal{R}_j) < d(\mathcal{R}_i, \mathcal{R}_k)$$

Hypothesis 1b (bar-wise) – We want to verify the following hypothesis:

In a sequence sliced into equal-length one-bar-long window, we hypothesize that given two *bars* of the same label, two *bars* of different labels, the activated heads for the *bars* with the same label are more similar than those for the *bars* with different labels.

More formally, a sequence of labels L can also be partitioned into N windows as $L = W_1, \dots, W_N$ such that $\forall i, j, |W_i| = |W_j| := \omega$. In other words, a sequence of labels can be split into windows containing the same number of tokens ω . In practice we take ω to be one-bar long because key changes often occur at the level of bars.

We introduce τ_k associated to a window W_k such that $\forall \ell, \ell' \in W_k, \ell = \ell' := \tau_k$. In other words, it represents the tonality of a bar in which all the labels are identical (*i.e.* bar which does not include musical modulations).

Let

$$\mathcal{R}_k = \frac{1}{\omega} \sum_{\ell_j \in W_k} \text{LRP}(\Phi_{\text{ton}}, X, \ell_j)$$

the average of maps of head relevances within a window. We want to verify the following hypothesis.

Given W_i, W_j, W_k such that $\tau_i = \tau_j$ and $\tau_i \neq \tau_k$, is the following relation true?

$$d(\mathcal{R}_i, \mathcal{R}_j) < d(\mathcal{R}_i, \mathcal{R}_k)$$

We then consider an *inter-sequence* hypothesis which extents the last hypothesis by considering multiple sequences.

Hypothesis 2b (bar-wise) – We want to verify the following hypothesis:

Given two *bars* of the same label, and two *bars* of different labels – which do not include label changes, but possibly coming from different pieces – we hypothesize that the activated heads for the pair of *bars* with the same label are more similar than those for the pair of *bars* with different labels.

Let B_a a bar extracted from the sequence $X_a = B_{a,1}, \dots, B_{a,N}$ having N bars. B_a of same label τ_a annotated by $B_a = \ell_1, \dots, \ell_M$ where M is the number of sub-beats in a

bar. In other words, $\forall i, \ell_i = \tau_a$ (e.g. with tonality labels, that means that B_a is in the key of τ_a). We consider

$$\mathcal{R}_a = \frac{1}{|B_a|} \sum_{\ell_j \in B_a} \text{LRP}(\Phi_{\text{ton}}, X_a, \ell_j)$$

We want to verify the following hypothesis.

Given two bars B_a and B_α both annotated with the same label $\tau_a = \tau_\alpha$, possibly extracted from two different sequences, and B_b annotated with the label $\tau_b \neq \tau_a$, is the following relation true?

$$d(\mathcal{R}_a, \mathcal{R}_\alpha) < d(\mathcal{R}_a, \mathcal{R}_b)$$

For all the hypotheses presented above, we aim to evaluate them across the entire dataset to capture overall behaviors. To do so, for each level considered by the hypotheses, we want to verify two elements:

- (i) The two distributions of pairwise distances between LRP maps for similar vs. different labels are distinct.
- (ii) If so, the average distance between LRP maps resulting from similar labels is lower than the average distance between LRP maps resulting from different labels.

For (i), we compute the distribution of pairwise distances when the labels are identical and the distribution of pairwise distances when the labels are different. We then compare these two distributions through a Jensen-Shannon Divergence (JSD) which quantifies the divergence between two distributions P and Q as a symmetrized version of the Kullback-Leibler divergence, noted as $\text{KL}(P \parallel Q)$:

$$\text{JSD}(P \parallel Q) = \frac{\text{KL}(P \parallel M) + \text{KL}(Q \parallel M)}{2}$$

with $M = \frac{1}{2}(P + Q)$

A high JSD would indicate that the distributions are well distinct. For comparison, we evaluate this value against a case in which the labels are randomly assigned. To ensure comparability with the true label sequences, the random sequences are constructed by preserving the same distribution of labels and section lengths, but assigning labels randomly.

For (ii), we report the average distance between LRP maps in the two cases: d_+ is the average distance between LRP maps when they are labeled identically and d_- is the average distance between LRP maps when they are labeled differently. A high distance suggests that different attention heads contribute to the labeling of the two

tokens (sections, sequences or bars), while a low distance indicates that similar heads are involved.

To ensure comparability across hypotheses, we also consider *normalized distances* $d_{=}^{\text{norm}}$ and d_{\neq}^{norm} , obtained by centering the raw distance values around the average over all the dataset and scaling them by the standard deviation.

7.4.2.2 Evaluating the hypotheses between LRP maps and local tonalities

	(i) JSD between same vs. different labels		(ii) Average distance between LRP maps		
	True labels	Random labels [†]	Same label ($d_{=}$)*	Different label (d_{\neq})*	$d_{\neq}^{\text{norm}} - d_{=}^{\text{norm}}$
1 Hypothesis 1a (token)					
2 <i>Local key</i>	0.156	0.027	10.231 -0.463	18.551 0.519	0.982
3 Hypothesis 1b (bar)					
4 <i>Local key</i>	0.132	0.009	11.589 -0.240	18.154 0.656	0.896
5 <i>Roman Numeral</i>	0.071	0.005	9.547 -0.518	14.393 0.143	0.661
6 <i>Quality</i>	0.029	0.003	11.247 -0.286	14.290 0.129	0.415
7 <i>Inversion</i>	0.007	0.002	12.681 -0.090	13.904 0.076	0.166
8 <i>Rhythmicity</i>	0.002	0.004	12.752 -0.081	13.685 0.046	0.127
9 Hypothesis 2b (bar)					
10 <i>Local key</i>	0.062	7.726e-5	19.148 -0.490	24.065 0.021	0.511
11 (MLM) Hypothesis 1b (intra-sequence, bar-wise)					
12 <i>Local key</i>	0.002	0.004	254.105 -0.020	258.002 0.057	0.077
13 (MLM) Hypothesis 2b (inter-sequence, bar-wise)					
14 <i>Local key</i>	0.006	3.034e-5	163.047 -0.087	170.511 0.004	0.091

Table 7.2: Jensen-Shannon Divergences and average distances between LRP maps computed from pairs of same and different labels. The lower the JSD values, the more similar the distributions. (*) The indicated sub-values are the normalized distances $d_{=}^{\text{norm}}$ and d_{\neq}^{norm} to ensure comparability between cases. (†) The random distribution follows the proportion and section lengths of the true distribution.

We evaluate the hypotheses presented in the previous section using the model presented in [Section 7.2.2](#) with only the local key classification head. Its performance is 79.5% accuracy on the task of local key detection ([Table 7.1](#)).

To begin with, we evaluate the *intra-sequence* hypotheses presented above in Section 7.4.2.1. The metrics presented above are presented in Table 7.2 (rows 1–10).

Hypothesis 1a (label-wise) – The hypothesis states that two *tokens* within a sequence that share the same label exhibit more similar relevant attention heads compared to *tokens* labeled with different labels. Here, we focus only on *tonality* labels.

In order to test the hypothesis, we randomly sample 100 sequences of length $T = 512$. This results in a dataset of 13M pairs of LRP maps⁴ from which we can compute pairwise distances. These pairs of LRP maps can be grouped into two categories: couples of tokens labeled with the same tonality, or different tonalities.

The distribution of these distances is presented in Figure 7.13 and tends to show that the distributions between same and different labels are distinct. This is notably confirmed by the JSD between the two distributions (Table 7.2, row 2) which is lower when the labels are well-assigned in comparison with random labeling, which confirms (i). Moreover, the average distances between LRP maps tend to validate (ii): attention heads relevant to the same label are more similar than those involved in labeling different classes.

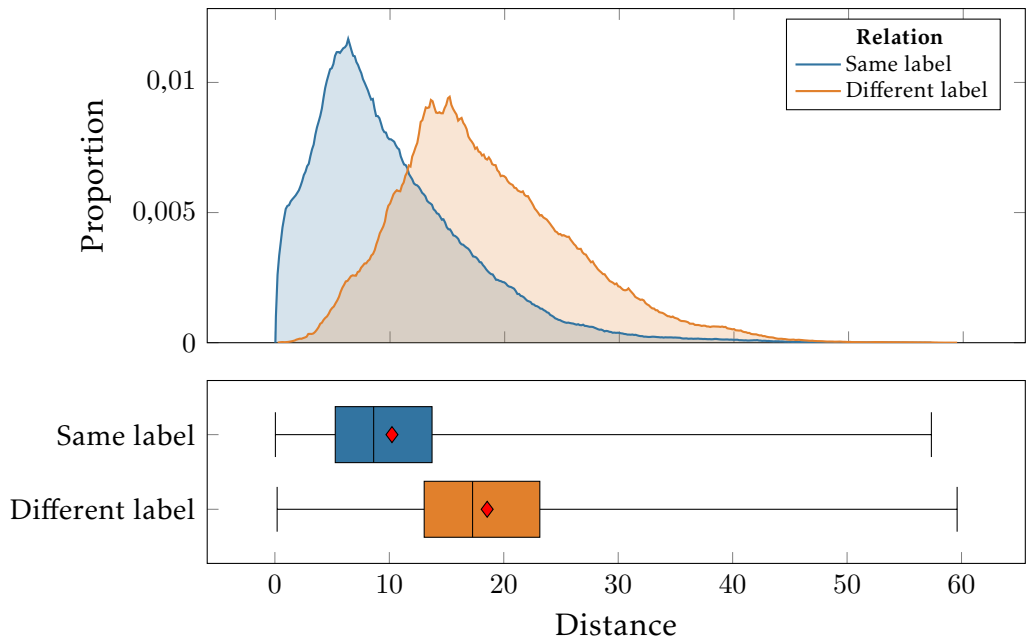


Figure 7.13: Hypothesis 1a (label-wise): distribution of distances between label-wise LRP maps grouped by same tonality vs. different tonality. The average of each distribution is indicated by red markers and corresponds to the values reported as $d_{=}$ and d_{\neq} in Table 7.2.

However, while this property holds on average, we observe in Figure 7.13 that some couples of tokens labeled with the same tonality exhibit LRP maps with higher distance than couples of tokens of different tonalities. We want to dive deeper into understanding this behavior by taking a particular piece as an example (Figure 7.14).

⁴100 sequences with $\binom{512}{2} = 130k$ pairwise distances for each.

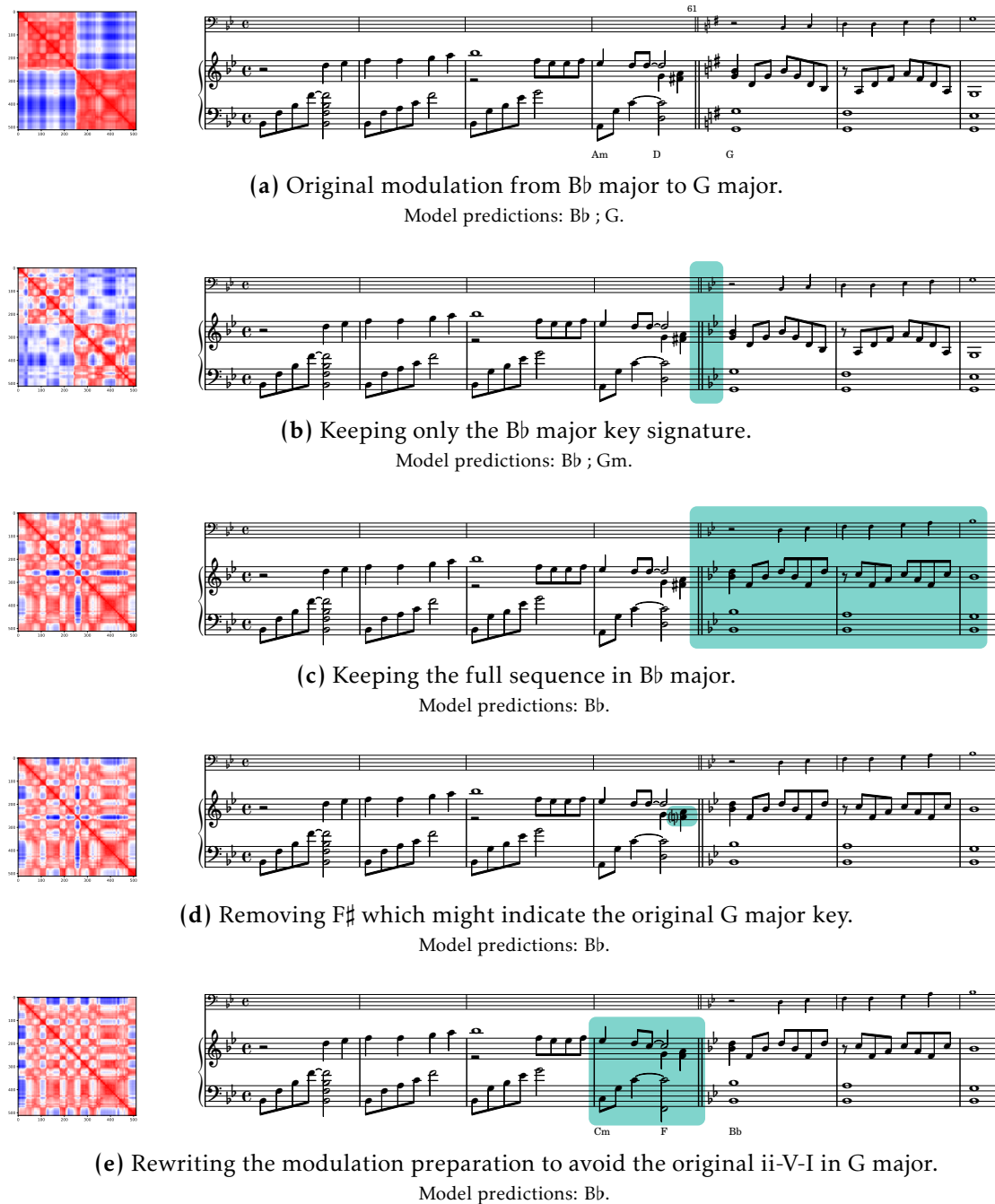


Figure 7.14: Impact of the musical content on the attention head relevances when approaching a modulation. LRP analysis is performed on multiple modified version of this extract, with the modifications highlighted in blue. In (c), transposing the modulated part makes the model only predict one key, but the relevances show that relevant heads still differ near the initial modulation. This phenomenon only disappear when re-writing the preparation as shown in (e). The left column represents the token-wise LRP maps with the modulation located at the middle of the sequence. (Piece: *For the beauty of the earth* (J. Rutter), bars 57–62)

We investigate this qualitatively by constructing a series of counterfactual modifications on a particular piece. The original piece includes a modulation from B♭ to G major (7.14a), and the distance matrix between LRP maps (7.14a, left⁵) shows that the set of attention heads are different when going through the modulation and the model do predicts this modulation. We want to observe what causes the model to predict these two keys.

- A first attempt is to change the key signature, by keeping the two flats of the B♭ key (7.14b) and switching all the B and E to B♭ and E♭. Naturally, this leads to a modulation into G minor instead (because they share the same key signature).
- Going further, we transpose the whole modulated part from G major to B♭ major (7.14c). The model no longer finds a modulation and only predicts a B♭ major tonality, as expected. However, at about mid-sequence, the set of attention heads still differs locally from the relevant heads for B♭ (blue lines). In other words, the internal representations of the model still detects a bit of modulation at this location, even though the final label remains a single tonality.
- As a possible explanation for this phenomenon, we turn the F♯ into a F♮ (7.14d). Indeed, as F♯ was the leading tone of the original new tonality, it might have prompted the model to trigger a modulation. Though, the phenomenon still remains (blue lines).
- Instead, re-writing the whole modulation preparation (which is a ii-V-I cadence preparation) makes the blue lines disappear (7.14e). Therefore, this specific case study seriously suggests that cadence preparations for modulations are captured by attention heads relevances.

Hypothesis 1b (bar-wise) – The hypothesis states that two *bars* within a sequence that share the same label exhibit more similar relevant attention heads compared to *bars* labeled with different labels.

In the same way as the previous hypothesis, as shown in Figure 7.15 and Table 7.2 (row 3–8), the metrics shows that the distributions between same and different tonalities are more distinct than a random labeling. Moreover, the relevant heads involved in the classification of bars with two different tonalities are more different that those classifying a same tonality. This is noticeable result as the aggregation of labels is performed through equal-size one-bar-long windows, independently from the position and the length of each contiguous sections of labels.

⁵The cell (i, j) of this matrix represents the distance between the LRP maps computed from the i^{th} and the j^{th} tokens. A blue color represents a low similarity between the LRP maps (*i.e.* high distance), and a red color represents a high similarity between the LRP maps (*i.e.* low distance).

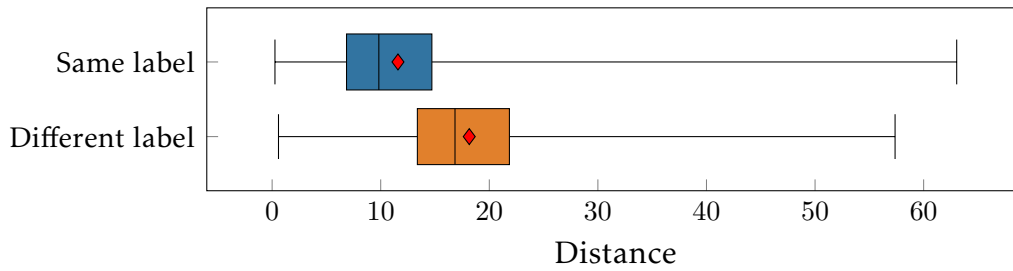


Figure 7.15: Hypothesis 1b (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality.

We then further explore whether our model trained *only* on local key prediction exhibits internal mechanisms that reflect other low-level features of harmonic analysis. In other words, we reproduce the same evaluation by comparing pairs of sections based not on their local key labels, but on properties such as chord quality, inversion, and other descriptors. In particular, we focus on the sub-tasks on “roman numeral” (which corresponds to the final roman numeral label for a chord), “chord quality” and “chord inversion”. We added a textural attribute, namely a “rhythmicity” feature which counts the number of onsets in a bar⁶ because musical texture and harmony are often considered as two orthogonal descriptions of music (Wang et al., 2020b). These comparisons are shown in Table 7.2 (rows 5–8).

Interestingly, for the “roman numeral” and “chord quality” sub-tasks (rows 5–6), their distributions are more distinct than a random distribution. In contrast, the “inversion” and “rhythmicity” (rows 7–8) show similar or worse divergences compared to a random distribution. From a musical viewpoint, this suggests that a model trained on key detection also includes information about lower-level harmonic features in its inner mechanisms. Going further, these sub-tasks can be ordered by presence in this mechanism: “inversion” and “rhythmicity” are the least present, and are also typically the musical features that may be the least relevant to determine a tonality.

In this hypothesis, we have not focused on the case of bars having a label change as defined in Section 7.4.2.1. Further study can dive into this case. For example, by grouping bars into types of modulation, it would be insightful to evaluate whether bars involving tonic-dominant or tonic-relative modulations exhibit distance values closer to the “same label” case, in contrast to bars with other types of modulations.

In summary, our experiments to test the *intra-sequence* hypotheses suggest that the set of attention heads that are relevant to classify labels in the same tonality is more similar compared to different tonalities *within a same sequence*. This phenomenon is observed on average across all three segmentation levels of the sequence: label-wise, section-wise, and bar-wise.

We then extend the evaluation of this hypothesis to the *inter-sequence* context. We

⁶The rhythmicity attribute is formally defined in Section 8.1.1.

follow the same framework as the one presented for the intra-sequence hypotheses. The same metrics defined above are reported in Table 7.2 (rows 9–10).

Hypothesis 2b (bar-wise) – The hypothesis states that two *bars* – possibly from two different sequences – that share the same label exhibit more similar relevant attention heads compared to *bars* labeled with different labels. Here, we also focus on the *tonality* labels.

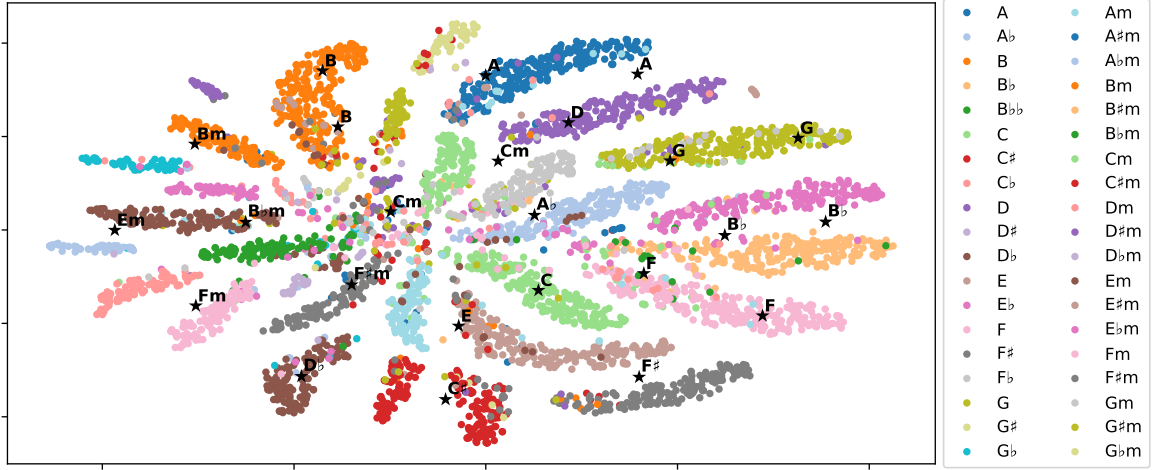


Figure 7.16: Hypothesis 2b (bar-wise): dimensionality reduction of bar-wise LRP maps through t-SNE. The sample represented in the figure is composed of 5k points.

For this hypothesis, we randomly sample a subset of 20k bars from all the test set due to computational limits. First, as an initial qualitative visualization (Figure 7.16), we perform a dimensionality reduction of sequence-level LRP maps, grouping them by tonality. Clusters are computed on this 2-dimension projection through a 24-cluster agglomerative clustering. Qualitatively, sequences with the same tonality seem to exhibit LRP maps that gather into clusters in this low-dimensions space. However, no musically meaningful high-level structure (e.g. Tonnetz, circle of fifths...) can be directly derived from this projection.

We then compute the pairwise distances between bar-wise LRP maps (Figure 7.17). The JSD between the distributions of distances between LRP maps annotated with the same vs. different labels confirms that these distributions remain distinguishable (Table 7.2, row 10). In terms of average distances, attention heads also tend to exhibit greater similarity when involved in the labeling of the same class, compared to when they are labeled with different ones.

However, when comparing the *intra-sequence* (row 4) and *inter-sequence* cases (row 10) with the same level of segmentation (i.e. at a bar-level), the difference $d_{\neq}^{\text{norm}} - d_{=}^{\text{norm}}$ is more important in the *intra-sequence* case. In other words, the relevance attention heads is influenced not only by the tonality but also by other differences between sequences. From a musical perspective, the intra-sequence framework implicitly assumes a degree of consistency in musical texture, meaning that tonality becomes

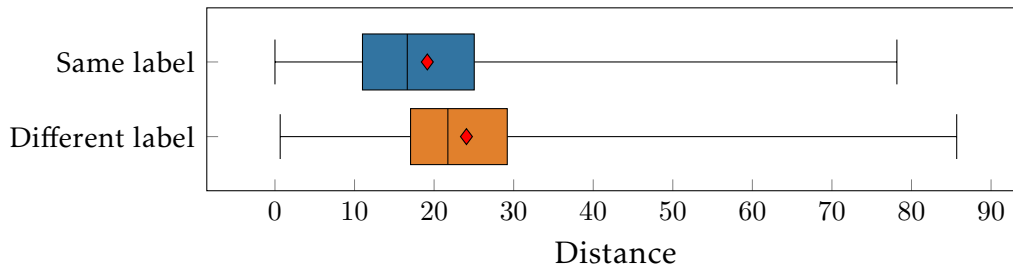


Figure 7.17: Hypothesis 2b (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality. These bars can be extract from *different* sequences.

the main varying factor. Consequently, variations in attention head relevance are primarily driven by changes in tonality. In contrast, across different sequences, variations in texture introduce additional differences. Although the model is only trained on local key detection, further study may explore whether it does capture additional musical attributes – such as textural differences between sequences – which, in turn, may mitigate the impact of tonality alone on attention head relevance.

In summary, attention head relevances remain more similar when processing same tonalities compared to different tonalities, even *across different sequences*. Though, in comparison with the intra-sequence framework, this shows that the model may capture additional feature, beyond the tonality.

7.4.3 Is tonality learnt during pre-training? Applying LRP to a MLM

In NLP, grammatical relations can be found by probing the attention mechanism of a pre-trained-only model such as BERT (Clark et al., 2019). As many studies often compare musical harmony to grammar, we attempt to probe a pre-trained model using the tools presented above based on LRP maps.

Pre-training for multi-hot time-slice representation – Following the architecture presented in Figure 7.2, we pre-trained this model as a Masked Language Model (MLM) following the pre-training of BERT (Devlin et al., 2019) models. MLM is a training objective where some tokens in a text sequence are replaced with a special mask token or randomly swapped with another token of the vocabulary, and the model is trained to predict the original tokens from the surrounding context. This allows the model to learn bidirectional contextual representations of language.

While the pre-training strategy of BERT relies on a cross entropy loss to predict the masked token from a pre-defined vocabulary, such an approach is not directly applicable to our multi-hot representation. To this end, we mask a whole time-slice (*i.e.* the three multi-hot vectors for the three components of the used representation

$(x^{\text{chroma}}, x^{\text{bass}}, x^{\text{onset}})$ which represent a slice). Therefore, the model's output are three multi-hot vectors $\hat{x}^{\text{chroma}} \in \mathbb{R}^{19}$, $\hat{x}^{\text{bass}} \in \mathbb{R}^{19}$ and $\hat{x}^{\text{onset}} \in \mathbb{R}^{14}$ which are passed through a $\text{Softmax}(\cdot)$ function so that each x is bounded in $[0, 1]$. The loss used for the reconstruction of the masked token is a sum of 52 binary cross-entropies (*i.e.* one for each element of the multi-hot vectors).

The implementation of the pre-training process is an adaptation of the pre-training implementation of MidiBERT-Piano (Chou et al., 2024). We adopt the same pre-training datasets as the original model which include Pop1K7 (Hsiao et al., 2021) and POP909 (Wang et al., 2020a), both consisting of piano covers of pop songs, Pianist8 (Chou et al., 2024), composed of piano performances by pop, contemporary and jazz pianists, and EMOPIA (Hung et al., 2021), composed of piano covers of Japanese and Korean anime and Western pop songs. To mitigate the predominance of pop-style music in the initial training data of MidiBERT-Piano, we additionally include ATEPP (Zhang et al., 2022), which features classical and romantic piano works. Given these considered datasets, we assume that our framework does stay within the realm of tonal music. The final pre-training dataset is composed of 14.5k pieces representing 10.4M tokens using our representation.

Performance of the pre-training – We evaluate the pre-training process quantitatively. To this end, we consider the reconstruction performance as being the component-wise accuracies between the ground truth multi-hot vector representing one component and its reconstruction by the model.

More precisely, for each component $c \in \{\text{chroma}, \text{bass}, \text{onset}\}$, we define \tilde{x}^c as being a binarized version of \hat{x}^c where:

$$\tilde{x}_i^c = \begin{cases} 1 & \text{if } \hat{x}_i^c > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Therefore, for each component c represent by a multi-hot vector of dimension N_c , we report the accuracy:

$$\text{Accuracy}^c = \frac{1}{N_c} \sum_{i=1}^{N_c} \mathbf{1}(x_i^c = \tilde{x}_i^c)$$

We compare a pre-trained model with a randomly-initialized model, as reported in Table 7.3. The pre-training proves beneficial, as the reconstruction accuracies are better for the pre-trained model compared to the random model across all components. However, the chromagram component shows surprisingly similar reconstruction accuracies between the two models. Therefore, assuming that this pre-training may have contributed to building potentially meaningful internal representations, we aim to investigate whether harmonic information can be detected within them.

Model	Chromagram (19 dim.)	Bass note (19 dim.)	Onset (14 dim.)	Average
Pre-trained MLM	0.62	0.80	0.66	0.70
Random model	0.61	0.56	0.39	0.53

Table 7.3: Performance of the masked language model pre-training. The values indicate the reconstruction accuracies between the ground truth multi-hot vectors and the model’s reconstruction of these vectors from a masked token.

LRP for multi-hot time-slice representation – The application of LRP is not direct because it must be back-propagated from a *single* logit value from the output. In our case, the output are three multi-hot vectors. To this end, instead of back-propagating LRP from each single value of these multi-hot vectors, we sum the values of the multi-hot vectors and back-propagate LRP from this aggregated sum. We show that these two processes are equivalent in [Appendix C](#).

Hypotheses & results – In the following, we evaluate hypotheses involving bar-wise segmentation for intra- and inter-sequence cases (*i.e.* 1b and 2b). More formally, we consider the same formulations of [Section 7.4.2.1](#) only by switching Φ_{ton} – which was only trained on the task of local key detection – into Φ_{MLM} , a pre-trained model which processes the same token representation, trained on a task of masked language modeling. While the other levels of segmentation would have also been judicious to be studied for an exhaustive analysis, we only focus on the bar-wise segmentation as our previous experiments with Φ_{ton} show differences between intra- and inter-sequence cases.

For the **hypothesis 1b (bar-wise)**, we present the distribution of distances between same vs. different labels in [Figure 7.18](#) and the quantitative metrics in [Table 7.2](#) (row 12). In contrast to the model specifically trained on local key detection ([Figure 7.15](#)), the distributions of LRP maps labeled with the same and different tonalities show only marginal differences compared to the random labeling scenario. This is also confirmed by the difference in average distances between the two distributions which also remains marginal (12 times less important when comparing the normalized distances – rows 4 and 12).

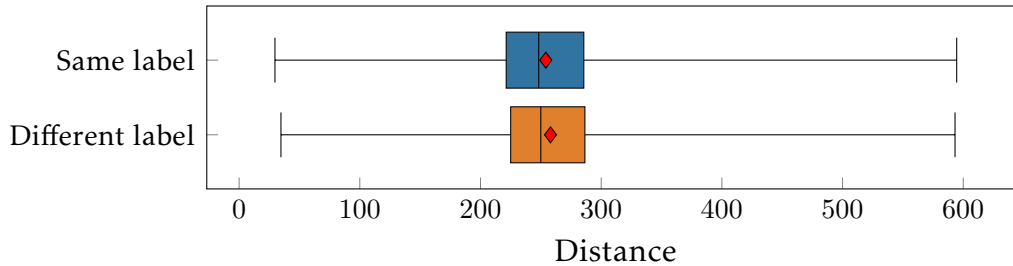


Figure 7.18: Hypothesis 1b MLM (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality.

For the **hypothesis 2b (bar-wise)**, we report the metrics in Table 7.2 (row 14) and we show the distribution of distances between LRP maps in Figure 7.20. We first perform a t-SNE on the LRP maps as presented in Figure 7.19. Similarly, in contrast with Figure 7.16, the dimensionality reduction does not show clear clusters in terms of tonality. However, points from the same tonality appear to not be spread at random in the dimensionality reduction. This is supported by the JSD value between the random sequence labeling vs. the true labels. Even though the raw value for the true labels is not as important as for the model trained on local key, there is still a notable difference with the random labeling. Similarly as the intra-sequence case, the difference between the distribution distances between of LRP maps annotated with the same label vs. different label is not as important than in the case involving the model trained on local key detection (row 10).

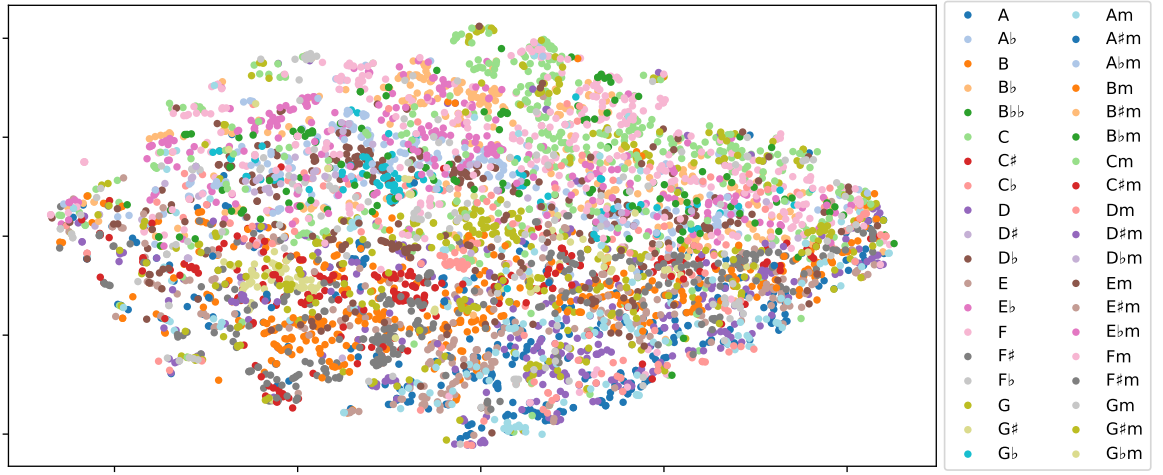


Figure 7.19: Hypothesis 2b MLM (bar-wise): dimensionality reduction of sequence-wise LRP maps through t-SNE. The sample represented in the figure is composed of 5k points.

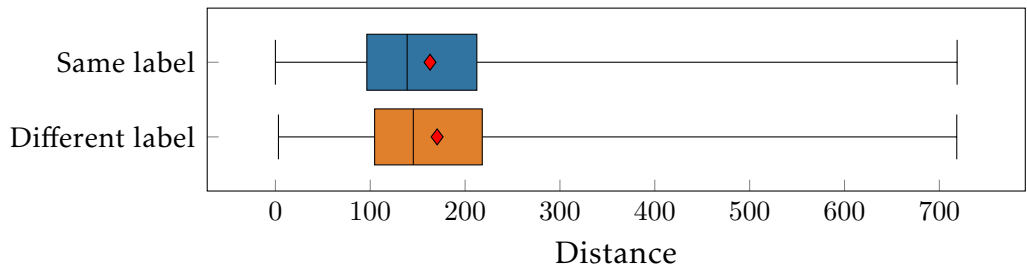


Figure 7.20: Hypothesis 2b MLM (bar-wise): distribution of distances between bar-wise LRP maps grouped by same tonality vs. different tonality. These bars can be extract from *different* sequences.

Discussion – Going further, these preliminary experiments on the presence or absence of tonality information within the inner mechanisms of a pre-trained-only model can raise questions about the relevance of considering harmony as a form of “musical grammar” inherently captured through a generic pre-training task. Using

attention head relevances as a probing method, our experiments do not give a clear answer regarding the ability for the model to learn harmonic concept through masked language modeling with our model and the considered pre-training dataset.

This result can be nuanced or questioned from two perspectives. First, we use local key as a probing feature, which might already be too high-level for the model to capture. In contrast, lower-level features – such as chord quality or scale degree – may be more likely to be internally represented. Moreover, we use the head relevances as a probing method. Alternative strategies, such as embedding analysis, may offer clearer insights into latent harmonic phenomena that the MLM could have implicitly encoded.

Chapter 8

Adapting NLP methods for a task of multi-track music generation

8.1	METEOR: melody-aware texture-controllable symbolic music re-orchestration	139
8.1.1	Textural attributes	139
8.1.2	Tokenization, model & control strategies	141
8.1.3	Inference guidance for melodic fidelity	143
8.1.4	Zero-shot lead sheet orchestration	145
8.2	Evaluating METEOR	145
8.2.1	Baseline models	146
8.2.2	Objective & subjective metrics	147
8.2.3	Results	150
8.2.3.1	Objective evaluation	150
8.2.3.2	Subjective evaluation	152
8.3	Towards realistic and humanly playable symbolic music generation	154

This chapter is based on a work published at the conference *IJCAI 2025* (International Joint Conference on Artificial Intelligence) for the Special Track on AI, Arts and Creativity (Le and Yang, 2025). This work was carried out during a 3-month international research stay at the AI & Music Lab of National Taiwan University in Spring 2024.

As described in Chapter 3, several types of musical tasks can be addressed with NLP tools. Although NLP and MIR share common tasks, certain musical tasks cannot be transposed to textual data, with *re-orchestration* being a notable example. In this chapter, we propose an adaptation of NLP tools to address this task in the context

of Western tonal music, framing it as a specific case of multi-track symbolic music generation.

Andante

Vln.1

Vln.2

Vla.

Vc. Cb.

C Dm G C Am D G

(a) Variation orchestrated with a string quartet, with a “sparse” texture (bar 1).

107

Fl.

Ob.

Fg.

Hrn.

Trp.

Timp.

Vln.1

Vln.2

Vla.

Vc. Cb.

C Dm G C Am D G

(b) Variation orchestrated with a full orchestra, with a “dense” texture (bar 107).

Figure 8.1: Example of re-orchestration in Haydn’s Symphony No. 94 (Mvnt. 2). Between these two extracts, the melody highlighted in yellow is broadly kept as well as the harmony (red labels) but the *musical texture* and the *instrumentation* are altered.

Re-orchestration – This task refers to the musical arrangement of an existing music piece for a different set of instruments (Cacavas, 1975). In the context of popular music, this notion is often associated with “song covers”. A key similarity between the original piece and a successful re-orchestration often lies in maintaining melodic fidelity. In Western music, music is often written under a homophonic texture (Young and Roens, 2022, p. 47). This musical texture is defined as a primary melody is supported by an accompanying background. In the composition process, effective orchestration requires knowledge of writing for various instruments by combining their timbres, while being restricted by their physical limitations (Adler and Hesterman, 1989).

Going further, re-orchestration extends beyond simply reassigning parts of the original piece to instruments in a new ensemble. It often involves altering the

overall *musical texture* of the piece to suit artistic goals or ensemble constraints. Musical texture characterizes how musical streams are organized and describes their content (Huron, 1989). An orchestral score can be described by global characteristics, such as instrument groupings or part diversity, and part-specific attributes such as rhythmicity or repetitiveness (Le et al., 2022). Re-orchestrations commonly appear in works written in the form of *variations*, as illustrated in Figure 8.1.

Model	Multi-track	Texture controllability		Melodic fidelity	Instrument choice		Open-source ¹
		Track-level	Bar-level		Full ensemble	Melody	
MuseMorphose (Wu and Yang, 2023b)	✗	✗	✓	✗	✗	✗	✓
MuseBarControl (Shu et al., 2024)	✗	✗	✓	✗	✗	✗	✗
FIGARO (von Rütte et al., 2023)	✓	✓	✓	✗	✗	✗	✓
PopMAG (Ren et al., 2020)	✓	✗	✗	(ind. track)*	fixed (6) [†]	✗	✗
GetMUSIC (Lv et al., 2023)	✓	✗	✗	(ind. track)*	fixed (5) [†]	✗	✗
BandControlNet (Luo et al., 2024)	✓	✓	✓	(ind. track)*	fixed (6) [†]	✗	✗
AccoMontage-band (Zhao et al., 2024b)	✓	✗	implicit**	(ind. track)*	implicit**	✗	✓
METEOR (presented in this chapter)	✓	✓	✓	✓	✓	✓	✓

Table 8.1: Models related to the style transfer sub-tasks performed by METEOR, our model presented in this chapter. ⁽¹⁾ We consider models to be open-source when both the code and trained models are publicly available. (*) The melody is added *a posteriori* as an independent track, in contrast with METEOR where the melodic instrument is chosen *among* the chosen instrumentation. (**) This model mimics the texture and the instrumentation of an *already existing* source: the choices are not explicit. (†) These models only handle a *fixed* number of instrument types (e.g. 5 or 6).

Re-orchestration as a generative task – In the field of symbolic music generation, re-orchestration can be considered as a *style transfer* task (Section 3.2.2), for which a model is designed to replicate a reference piece while altering high-level musical attributes. In particular, re-orchestration can be depicted under two music style transfer sub-tasks: *instrumental* style transfer, where the instrumentation of the reference piece is altered and *texture-based* style transfer, where high-level musical features from the reference are adjusted to generate a new piece. Existing style transfer systems among those presented in Chapter 5 may be inadequate to specifically perform a re-orchestration task (Table 8.1). They often focus on band arrangements (Zhao et al., 2024b; Luo et al., 2024) which restricts the instrument choices to a fixed and small ensemble and does not allow fine-grained selection of instrumentation. Beyond the instrumentation choice, re-orchestration implies textural controls, for which style transfer systems have also been implemented. This control is often performed at a piece-level (Lu et al., 2023) or bar-level (Wu and Yang, 2023b). For orchestral music – more generally, multi-track music – such control can also occur at the track level.

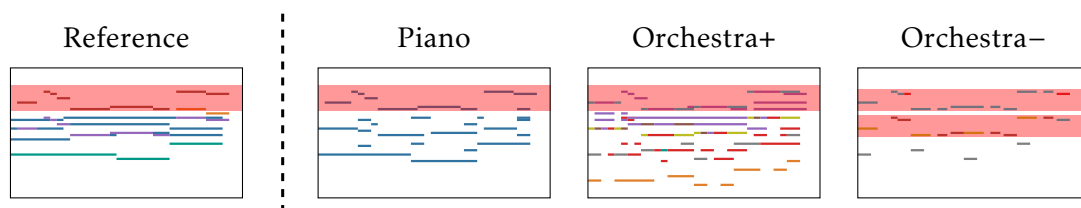


Figure 8.2: METEOR’s re-orchestration task. The model can re-orchestrate a reference for multiple instrumentations (e.g. solo piano, or orchestra) with texture controls, with more (orchestra+) or less (orchestra-) “polyphonicity” and “rhythmic intensity” (Section 8.1.1). The models ensures melodic fidelity (red highlight) with fine-grained controls (melodic instrument choice and pitch range).

In this chapter, we specifically perform the task of re-orchestration within the context of *homophonic* music, which consists of a melody supported by an accompaniment. Though, existing style transfer systems often overlook or even disregard the melodic fidelity of the generated content. For example, in the analysis of the style transfer model FIGARO (von Rütte et al., 2023), it is stated that “some salient features such as melodies are often not preserved”. Other models only generate the accompaniment and insert the melodic content *a posteriori* into a track played by a *fixed* instrument such as a synthesizer (Luo et al., 2024) or a “lead” track (Zhao et al., 2024b), without strict physical restrictions like its ambitus or register.

In this part, we present METEOR, a model for **M**elody-aware **T**exture-controllable re-**O**rchestration (Section 8.1). The model is based on MuseMorphose (Wu and Yang, 2023b) and is designed to achieve the following (Figure 8.2):

- *Multi-track music re-orchestration*: the model automatically orchestrates a reference multi-track piece, with the instrumentation possibly specified by the user.
- *Texture-controllability*: textural attributes can be controlled at both bar and track levels.
- *Melodic fidelity*: the melody is preserved in the re-orchestrated piece, with the option for the user to select the melodic instrument.

Beyond the task of re-orchestration with instrumental and texture-based style transfer with melodic fidelity, we show that our model can perform a lead sheet orchestration task without further training in a zero-shot manner (Section 8.1.4). We perform an objective evaluation which demonstrates METEOR’s effectiveness in bar- and track-level controllability, melodic fidelity, and melodic instrument playability (Section 8.2). A subjective evaluation further supports that it generates higher-quality re-orchestrations than baseline models. For this study, a demo website is available at: <https://dinhviettoanle.github.io/meteor/>.

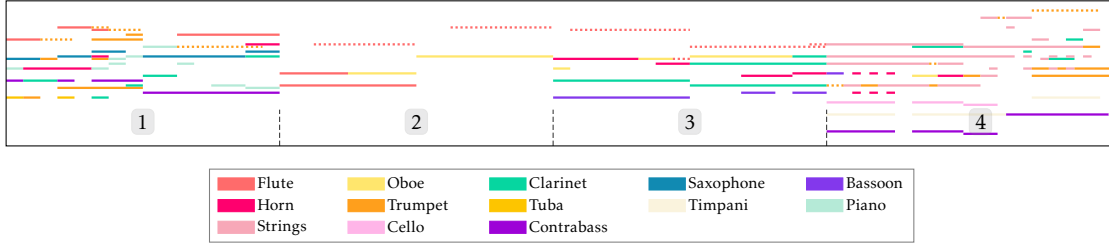


Figure 8.3: Pianoroll of a 8-bar re-orchestration generation by METEOR with various textural and instrumentation constraints changing each 2 bars resulting in 4 segments with different re-orchestrations throughout the same reference piece. The melody is represented as dashed lines. The orchestration constraints on the 4 segments are set as follows: (1) Automatic instrumentation, no textural changes. (2) Flute + oboe duet, melodic instrument: flute, texture: low polyphonicity. (3) Wind quintet, melodic instrument: flute, texture: low rhythmicity. (4) Classical orchestra, melodic instrument: trumpet, texture: high rhythmicity and polyphonicity.

8.1 METEOR: melody-aware texture-controllable symbolic music re-orchestration

In this section, we introduce METEOR, a Transformer-based VAE for multi-track re-orchestration with instrumentation controllability, bar- and track-wise texture controllability and melodic fidelity. We first present the musical attributes considered for textural controllability, followed by the technical contributions, particularly the tokenization strategies developed for the controllability aspect of the task and the melodic fidelity.

8.1.1 Textural attributes

METEOR is a model designed for both instrumental and textural style transfer (Figure 8.3). Specifically, its textural style transfer function enables the control of various textural attributes. We consider two levels of controllability: “bar-wise” (*i.e.* all tracks may be influenced by the control attribute) and “bar- and track-wise” (*i.e.* each track can be individually controlled at a bar level). We first consider bar-wise control attributes following the ones introduced in MuseMorphose (Wu and Yang, 2023b).

- *Rhythmic intensity* (or *rhythmicity*): number of sub-beats having at least one note hit within a bar containing B sub-beats, regardless of the track. With $\mathbf{1}(\cdot)$ the indicator function,

$$s^{\text{rhy}} = \frac{1}{B} \sum_{b=1}^B \mathbf{1}(n_{\text{onset},b} \geq 1)$$

- *Polyphonicity*: average number of notes played (hit or held) during a sub-beat in a bar containing B sub-beats, including all tracks. We consider

$$s^{\text{poly}} = \frac{1}{B} \sum_{b=1}^B (n_{\text{onset},b} + n_{\text{hold},b})$$

Each bar is characterized by a raw value of polyphonicity and rhythmicity. These raw values are then converted into categories by splitting the distribution of polyphonicity and rhythmicity values in the dataset into 8 bins, with a similar number of bars in each bin. We illustrate low and high values of these bar-wise textural features in Figure 8.4.

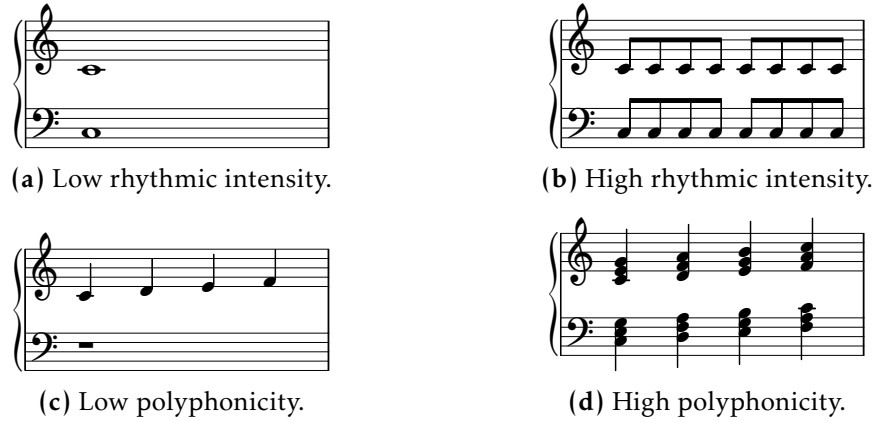


Figure 8.4: Examples of bars with different levels of bar-wise textural features: rhythmic intensity and polyphonicity.

For finer-grained control, we propose “bar-wise and track-wise” control attributes aiming at controlling each instrument individually among those initially selected at a bar level.

- *Average pitch*: average pitch of the set of pitches $\{p_1, \dots, p_M\}$ played in a track t in a bar, expressed in MIDI value and rounded to the nearest ten.

$$p_t^{\text{avg}} = \text{round} \left(\frac{1}{M} \sum_{i=1}^M p_i, 10 \right)$$

Levels of average pitches are thus divided into 13 classes, spanning from 10 to 130. For example, this attribute can be used to assign higher registers to melodic instruments and lower registers to bass parts, or, more rarely, to have all instruments play in their upper range to create a locally “brighter” sound, which is a common desired orchestral effect.

- *Pitch diversity*: number of different pitch classes played in a track in a bar.

$$p_t^{\text{diversity}} = |\{p_i \bmod 12 \mid i = 1, 2, \dots, M\}|$$

Levels of pitch diversity are divided into 13 classes, spanning from 0 to 12. Low pitch diversity can relate to bass parts, repeated notes or arpeggios, while high pitch diversity can encourage passing notes, embellishments or extended chords.

We illustrate low and high values of these bar-wise and track-wise textural features in Figure 8.5.

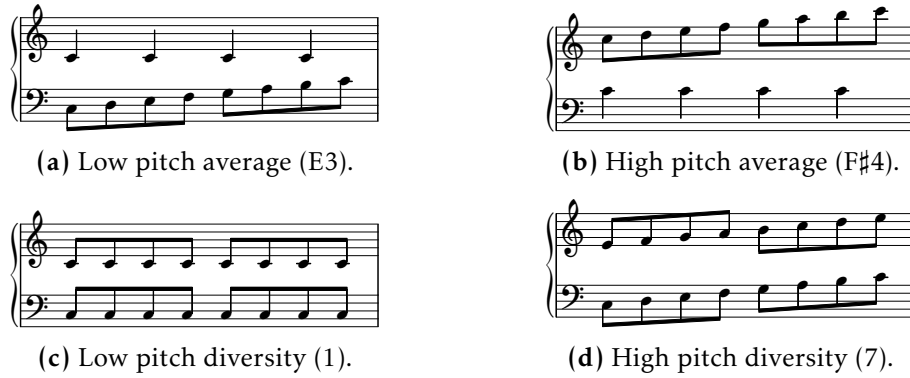


Figure 8.5: Examples of bars with different levels of bar-wise and track-wise textural features: average pitch and pitch diversity.

8.1.2 Tokenization, model & control strategies

Model architecture – METEOR’s architecture is based on MuseMorphose (Wu and Yang, 2023b), originally developed for piano style transfer. The model implements a Transformer-based Variational Auto-Encoder (VAE). As introduced in Chapter 5, a VAE is an architecture which generates new data (e.g. text, image or symbolic music) by sampling from a learned probabilistic latent space. More precisely, it consists of an encoder (Figure 8.6, left), with parameters ϕ , that maps input data x to a distribution $q_\phi(z | x)$ over latent variables z . From this latent space, a decoder (Figure 8.6, right), with parameters θ , then reconstructs the data by sampling from this latent space and generating x through the conditional distribution $p_\theta(x | z)$.

The model is trained by optimizing a loss function composed of two components: a reconstruction loss, which measures how well the decoder reconstruct the original input from latent variables z sampled from the approximate posterior $q_\phi(z | x)$ and a regularization term that ensures that the approximate posterior $q_\phi(z | x)$ is close to a prior distribution, typically a standard normal distribution $p_\theta(z) = \mathcal{N}(0, 1)$.

In the case of MuseMorphose, the model is trained by a β -VAE objective, where the regularization term is weighted by a β hyper-parameter:

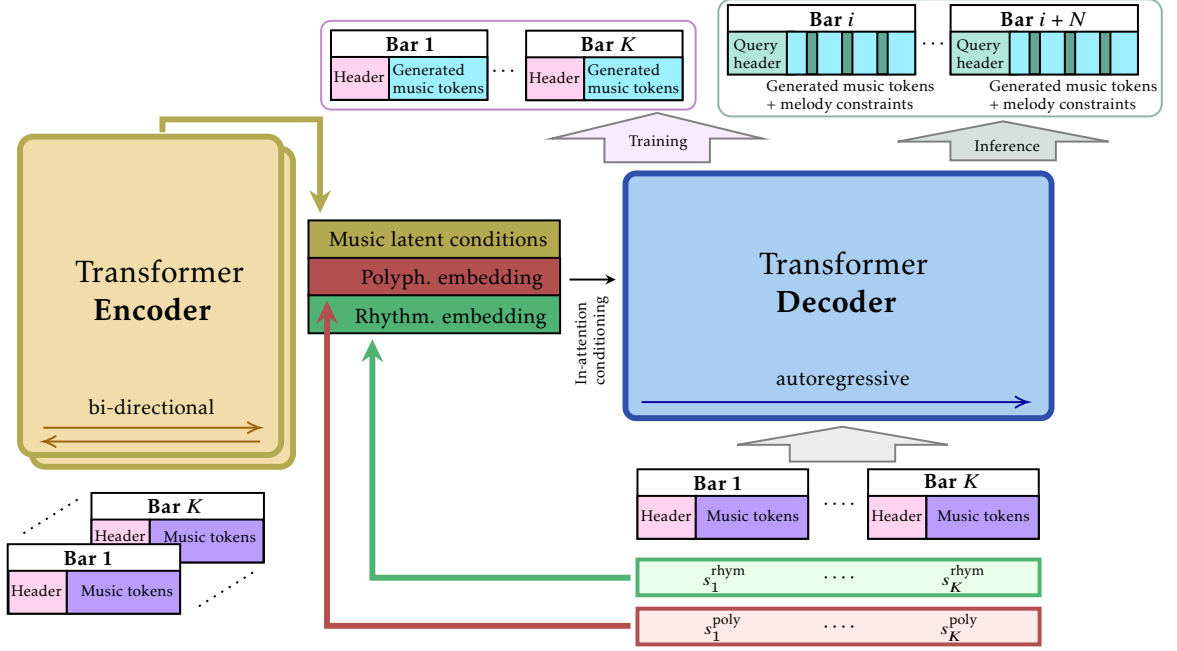


Figure 8.6: Architecture of METEOR, based on MuseMorphose. The musical content in each bar is preceded by a *header* describing the playing instruments in this bar and track-wise controls. During training, the model is trained to reconstruct K bars. At inference time, the user can specify different headers for each bar and starts the generation of N bars starting from bar $i < K$ (*i.e.* the user can ask to generate only from a sub-part of the full piece). The inference is guided with melody constraints at a beat level.

$$\mathcal{L}(x, \theta, \phi) = \underbrace{\mathbb{E}_{z \sim q_{\phi}(z|x)} [\log p_{\theta}(x | z)]}_{\text{Reconstruction loss}} - \beta \times \underbrace{\text{KL}\left(q_{\phi}(z | x) \parallel p_{\theta}(z)\right)}_{\text{Regularization term}}$$

Tokenization – We first extend its initial REMI tokenization (Huang and Yang, 2020) using the REMI+ tokenization (von Rütte et al., 2023) which handles multi-track music. Based on early experiments, we implement REMI+ using a vertical parsing (Figure 4.5), where notes are grouped and ordered based on time rather than track. We also implement a “pitch class + octave encoding” for the encoding of pitch information in the alphabet (Section 4.1.2). instead of absolute MIDI values, in particular, to handle melodies independently of the original octave register.

For instrumentation controllability, the user can select the playing instruments from a subset of 64 instruments defined in (Dong et al., 2023) or the ensemble can be automatically defined by the model. Instrument selection is handled through `<DescriptionTrack-[track]>` tokens that indicate instruments playing in a bar which are added in a *header* at the start of each bar in the token sequence.

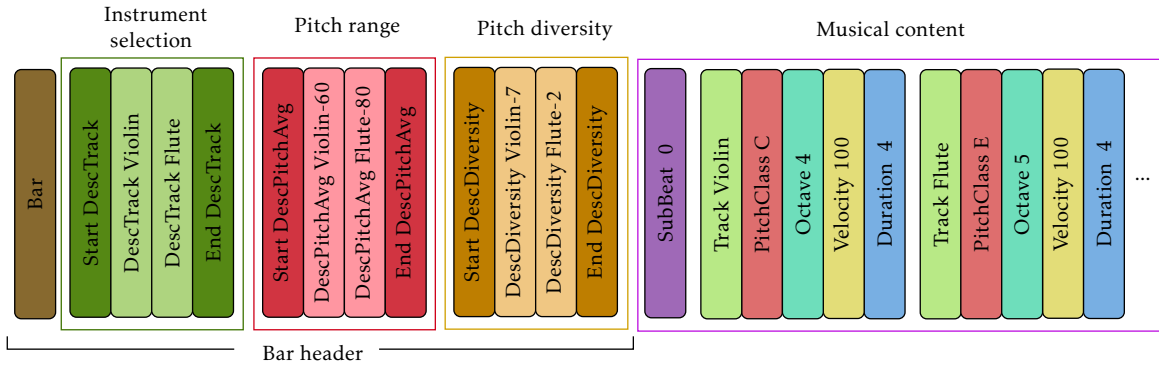


Figure 8.7: Example of a token sequence for a bar with a violin and a flute. As indicated in the *bar header*, the violin plays in a medium register and has a high pitch diversity, while the flute is in the upper range, with a low pitch diversity.

Control strategies – For texture controllability, the model implements multiple controls over various textural attributes (Section 8.1.1). For bar-wise and track-wise controls, `<PitchAvg-[track]-[level]>` tokens and `<PitchDiversity-[track]-[level]>` tokens are added jointly in this header to describe the average pitch and pitch diversity level of each track. An example of token sequence is shown in Figure 8.7. The resulting vocabulary is presented in Table 8.2 and no additional subword tokenization has been applied.

Following MuseMorphose, the bar-wise polyphonicity and rhythmicity classes are encoded in a separate sequence of bar-level conditions, which is embedded and concatenated with the latent vector and used as condition in the decoder through an “in-attention” mechanism (Wu and Yang, 2023b). This mechanism replicates the latent information across each timestep and layer of the decoder.

Training process – The model is trained as an end-to-end model on the SymphonyNet dataset composed of 46k multi-track pieces compiled from multiple websites (Liu et al., 2022)¹. The resulting model is 67M parameter-large and is trained for one week, set as an arbitrary time limit, on a single RTX 6000 24GB GPU.

8.1.3 Inference guidance for melodic fidelity

Melodies are crucial elements in music, as they often make a piece easily recognizable (Stefani, 1987). Thus, a key focus of our model is melodic fidelity, ensuring that the original melody is preserved in the generated extract, with possibly different textures in the accompaniment parts. Models preserving the melody often insert *a posteriori* a track containing the melody played by a generic instrument (e.g. synthesizer), which notably prevents any melodic ornamentation (Le et al., 2022). More

¹The dataset is not accompanied with musical metadata. Though, the exploration of this dataset through our experiments tends to show that it is not restricted to “symphonies” but extends to multi-track arrangements of pop songs, film music, etc.

Musical attributes	Vocab. size		
Bar	1		
Sub-beat	16	Control attributes	Vocab. size
Pitch class	12	Description track	64 (+2)
Octave	13	Pitch range	896 (+2)
Velocity	44	Pitch diversity	768 (+2)
Duration	16	Full vocabulary size	2078
Chord	113		
Track	64		
Tempo	65		

Table 8.2: Token vocabulary of METEOR. Pitch range and diversity tokens exist for each 64 tracks and for each 13 levels. Each control attribute has a <Start> and <End> token.

importantly, assigning the melody to a generic instrument restricts its integration within the queried ensemble, preventing it from being performed with the characteristics of a selected instrument. Thus, we propose an *inference guidance* process designed to ensure the melodic fidelity in a more flexible way in the generation.

Firstly, the melody is identified in the original piece during a pre-processing step using a bar-wise and track-wise skyline algorithm. The melody in each bar is estimated as being the track with the highest average pitch within that bar². Formally, the melody in a bar is represented as a list S^{mel} of the melodic tokens at each sub-beat of the bar. For example,

$$S_2^{\text{mel}} = [\text{<PitchClass_C>, <Octave_3>, <Duration_4>}]$$

means that the melody note on the second sub-beat of the bar is a C3 with a quarter-note duration.

Then, the instrument playing the melody is chosen beforehand by the model or can be specified by the user. In particular, the model or the user may choose to use different instruments to play the melody in different bars of the generated piece. The melody notes are then generated alongside with the re-orchestration: tokens identified as melody in the original piece are treated as beat-level conditions at inference time. This process is described in [Algorithm 1](#), omitting the header construction, edge cases, and implementation details.

Following [Figure 8.7](#), after a <Bar> token and the enforced *header* describing this bar, each <Sub-beat> token generated by the model is followed by an enforced <Track> token corresponding to the chosen melodic instrument, along with the tokens corresponding to the melody note played at this time position ([Algorithm 1](#), line 6). The next tokens (*i.e.* all the accompaniment tokens until the next melodic tokens)

²This assumption is a compromise as the melody can possibly be misdetected (*e.g.* melodic bassoon or cello). Further improvements may be implemented with track role identification ([Guo et al., 2019](#)).

are then generated auto-regressively (Algorithm 1, line 9). In particular, we do not restrict the model to generate additional notes played by the melodic instrument.

Algorithm 1 Inference guidance for a bar n containing B sub-beats

Parameter: S^{mel} , a dictionary containing melodic tokens in the bar n .

Parameter (optional): i , the melodic instrument at bar n .

```

1:  $X_{\text{gen}} \leftarrow []$                                 ▶ generated sequence of tokens
2:  $b \leftarrow 0$                                        ▶ current sub-beat position
3: while  $b < B$  do
4:    $t \leftarrow \text{METEOR}(X_{\text{gen}})$                  ▶  $t$  is a single generated token
5:   if  $t$  is  $\langle \text{Sub-beat}_k \rangle$  then
6:      $X_{\text{gen}} \leftarrow \text{CONCATENATE}(X_{\text{gen}}, [\langle \text{Sub-beat}_k \rangle, \langle \text{Track}_i \rangle], S_k^{\text{mel}})$ 
7:      $b \leftarrow k$ 
8:   else
9:      $X_{\text{gen}} \leftarrow \text{CONCATENATE}(X_{\text{gen}}, [t])$ 
10:  end if
11: end while

```

In further experiments described in Section 8.2.3.1, the melodic enforcement is partially released by allowing the model to infer $\langle \text{Octave} \rangle$ tokens to evaluate its relation with the instruments' register. Going further, one advantage of keeping the original melody is that our re-orchestration system can be used for lead sheet orchestration.

8.1.4 Zero-shot lead sheet orchestration

While METEOR has been specifically trained for a re-orchestration task, it can be adapted into a lead sheet orchestration model without requiring further training, effectively performing as a zero-shot learning model. A lead sheet is a melody with chord annotations. The model takes as input a lead sheet provided as a multi-track MIDI file, composed of a melodic track and a second track with block chords (*i.e.* non-arpeggiated). By interpreting the lead sheet as a low-rhythmicity multi-track piece, METEOR is able to orchestrate this lead sheet with specific instruments by increasing the rhythmicity.

8.2 Evaluating METEOR

In this section, we first present an objective evaluation to assess our model's performance in terms of fidelity and controllability in comparison with baseline models. Compared to the field of NLP where metrics have been developed and accepted to evaluate generative models, the MIR community still assumes that a human subjective evaluation is necessary to assess the quality of music generation. To this

end, our objective evaluation is supported by a user study conducted as a subjective evaluation on the tasks of re-orchestration and lead sheet orchestration.

8.2.1 Baseline models

We compare METEOR with two open-source and state-of-the-art style transfer models selected from the ones presented in Table 8.1³ and adapt them as multi-track re-orchestration models:

- FIGARO (von Rütte et al., 2023): This multi-track style transfer model is a texture style transfer model and can directly perform the re-orchestration task. For the evaluation of the controllability, we focus on the proposed “note density” controls, which corresponds to the rhythmic intensity in our work.
- AccoMontage-band (Zhao et al., 2024b): This model is originally designed to take a lead sheet as input and generate a multi-track pop band arrangement. We adapt this model to evaluate its performance as a re-orchestration system. To this end, we first pre-process a multi-track input its lead sheet representation *i.e.* we extract the melody as the skyline stream and the chords using the Chorder package⁴. This extracted lead sheet is then used as the input for the model which generates the re-orchestration of the initial input. Due to the potentially inaccurate assumption that the melody corresponds to the skyline stream, we perform earlier experiments using SheetSage (Donahue et al., 2022) to extract both melody and chords for the lead sheet extraction from the multi-track input. However, this approach produced lower-quality generations. The model can *a priori* perform texture transfer, but its controllability is limited. It transfers the texture of an existing music piece, referred to as the “texture donor”, onto the target musical content. This thus restricts texture controllability to the set of *pre-existing* texture donors.

We also consider a multi-track extension of MuseMorphose (Wu and Yang, 2023b), initially developed for piano textural style transfer, in which the original REMI tokenization is simply replaced with a REMI+ tokenization.

For the objective metrics, we compare these baselines with two versions of our model: “METEOR without inference guidance”, which includes bar- and track-level controllability but without melody constraints, and “METEOR” which includes these melody constraints.

³BandControlNet (Luo et al., 2024), PopMAG (Ren et al., 2020) and GetMUSIC (Lv et al., 2023) were not included as their codes or checkpoints were not publicly available at the time of this work.

⁴<https://github.com/joshuachang2311/chorder>

8.2.2 Objective & subjective metrics

Objective metrics – We first consider objective metrics to evaluate the *fidelity* with respect to the reference piece, both overall and specifically for the melody. We also consider a metric to assess how realistic the inference of instruments is with respect to their actual pitch distributions.

- *Overall fidelity* – Following (von Rütte et al., 2023), we consider the overall fidelity $\varphi_{\text{overall}} \in [0, 1]$ as the chroma similarity between the original piece and the generation. Let $\mathbf{v}_b^{\text{init}}$ (resp. $\mathbf{v}_b^{\text{gen}}$) the chroma vectors of bar b of the initial (resp. generated) piece, including all the tracks, and N the number of bars. The chroma similarity is defined as the average of the bar-wise cosine similarities $S_C(\cdot, \cdot)$ between these chroma vectors:

$$\varphi_{\text{overall}} = \frac{1}{N} \sum_{b=1}^N S_C(\mathbf{v}_b^{\text{init}}, \mathbf{v}_b^{\text{gen}})$$

- *Melodic fidelity* – For a piece, let $X_{b,\text{mel}}$ be the token sequence representing the melody in bar b . For a track t in the generation, let $X_{b,t}$ the token sequence of one track t at this bar b . Let $d(\cdot, \cdot)$ be the normalized Levenshtein edit distance: given two sequences X_1 and X_2 , we normalize the raw distance by $\max(|X_1|, |X_2|)$ so that $|d(X_1, X_2)| \leq 1$. We define the melodic fidelity of a track t at a bar b as $d(X_{b,\text{mel}}, X_{b,t})$. Intuitively, by taking the minimum of these distances among the tracks, we aim at selecting the track which is playing the melody within a bar. Therefore, the smaller the distance, the greater the melodic fidelity. Namely, we define the melodic fidelity φ_b at a bar b as:

$$\varphi_b = 1 - \min_{t \in \text{tracks}} d(X_{b,\text{mel}}, X_{b,t})$$

Finally, we define the melodic fidelity $\varphi_{\text{mel}} \in [0, 1]$ of a full multi-track generation of N bars as the average of these bar-wise fidelities:

$$\varphi_{\text{mel}} = \frac{1}{N} \sum_{b=1}^N \varphi_b$$

- *Pitch distribution similarity per instrument* – To evaluate the re-orchestration instrumental “realistic”⁵, we compare the distribution of pitches per instruments between a generated content and a reference dataset. Let P_i (resp. Q_i) the distribution of pitches played by the instrument i in a reference dataset⁶ (resp.

⁵We call an inferred instrument “realistic” if its generated pitch distribution is similar to its observed pitch distribution

⁶This reference dataset includes the SymphonyNet dataset and an equal number of pieces from the LakhMIDI dataset as they are training datasets for METEOR or FIGARO.

in the generated music). We consider i an instrument among the I available instruments. For $\text{JSD}(\cdot \parallel \cdot)$ the Jensen-Shannon Divergence, we define the instrument pitch distribution similarity $\rho \in [0, 1]$:

$$\rho = \frac{1}{I} \sum_{i=1}^T (1 - \text{JSD}(D_i \parallel Q_i))$$

Regarding textural controllability, we then consider two bar-level metrics for polyphonicity and rhythmicity and two bar- and track-level metrics for average pitch and pitch diversity.

- *Bar-controllability* – Polyphonicity and rhythmicity are evaluated at the bar level by including all tracks. Following (Wu and Yang, 2023b), we define

$$\rho_{attr} = \text{SpearmanCorr}(\tilde{a}^{\text{attr}}, s^{\text{attr}})$$

where s^{attr} is the user-specified attribute class (*i.e.* polyphonicity or rhythmicity) and \tilde{a}^{attr} is the class computed from the model generations given the user inputs.

- *Track-controllability* – Average pitch and pitch diversity are also evaluated with a Spearman correlation between the user input and the class computed from the generation, for each track and each bar.

For this evaluation, each model generates 20 samples of 8 bars each from picked randomly reference pieces. The control signals are randomly set and the instruments automatically chosen by the models.

Subjective metrics – We then conduct a user study to compare METEOR with the two baseline models. We evaluate the quality of the generations on the task of re-orchestration (multi-track to multi-track) and lead sheet orchestration (lead sheet to multi-track). For both tasks, participants listen to a 8-bar long reference (multi-track piece or lead sheet) and samples generated by the 3 models (Section 8.2.1). In particular, the generated MIDI from all the models are converted into audio samples using the basic MuseScore sound fonts. For the first task, they are asked to rate the generation contents on a 6-point Likert scale from 0 (very low) to 5 (very high) based on the following criteria and guidelines:

- **Overall musicality:** how enjoyable is the music?
- **Naturalness of the generation:** to what degree does the piece meet your expectations for musical plausibility?
- **Textural fidelity with the reference:** how does the extract reflect the reference “mood” (calmness, energy...)?

- **Convincing use of instruments:** how well do the instruments blend together within the overall arrangement?
- **Content coherency with the reference:** how much do you recognize the reference by listening to the sample?

The same aspects are evaluated for the lead sheet orchestration task, without “textural fidelity” and with the additional criterion:

- **Creativity:** how inventive while being still pleasant to hear, is the audio extract?

We let the model choose the melodic track automatically (or randomly for FIGARO and AccoMontage-band) in the re-orchestration task. For METEOR, the textural parameters are kept identical to those of the reference, ensuring that the criterion of “textural fidelity” is appropriate. Instead, for lead sheet orchestration, we insert *a posteriori* the melodic track played by a synthesizer, following the method of AccoMontage-band. This ensures a fair comparison of all models in terms of melody perception by the listener, allowing for a focused comparison between the generated accompaniments.

Re-orchestration - Piece 3

You will be asked to rate 3 generated extracts in comparison with a reference music following these guidelines on a scale from 0 to 5 (0 = very poor; 5 = very good):

- **Overall musicality:** How enjoyable is the music?
- **Content coherency with the reference:** How much do you recognize the reference by listening to the audio extract? To what extent does the audio extract feel connected to the reference, rather than diverging into something unrelated? How close to the reference the musical content is?
- **Textural fidelity with the reference:** How accurately does the extract reflect the mood (e.g., calmness, tension, energy) of the reference? How close to the reference the musical texture is?
- **Use of the instruments:** How convincingly and realistically the instruments are utilized? How well are the instruments balanced in the mix? How well do the instruments blend together within the overall arrangement?
- **Naturalness:** To what degree does the piece meet your expectations for musical plausibility? How smooth and logical do the melodies, harmonies, and rhythms feel?

First, listen to the reference (you can then listen back if needed):

Listen and rate this extract:

	0	1	2	3	4	5
Overall musicality	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Content coherency with the reference	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Textural / mood fidelity with the reference	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Use of instruments	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Naturalness	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Previous Next

Figure 8.8: Screenshots of the user study for METEOR subjective evaluation.

The survey consists of 6 pieces for the re-orchestration task and 4 for lead sheet orchestration, chosen to ensure diversity. For each piece, the instrumentation is fixed for all models, including different cases: where the number of target instruments is smaller or greater than the source instruments. Each model generates four re-orchestrations for each 6 pieces. Participants are randomly assigned to one of the four groups, with each group evaluating a different set of samples. A total of 24 participants for the re-orchestration task, and 13 for lead sheet orchestration have answered the survey. They have various musical backgrounds, from individuals with no musical experience (15%) to professional musicians (8%), with a majority of amateur (46%) to intermediate musicians (31%). The user study has been conducted between January and February 2025 using the Université de Lille’s survey platform⁷ as shown in Figure 8.8. The audio samples presented to the users can be accessible on the demo website of this project⁸.

⁷<https://enquetes.univ-lille.fr/index.php/445726?lang=en>

⁸<https://dinhviettoanle.github.io/meteor/>

Model		Overall fidelity \uparrow	Melodic fidelity \uparrow	Instr. pitch similarity \uparrow	Bar-controllability \uparrow		Track-controllability \uparrow	
					Rhyth.	Polyph.	Pitch diver.	Avg. pitch
1	FIGARO	.735 \pm .24	.271 \pm .08	.617 \pm .14	.867	–	–	–
2	AccoMontage-band	.756 \pm .10	.338* \pm .09	.583 \pm .17	–	–	–	–
3	Multi-track MuseMorphose	.932 \pm .10	.527 \pm .13	.696 \pm .18	.941	.936	–	–
4	METEOR (w/o inference guidance)	.918 \pm .11	.491 \pm .16	.755 \pm .13	.972	.951	.929	.926
5	METEOR	.927 \pm .10	.632 \pm .18	.780 \pm .12	.950	.932	.897	.821
6	Multi-track <i>Flute-oboe duet</i>	.888	.479	–	.919	.710	–	–
7	MuseMorphose <i>Woodwind quintet</i>	.932	.519	–	.956	.875	–	–
8	<i>Classical orchestra</i> [†]	.947	.511	–	.921	.676	–	–
9	METEOR <i>Flute-oboe duet</i>	.837	.457	–	.967	.782	.949	.873
10	<i>Woodwind quintet</i>	.903	.519	–	.971	.860	.961	.853
11	(w/o infer. guidance) <i>Classical orchestra</i> [†]	.917	.493	–	.975	.898	.958	.798
12	METEOR <i>Flute-oboe duet</i>	.837	.650	–	.936	.715	.786	.711
13	<i>Woodwind quintet</i>	.909	.651	–	.927	.862	.889	.875
14	<i>Classical orchestra</i> [†]	.912	.720	–	.953	.867	.909	.780

Table 8.3: Objective metrics for the re-orchestration task, with automatic choice and user-defined ensembles. (*) we evaluate the generated content only, without the inserted melodic track. (†) Classical orchestra includes 11 instruments (4 woodwinds, 2 brasses, timpani, 4 strings).

8.2.3 Results

In this section, we present the results of our objective and subjective evaluations. Our results show that METEOR can outperform the baseline models in the re-orchestration task, both on the objective and subjective metrics. For the lead sheet orchestration task learned as a zero shot task, METEOR can perform as well as models specifically trained for this task.

8.2.3.1 Objective evaluation

Quantitative metrics for re-orchestration are summarized in Table 8.3 (rows 1–5). MuseMorphose and the two versions of METEOR manage to outperform baseline models in all metrics. With FIGARO, they outperform AccoMontage-band in pitch distribution fidelity, as both are trained on orchestral instruments while AccoMontage-band is trained on band instruments. MuseMorphose and the two METEORS achieve comparable overall fidelities and adding melodic constraints naturally leads to an improvement in melodic fidelity. It is worth noting that METEOR does not reach a perfect score: inference guidance does not prevent the melodic instrument from adding extra notes beyond the exact melody, a phenomenon which can be found in orchestral music, often referred to as “decorative melody” (Le et al., 2022).

Though, this increase in melodic fidelity results in a drop in controllability metrics compared to METEOR without melody. This may result from using independent control methods, either latent or token-based, for beat-, bar- and track-level attributes. The compatibility between latent space-based or token-based controls

remains unexplored and could be further investigated to improve the understanding of controllable models.

Instrumentation impact – We further study the impact of the chosen instrumentation on our models’ performances (Table 8.3, rows 6–14). We select three musical ensembles: woodwind duet, quintet, and classical orchestra, assigning the melody to the flute in each case. For METEORS, increasing the number of instruments helps the model maintaining better fidelity to the reference piece and improves bar-wise attributes. With more instruments, the model has a larger instrumental flexibility and a broader range of options to assign each track a part that aligns with the control signals. Moreover, all the models demonstrate better bar-wise polyphonicity controllability when the instrumentation is chosen automatically (rows 3–5) compared to each user-defined ensembles. In other words, they manage to effectively select the most suitable ensemble to match the requested polyphonicity.

	Melodic instrument	Average note in instr. range (register bounds)	Average pitch in generations	Out of range generated notes
1	Flute	F5 (B3-C7)	D5	0.0%
2	Bassoon	E3 (Bb1-B4)	Bb2	4.8%
3	Trumpet	A4 (F#3-C6)	G4	4.3%
4	Violin	A5 (G3-B7)	C5	1.6%
5	Cello	Bb3 (C2-A5)	F3	3.8%

Table 8.4: Average pitch of melodic instruments with octave inference in the generated music by METEOR compared to their real instrumental range.

Melodic instrument range playability – We then study the playability of generations in terms of physical constraints of the melodic instrument. Unlike generic instruments such as synthesizers (Luo et al., 2024; Zhao et al., 2024b), orchestral instruments are limited in their range and usually play in a specific register (Rimsky-Korsakov, 1964). To evaluate such range playability, we generate five extracts from the same original reference without textural control attributes and assign the melody to an instrument. Based on our pitch class-based tokenization, we let the model infer the <Octave> tokens of the melody notes, while the other components (pitch class and duration) are enforced during inference guidance. As presented in Table 8.4, the model manages to generate instrumental parts which match with their usual register, with still a limited amount of out of range notes.

However, while the difference between the generated average pitch and its middle-range note is below a fourth for woodwinds and trumpet (rows 1–3), the average generated pitch for the cello and the violin (rows 4–5) are much lower than the theoretical average pitch (*e.g.* a sixth lower than the midpoint note of the violin’s full register). Violin parts and, more generally, string parts, are indeed typically written below the extreme high register of the instrument (Adler and Hesterman, 1989, p. 52):

It must be kept in mind that the extremely high range on any string instrument is difficult to control, and only in the past one hundred fifty years has it been used extensively.

Analysis of token embeddings – With a more intrinsic approach, latent spaces can often be structured by musically-meaningful concepts. Therefore, we further explore the internal representation of <Track> token embeddings. We present the projection through t-SNE of learned embeddings for each <Track> token in Figure 8.9. Interestingly, instruments having similar roles or pitch range or ambitus have embeddings which are close in this space. Instrument families, such as keyboard instruments, guitars, bass and treble (or high-pitch) instruments seem to stand out in the embedding space structure.

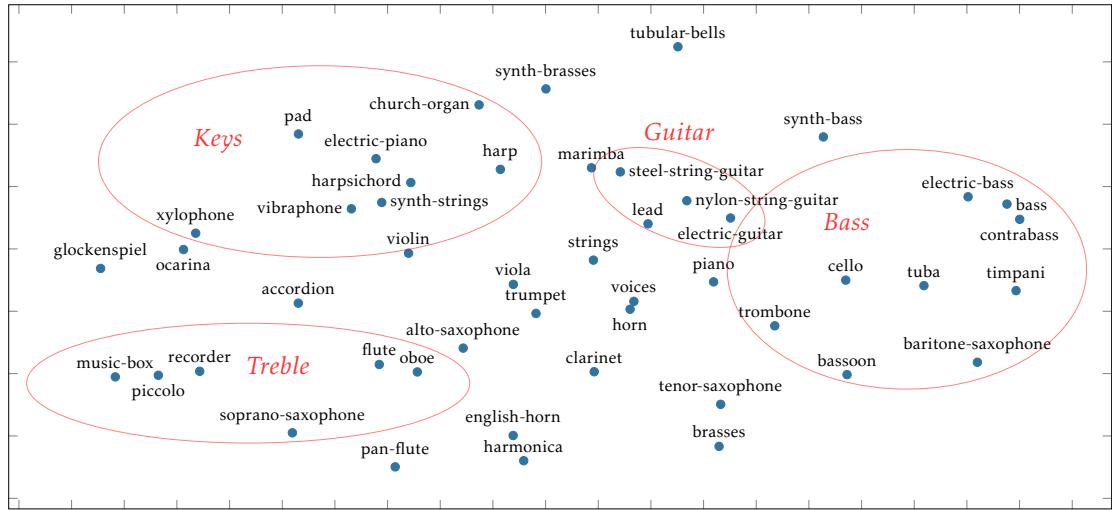


Figure 8.9: Projection through t-SNE of <Track> tokens embeddings. Instruments having similar roles are found to be close in this space (red groups).

8.2.3.2 Subjective evaluation

The results from our user study following the subjective metrics presented above on the task of re-orchestration and lead sheet orchestration are shown in Figure 8.10.

Re-orchestration task – METEOR outperforms in four of the five criteria on average (Figure 8.10, left). In particular, it holds significant advantage over the two other models on the overall musicality and naturalness (t-test: $p < .01$ for both). Further analysis highlights notable insights on other criteria.

- Texture fidelity. METEOR achieves significantly better results than the baseline models, in particular, compared to AccoMontage-band ($p < .01$). This may be attributed to the lead sheet input which simplifies the original piece by

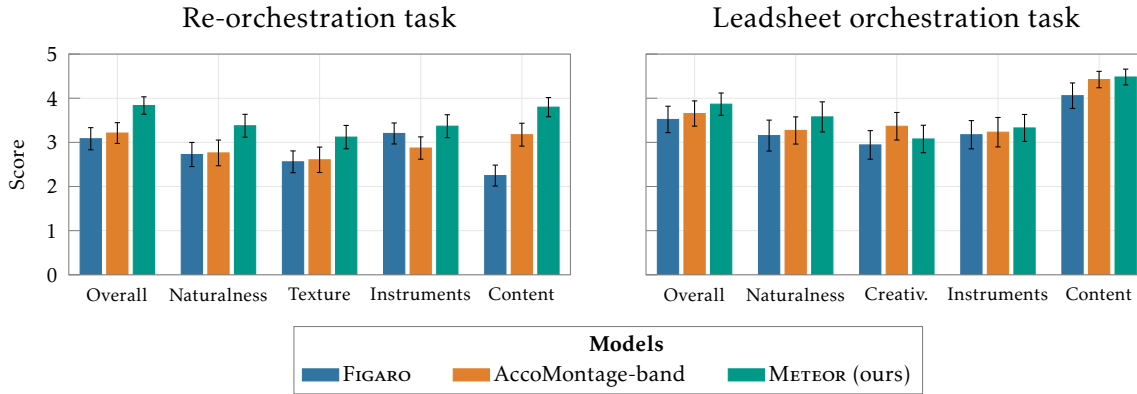


Figure 8.10: Average scores obtained on the re-orchestration and lead sheet orchestration tasks according to the user study. A 6-point Likert scale ranging from 0 (very low preference) to 5 (very high preference) is used.

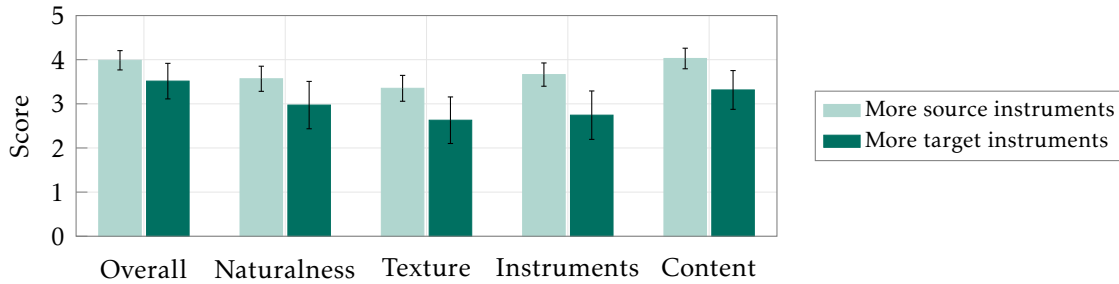


Figure 8.11: Impact of the number of source and target instruments on METEOR's orchestration.

reducing it to melody and chords, losing crucial textural characteristics and making it challenging for the model to generate a similar musical texture.

- **Instrumental use.** METEOR and FIGARO show comparable performances and both outperform AccoMontage-band. Given that the ensembles have been set to standard orchestral instruments, this shows that AccoMontage-band, which was trained with pop band instruments, can weakly adapt to unseen instruments. However, when comparing scenarios with varying numbers of target instruments relative to source instruments (Figure 8.11), METEOR performs better on instrumentation reduction and weaker when the target ensemble is larger than the reference on all criteria. This may be attributed to the need for generating longer sequences for these larger ensembles, highlighting a potential limitation in the model's ability to capture long-term dependencies.
- **Content coherency.** FIGARO has an average score significantly lower than METEOR and AccoMontage-band ($p < 1e-6$). As noted in the original study, FIGARO often fails to preserve the melody, highlighting that content coherency is strongly influenced by the retention of the melodic line. This effect is supported by the observation that FIGARO's content fidelity is more comparable to

other models in the lead sheet orchestration task (Figure 8.10, right), where the melody is inserted unchanged *a posteriori*.

Lead sheet orchestration task – Across all metrics, METEOR achieves performance ranging from comparable to better than the other models (Figure 8.10, right). In particular, while AccoMontage-band has been specifically trained on this task, it only outperforms METEOR on average on the creativity criterion. This zero-shot learning ability highlights METEOR’s versatility in performing tasks closely related to orchestration with comparable performances with state-of-the-art models.

8.3 Towards realistic and humanly playable symbolic music generation

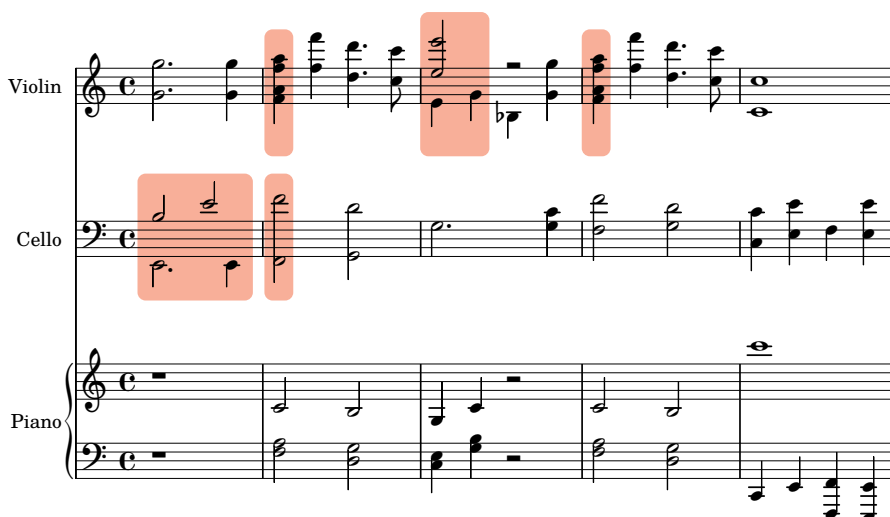


Figure 8.12: Extract of a score generated by METEOR with difficult fingerings which may not respect physical constraints of the instrument. These includes awkward fingerings (large gaps,...) or fingering transitions (quick transitions between chords and double stops, extreme ambitus,...). This extract is a sample from the user study.

In this chapter, we presented METEOR, a model for texture-controllable multi-track style transfer with a focus on melodic fidelity specifically trained for a task of re-orchestration. The model performs this task through token constraints at a bar- and track-level, with inference guidance for melodic fidelity. On a re-orchestration task, METEOR outperforms multi-track style transfer models on subjective and objective evaluations. We show that our model can be adapted into a lead sheet orchestrator and is comparable to a model trained for this task.

However, while this model could be improved in many ways, such as its controllability at the scale of the piece (Lu et al., 2023) or its long-term structure (Shih et al.,

2023), a limitation of many symbolic generation systems is their ability to generate realistic and *playable* sheet music. In our case, although METEOR succeeds in ensuring that melodic instruments fit their range constraints, their technical playability such as convenient fingerings, breath considerations, or logical articulations, have not been thoroughly studied and are systematically overlooked in music generation studies.

For example, Figure 8.12 shows a part of a score extracted from a generation from METEOR, where the requested ensemble is a piano trio. First, since the generation is performed auto-regressively, the model tends to produce multiple simultaneous notes even for instruments like the violin and cello, which are typically monophonic despite their ability to play chords. However, while the generated extract includes chords that fall within the playable pitch range of the instruments, they often involve complex fingerings or fingering transitions, making the resulting score challenging – if not impossible – for a single piano trio to perform.

Pursuing the generation of humanly playable music would necessitate rethinking the training data, as it would require high-quality sheet music that already incorporates well-considered instrument-related physical constraints. Though, going further, ensuring playability in relation to instrumental constraints, timbre effects, and instrument groupings (Goodchild and McAdams, 2018), as well as incorporating varying levels of performance difficulty (Vásquez et al., 2023) would represent a significant advancement towards automatic humanly playable orchestration and more broadly, symbolic music generation.

Chapter 9

Conclusion and future directions

9.1	Discussion and future directions	158
9.1.1	To what extent can NLP be applicable to MIR?	158
9.1.2	Towards lighter models	160
9.1.3	Towards model explainability	161
9.1.4	A need for benchmarking and comparative analysis	162
9.1.5	Exploring further models for symbolic MIR	163
9.2	Conclusion	164

The Music Information Retrieval (MIR) community has been increasingly adapting Natural Language Processing (NLP) approaches, in particular due to both similarities between language and music, but also the breakthrough and the performances of Transformer models in NLP. Thus, this thesis first proposed a structured organization of methods developed initially for NLP in the field of symbolic MIR following three axes: *sequential representations*, *models* and *tasks*. Beyond this structured overview, our technical contributions follow this organization and explore multiple levels of symbolic music content representations resulting from tools adapted from NLP approaches in MIR.

- We start by reconsidering the choice of low-level representation to increase the **sequential representation expressiveness**. We study how choosing a token alphabet based on interval instead of absolute pitch tokens and grouping tokens through BPE can impact the performance of models on downstream tasks.
- We then study **model mechanistic interpretability** by analyzing an abstract representation of symbolic music processed by the model: the attention mechanism. We study the attention behavior through attention span, as well as the relevance of attention heads in a model trained on functional harmony analysis.

- Finally, we manipulate representations for a **symbolic music generation task**. We present METEOR, developed for a task of re-orchestration with texture controllability and melodic fidelity and which relies on token constraints to perform this task.

The study of the relationships between NLP and MIR, along with the technical work presented in this thesis, raises several points of discussion and points toward potential future directions for further applications of NLP approaches within the MIR domain.

9.1 Discussion and future directions

The previous chapters outline various NLP approaches adapted to music data, resulting in the development of state-of-the-art tools for multiple symbolic MIR tasks. While these results are shown to be empirically effective, it is worth taking a step back on this practice by questioning the musical appropriation of tools that have originally been thought for natural language, given that both modalities still share several differences as discussed in [Chapter 2](#). We believe that taking these distinctions into consideration and incorporating common practices from the NLP field could help guide future directions in the MIR field.

9.1.1 To what extent can NLP be applicable to MIR?

Despite the numerous borrowings of the MIR community from NLP methods, the *direct* application of such NLP methods to symbolic music data can be questioned, due to a few technical differences between these two fields.

Data availability – Text data differ from symbolic music data by a much wider availability. For example, large language models such as GPT-3 ([Brown et al., 2020](#)) are trained on datasets containing 300 billion tokens. Compared to symbolic music, multiple models ([Ens and Pasquier, 2020](#); [von Rütte et al., 2023](#)) are trained on the LakhMIDI dataset which is composed of 175k songs, resulting in only 26M tokens using a basic MIDI-like tokenization. Moreover, while new text data are released in large amounts, contributing to extending datasets such as CommonCrawl based on publicly available text, symbolic music data is far less likely to grow at a comparable rate, since music transcriptions or compositions seen as symbolic music are released much less frequently than text data, such as textual exchanges, news articles, or blog posts, etc... Thus, there is a huge gap between the amount of data needed to train text models, on which Transformers are inherently efficient with such a large amount of data, and the availability of symbolic music data.

One way to expand symbolic music datasets could be through the use of audio datasets transcribed into symbolic music data. Audio-to-symbolic transcription tools have shown strong performance (Jamshidi et al., 2024) and could be leveraged to significantly increase the volume of symbolic music data. Hsiao et al. (2021) have notably built a symbolic music dataset through transcription, Pop1k7 which is however restricted to pop piano music. This further raises the issue of music *diversity* in available datasets, which may be limited not only in quantity but also in variety. Similarly to classical music data which is largely biased towards Western music (Hawthorne et al., 2019), contemporary music such as pop music may be stemming from non-western countries (Wang et al., 2020a) but is still restricted to tonal music. In generative tasks, these biases in training data are naturally reflected in the generated content. When using Transformers for symbolic music processing – which are inherently designed for very large datasets – it is therefore crucial to take into account the limited amount and restrained diversity of available musical data compared to textual data for which these models were originally developed.

Musical alphabet – The Latin alphabet, on which most NLP studies are based, is composed of homogeneous elements or characters. When applying NLP methods to musical data, there is a need of defining an alphabet. Though, musical alphabets based on the MIDI protocol are heterogeneous, consisting of multiple types of tokens, such as <Velocity> or <Duration>. Therefore, musical notes are represented by combinations of these atomic elements assembled in a unidimensional sequential way which also raises the issue of simultaneity in music presented in Section 2.2.1. Moreover, this combinatorial aspect is fundamental in music as two slightly different combinations can lead to radically different notes. In substance, this is comparable to Chinese characters that can be based on different radicals, leading to entirely different meanings (Wong et al., 2022). Such models have been developed for Chinese NLP (Tao et al., 2019), where a model not only takes as input words like standard NLP models, but also their decomposition into radicals.

Moreover, NLP models can handle noisy text as long as it does not constitute a significant portion of the data (Khayrallah and Koehn, 2018). However, even though symbolic music generation is often synonymous with *MIDI generation* as illustrated in Chapter 3, producing high-quality sheet music from this generated MIDI is challenging since raw MIDI data is *inherently* noisy when converted to sheet music. For example, enharmonic equivalences are generally not disambiguated, or *ritardando* or *accelerando* are often transcribed as successive precise tempo changes events. Building a music alphabet relying on sheet music-oriented formats, such as musicXML or **kern may be a starting point to generate directly high-quality sheet music. This may also be a step towards playable score generation, as underlined in Section 8.3. This raises however the issue of dealing with much lower amount of available sheet music-oriented training data, compared to MIDI data.

However, despite these differences, current practices and developments from the NLP field can presumably *inspire* future research in symbolic MIR, beyond a simple

application of NLP tools in MIR.

9.1.2 Towards lighter models

In response to increasing computational demands, especially with the rise of LLMs (Sevilla et al., 2022), various studies have focused on developing computationally efficient yet lighter models (Zhu et al., 2023a). Such optimizations leading to lighter models are desired for multiple reasons, including reducing training or inference time, as well as energy consumption or hardware costs. Multiple studies have explored model compression with **knowledge distillation** (Gou et al., 2021). This distillation process implements a lightweight student network which is trained to reproduce a pre-trained teacher network. This results in a lighter student network that is faster, in particular, faster at inference. In NLP, this has led to lightweight models such as DistilBERT (Sanh et al., 2020), 40% lighter than a BERT model. In contrast with distillation, **pruning methods** are based on altering an initial model by removing weights. Transformers are shown to be possibly pruned by removing most of the attention heads while keeping decent performance (Michel et al., 2019) and can help model explainability (Voita et al., 2019). Finally, model **design optimizations** for lightweight processing have been developed such as token skipping in PoWER-BERT (Goyal et al., 2020) or sliding window attention with cache in Mistral 7B (Jiang et al., 2023). In MIR, such advances towards lighter models have been tackled for audio music (Douwes et al., 2023).

In the field of symbolic MIR, models are currently not as big as NLP models which can reach 175B parameters in the case of GPT-3 (Brown et al., 2020). However, recent models are increasingly requiring higher computational power, such as the use of 4×40GB GPUs to train a multi-track generative model (Shu et al., 2024). Therefore, there is a growing recognition of the efficacy of lighter models for symbolic music data. This can be done through lighter *sequential representations* with the development of Compound Words (Hsiao et al., 2021) for smaller sequences, or smaller vocabulary resulting in smaller embeddings (Li et al., 2023b). Such improvement towards lighter *models’* internal representations can also occur through lighter attention mechanisms designed to take into account music repetitiveness (Yu et al., 2022). The development of *expressive* tokenization strategies presented in Chapter 6, such as interval-based tokenization or BPE supertokens, typically contributes to this objective by providing the model with more compact representations of music.

These studies emphasize a promising direction for the application of lighter models in symbolic MIR research. This direction may involve developing light methods specifically tailored for symbolic music, featuring fewer parameters, reduced memory usage, or shorter training or inference times. Such light models can have practical applications in real-time music generation, including improvisation where an instantaneous inference time is required.

9.1.3 Towards model explainability

Deep learning models are often perceived as black boxes, lacking explanations for the decisions they make. Several studies address the explainability aspects of NLP tools (Zhao et al., 2024a). Beyond model-agnostic tools, explainability methods that take a model’s internal structure into account can bring different ways of understanding it (Bereska and Gavves, 2024).

Concept-based interpretability involves assessing a model’s performance on probing tasks. They assume that its effectiveness is directly tied to its capacity to accurately represent high-level concepts from data. In NLP, these probing tasks can vary in nature (Conneau et al., 2018a), encompassing syntactic or semantic information retrieval (Kim et al., 2019). **Mechanistic interpretability** assumes that inner mechanisms of a model may behave following human intuitions. It is frequently conducted on word embeddings to assess how well a model represents words in relation to each other by examining relations like word similarity or analogies (Wang et al., 2019). The attention mechanism for Transformer-based models is notably studied in depth as it can reflect several links between tokens of a sentence. In Chapter 7, we have chosen this approach by analyzing models through its attention mechanism.

Recently, rationalization (*i.e.* providing a natural language explanation of the process) based on LLMs has been explored to provide musical descriptions of symbolic music data (Krol et al., 2022). LLMs developed for chat can also be evaluated in their reasoning (Yuan et al., 2024) in particular through **chain-of-thoughts** in audio (Deng et al., 2024b) or symbolic music (Zhou et al., 2024), assessing their musical understanding and knowledge for future human-computer co-creation systems. These chain-of-thoughts are notably a current trend, leading to initiatives such as a task at MIREX 2025 on audio music reasoning Q&A¹.

Going further, providing interpretable tools that align with human perception can be challenging due to the inherent subjectivity of music. In the context of music composition, stylistic aspects may offer different explanations, and certain passages may only be explained by artistic effects desired by the composer (Crocker, 1966). Despite this subjectivity and artistic aspect present in music, studying the explainability of tools for symbolic music can be a way to gain a better understanding of how models process music data. For instance, analyzing models on simple tasks such as style classification can highlight or confirm musicological characteristics in a particular style. Only a few studies have considered linking a model’s behavior with music theory aspects such as cadences (Loiseau et al., 2021) or chord progressions (Cosme-Clifford et al., 2023). Similarly, with the increasing popularity of text-to-music systems, interpreting models on such tasks may reveal relations between specific words with the resulting generated content, potentially leading to questions regarding biases within the currently available datasets of symbolic

¹https://www.music-ir.org/mirex/wiki/2025:Music_Reasoning_QA

music. For example, the word “sadness” is often simplistically associated with minor keys; though, it is rarely limited to tonality alone². Understanding which musical features are reflected in music generated by a text-to-music system prompted with such terms would be a valuable contribution. The MIR community is more and more encouraging model explainability, for example with a tutorial at ISMIR 2025³.

9.1.4 A need for benchmarking and comparative analysis

Benchmarks (*i.e.* commonly accepted combinations of datasets, tasks, and evaluation metrics against which new models can be tested) are crucial for meaningful model comparisons. In several other research domains, such benchmarks do exist. In image processing, there are historically established datasets like MNIST (Lecun et al., 1998) and ImageNet (Deng et al., 2009) that have served as widely accepted benchmarks for evaluating model performance. The NLP community has introduced several benchmarks such as GLUE (Wang et al., 2018) to evaluate language understanding. Other specific NLP benchmarks have also been developed, such as cross-lingual benchmarks (Liang et al., 2020b) or domain-specific benchmarks (Peng et al., 2019). In the music audio domain, common benchmarks are starting to gain prominence. The HEAR challenge (Turian et al., 2022) includes 19 evaluation tasks and brings together multiple models for comparative analysis. Such benchmark have begun to be built in audio music, in particular to evaluate latent diffusion-based text-to-music systems⁴.

In symbolic MIR, there is currently an apparent lack of **standardized benchmarks**. Bundling of datasets, tasks, and evaluation metrics for symbolic music data may provide frameworks to compare and evaluate models. The re-introduction of MIREX challenges⁵ in 2024 is an encouraging step towards model benchmarking. However, such challenges have mainly covered audio tasks. With the recent spread of text LLMs capable of processing ABC notation, ZIQI-Eval (Li et al., 2024) has been proposed to objectively compare models trained to answer multiple choice music-related questionnaires.

The question of model evaluation is fundamental. Subjectivity is often present in music, both in analysis tasks, such as functional harmony analysis in which annotator biases can emerge, and in generation tasks. Evaluation of generative systems through listening tests can be even more subjective (Yannakakis and Martínez, 2015), in particular when performed by non-experts (Amabile, 1982). Despite the fact that objective evaluation metrics have been proposed (Wu and Yang, 2020; Kumar and Sarmiento, 2023), MIREX’s symbolic music generation tasks still rely on listening tests for evaluation. Valuable contributions regarding these benchmarking issues can

²For example, Chopin’s Étude Op. 10, No. 3 in E major is often identified as “Tristesse” (sadness).

³<https://ismir2025.ismir.net/program-tutorials#tutorial-2>

⁴https://github.com/haoheliu/audioldm_eval

⁵<https://www.music-ir.org/mirex>

be an evaluation toolkit library aiming at retrieving objective features from generated pieces and comparing them to those extracted from a test set. Symbotunes (Skiers̄ et al., 2024), a standardized framework aiming at manipulating symbolic music generative models, may be a first step towards such toolkit but lacks the evaluation part. However, this may explain the challenges in establishing such music benchmarks: the inherent subjectivity of music aesthetics restricts the possibility of “reference data”, which are essential for model evaluation.

Another key challenge in music generation is that each model is typically specialized in a specific task. For example in Chapter 7, where we introduced the new task of re-orchestration, the choice of baseline models was limited to those adaptable to re-orchestration, though not specifically designed for it. For fair benchmarks, introducing objective auxiliary tasks for MIR may allow models to be evaluated on objective metrics in addition to their primary task.

9.1.5 Exploring further models for symbolic MIR

Beyond improving existing MIR models, several NLP models implement mechanisms or optimizations that can be relevant to symbolic music data. The **Longformer** model (Beltagy et al., 2020) aims to represent long documents by implementing linear complexity attention. Moreover, it also manages to perform well on character-level language modeling tasks. These two characteristics are fundamental in symbolic music, as long-range dependency is essential in musical token sequences. Additionally, unlike text where words are often considered as basic tokens, such grouping is less direct in music, so that symbolic music tasks are more similar to textual character-level tasks. On the representation side, **BERT-sentence** (Reimers and Gurevych, 2019) may be relevant in the field of symbolic MIR. This model builds embeddings for entire sentences and performs comparisons between pairs of sentences with a faster computing time. In symbolic music, where segmentation is a recurrent issue, such textual sentence-derived representation holds potential relevance. In more practical cases, pattern matching is often used in incipit search engines such as RISM⁶: an embedding-based query method may instead help improving the tool’s flexibility.

Finally, beyond NLP and the excitement of the general public for tools based on natural language generation, another trend stemming from research studies is image generation, in particular, text-to-image. Image processing models have already been used for symbolic music, including convolutional neural networks (Choi et al., 2017), and the recent rise of **diffusion models** in this field has motivated its adaption for music. Numerous recent models integrate state-of-the-art techniques from NLP and image processing, using diffusion models coupled with Transformer blocks for music generation (Li and Sung, 2023a; Min et al., 2023), also leading to tutorials on diffusion models for music at ISMIR 2024⁷. Therefore, as observed in recent

⁶<https://opac.rism.info>

⁷<https://ismir2024.ismir.net/tutorials#page-section-2>

publications and preprints (Figure 1.1), one current trend from recent MIR studies is to adapt such diffusion models initially developed for images to process music, in the same way as state-of-the-art NLP models have been adapted for symbolic music.

An interesting point is that these various applications of techniques from other fields remain *adaptations* for music data. In contrast, mechanisms such as convolutions for images or attention for sequential data were specifically conceived and designed for their respective data types. This raises a long-term question for the MIR community: is it possible to develop a mechanism tailored specifically to music, taking into consideration its unique characteristics without relying on adaptations of existing models?

9.2 Conclusion

From the work conducted and presented throughout this thesis, we aim to convey two main takeaways beyond these practical contributions presented in Part II and precise future directions outlined in Section 9.1.

Components of a MIR research study – Beyond the application of NLP tools for MIR, we think that the development of a MIR research study, in the context of study based on machine learning for symbolic music, should be guided by the three axes outlined in the thesis (Part I). Accordingly, the following questions must be explicitly addressed:

1. What **task** is being performed, and how precisely is it defined?
2. What **representation** is used to encode musical content, and what motivates this choice?
3. What **model** is employed, and why is it suitable for the chosen task and representation?

These questions then typically guide the choice of evaluation methods (subjective user study, objective metrics, ...) and help define the musical scope regarding the data involved. Ultimately, they can provide clear directions towards identifying and highlighting the core contribution of a research study in MIR.

NLP methods as tools rather than main motivations – High-level parallels do exist between music and natural language. However, when applied to the field of computer science, in particular in the era of deep learning models, the set of existing NLP methods seem to be more beneficial and judicious when considered as a toolbox, rather than a motivation by itself.

Natural language and symbolic music still holds their specificities or technical constraints, so that what is applicable in NLP is not necessary directly transferable to

MIR, or may not be musically meaningful. For example, although musical harmony is often compared to grammar in language (Section 2.2.2), our probing protocol developed in Section 7.4.3 – which assumed that harmony information could be found in a pre-trained model similarly to grammar in language models – does not seem to confirm this parallel.

Instead, as MIR manipulates musical data, the main motivation guiding a MIR study should be *musical questions* and NLP can provide technical approaches address these issues. Such musical questions can arise from the considered task, as illustrated in Chapter 8, where technical tools from NLP, such as tokenization or attention-based models in our contribution, serve only as tools to address the presented musical issues. Beyond NLP, this perspective of considering external fields as *toolboxes* is typically what drives the MIR community to adopt diffusion models for music processing as mentioned in future directions proposed in Section 9.1.5. Initially developed for image processing, their performance in that domain largely explains their growing popularity in MIR, rather than any direct parallel between music and image which remains far from straightforward. Therefore, we believe that adapting tools from other fields is judicious and should be continued, as long as their application remains guided first and foremost by musical questions.

Ultimately, as illustrated throughout this thesis, the field of NLP still remains a rich toolbox and offers a versatile framework of methods from which the MIR community can draw inspiration to design new approaches suited to the particular challenges of symbolic music analysis or generation.

Bibliography

Publications

Le, D.-V.-T.; Bigo, L.; Herremans, D.; and Keller, M. 2025a. [Natural Language Processing Methods for Symbolic Music Generation and Information Retrieval: A Survey](#). *ACM Computing Surveys*, 57(7).

Le, D.-V.-T.; Bigo, L.; and Keller, M. 2024. [Analyzing Byte-Pair Encoding on Monophonic and Polyphonic Symbolic Music: A Focus on Musical Phrase Segmentation](#). In *Proceedings of the 3rd Workshop on NLP for Music and Audio (NLP4MusA)*, pages 69–74, Oakland, USA. Association for Computational Linguistics.

Le, D.-V.-T.; Bigo, L.; and Keller, M. 2025b. [Evaluating Interval-based Tokenization for Pitch Representation in Symbolic Music Analysis](#). In *Workshop Artificial Intelligence for Music at AAAI 2025*, Philadelphia, United States.

Le, D.-V.-T.; and Yang, Y.-H. 2025. [METEOR: Melody-aware Texture-controllable Symbolic Music Re-Orchestration via Transformer VAE](#). In *Proceedings of the 34th International Joint Conference on Artificial Intelligence (IJCAI), Special Track on AI, Arts and Creativity*, Montreal, Canada. International Joint Conferences on Artificial Intelligence Organization.

Posters & Other communications

Bigo, L.; Keller, M.; and Le, D.-V.-T. 2024. [Le langage des partitions musicales face à l'intelligence artificielle](#). *Culture et recherche*, pages 34–35.

Le, D.-V.-T.; Bigo, L.; and Keller, M. 2023. [Tokenization Informativeness and its Impact on Symbolic MIR Tasks](#). In *DMRN+ 18: Digital Music Research Network One-day Workshop 2023*, London, United Kingdom.

References

- Achtibat, R.; Hatefi, S. M. V.; Dreyer, M.; et al. 2024. [AttnLRP: attention-aware layer-wise relevance propagation for transformers](#). In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.
- Acosta, M.; Bukey, I.; and Tsai, T. J. 2022. [An Exploration of Generating Sheet Music Images](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 701–708. ISMIR.
- Adler, S.; and Hesterman, P. 1989. *The study of orchestration*, volume 2. WW Norton New York, NY.
- Adorno, T. W.; and Gillespie, S. 1993. [Music, Language, and Composition](#). *The Musical Quarterly*, 77(3):401–414.
- Agirre, E.; Gonzalez-Agirre, A.; Lopez-Gazpio, I.; et al. 2016. [SemEval-2016 Task 2: Interpretable Semantic Textual Similarity](#). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 512–524, San Diego, California. Association for Computational Linguistics.
- Agostinelli, A.; Denk, T. I.; Borsos, Z.; et al. 2023. [MusicLM: Generating Music From Text](#). *Preprint*, arXiv:2301.11325.
- Allegraud, P.; Bigo, L.; Feisthauer, L.; et al. 2019. [Learning Sonata Form Structure on Mozart's String Quartets](#). *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 2(1):82–96.
- Amabile, T. M. 1982. [Social psychology of creativity: A consensual assessment technique](#). *Journal of personality and social psychology*.
- Amrami, A.; and Goldberg, Y. 2019. [Towards better substitution-based word sense induction](#). *Preprint*, arXiv:1905.12598.
- Angioni, S.; Lincoln-DeCusatis, N.; Ibba, A.; and Recupero, D. R. 2023. [A transformers-based approach for fine and coarse-grained classification and generation of MIDI songs and soundtracks](#). *PeerJ Computer Science*, 9.
- Arras, L.; Horn, F.; Montavon, G.; Müller, K.-R.; and Samek, W. 2016. [Explaining Predictions of Non-Linear Classifiers in NLP](#). In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 1–7, Berlin, Germany. Association for Computational Linguistics.
- Bach, S.; Binder, A.; Montavon, G.; et al. 2015. [On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation](#). *PLOS ONE*, 10(7):1–46.
- Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. [Neural machine translation by jointly learning to align and translate](#). In *International Conference on Learning Representations (ICLR)*.
- Bai, H.; Wang, P.; Zhang, R.; and Su, Z. 2023. [SegFormer: A Topic Segmentation Model with Controllable Range of Attention](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(11):12545–12552.
- Bandy, J.; and Vincent, N. 2021. [Addressing "Documentation Debt" in Machine Learning: A Retrospective Datasheet for BookCorpus](#). In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

- Bao, H.; Huang, S.; Wei, F.; et al. 2019. [Neural melody composition from lyrics](#). In *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part I* 8, pages 499–511. Springer.
- Bassan, S.; Adi, Y.; and Rosenschein, J. 2022. [Unsupervised Symbolic Music Segmentation using Ensemble Temporal Prediction Errors](#). In *Interspeech 2022*, pages 2423–2427.
- Belrose, N.; Furman, Z.; Smith, L.; et al. 2023. [Eliciting Latent Predictions from Transformers with the Tuned Lens](#). *Preprint*, arXiv:2303.08112.
- Beltagy, I.; Peters, M. E.; and Cohan, A. 2020. [Longformer: The Long-Document Transformer](#). *Preprint*, arXiv:2004.05150.
- Benward, B. 2018. *Music in theory and practice*. McGraw Hill Higher Education.
- Bereska, L.; and Gavves, E. 2024. [Mechanistic Interpretability for AI Safety - A Review](#). *Transactions on Machine Learning Research*.
- Bernstein, L. 1959. *The joy of music*. Simon and Schuster.
- Bernstein, L. 1976. *The unanswered question: Six talks at Harvard*, volume 33. Harvard University Press.
- Bertiaux, A.; Gabrielli, F.; Giraud, M.; and Levé, F. 2023. [Associations between music training and the dynamics of writing music by hand](#). *Musicae Scientiae*, 27(1):247–259.
- Besson, M.; and Schön, D. 2001. [Comparison between language and music](#). *Annals of the NY Academy of Sciences*, 930(1):232–258.
- Bhandari, K.; Roy, A.; Wang, K.; et al. 2025. [Text2midi: Generating Symbolic Music from Captions](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(22):23478–23486.
- Bockmon, G. A.; and Starr, W. J. 1962. *Perceiving music: problems in sight and sound*. Harcourt, Brace & World.
- Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. [Enriching Word Vectors with Subword Information](#). *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Boulanger-Lewandowski, N.; Bengio, Y.; and Vincent, P. 2012. [Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription](#). In *International Conference on Machine Learning (ICML)*.
- Bradshaw, L.; and Colton, S. 2025. [Aria-MIDI: A Dataset of Piano MIDI Files for Symbolic Music Modeling](#). In *International Conference on Learning Representations (ICLR)*.
- Briot, J.-P.; Hadjeres, G.; and Pachet, F.-D. 2020. *Deep learning techniques for music generation*, volume 1. Springer.
- Brown, S. 2017. [A joint prosodic origin of language and music](#). *Frontiers in psychology*, 8:1894.
- Brown, T.; Mann, B.; Ryder, N.; et al. 2020. [Language Models are Few-Shot Learners](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901. Curran Associates, Inc.
- Brunner, G.; Konrad, A.; Wang, Y.; and Wattenhofer, R. 2018. [MIDI-VAE: Modeling Dynamics and Instrumentation of Music with Applications to Style Transfer](#). In *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 747–754. ISMIR.

- Brunner, G.; Wang, Y.; Wattenhofer, R.; and Wiesendanger, J. 2017. [JamBot: Music Theory Aware Chord Based Generation of Polyphonic Music with LSTMs](#). In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 519–526.
- Bubeck, S.; Chandrasekaran, V.; Eldan, R.; et al. 2023. [Sparks of Artificial General Intelligence: Early experiments with GPT-4](#). *Preprint*, arXiv:2303.12712.
- Cacavas, J. 1975. *Music arranging and orchestration*. Alfred Music.
- Calvo-Zaragoza, J.; and Rizo, D. 2018. [End-to-End Neural Optical Music Recognition of Monophonic Scores](#). *Applied Sciences*, 8(4).
- Cancino-Chacón, C. E.; Peter, S. D.; Karystinaios, E.; et al. 2022. [Partitura: A Python Package for Symbolic Music Processing](#). In *Proceedings of the Music Encoding Conference (MEC2022)*, Halifax, Canada.
- Caplin, W. E. 1998. *Classical form: A theory of formal functions for the instrumental music of Haydn, Mozart, and Beethoven*. Oxford University Press.
- Cardoso, I.; O. Moraes, R.; and N. Ferreira, L. 2024. [The NES Video-Music Database: A Dataset of Symbolic Video Game Music Paired with Gameplay Videos](#). In *Proceedings of the 19th International Conference on the Foundations of Digital Games, FDG '24*, New York, NY, USA. Association for Computing Machinery.
- Carr, D. 2004. [Music, Meaning, and Emotion](#). *The Journal of Aesthetics and Art Criticism*, 62(3):225–234.
- Celikyilmaz, A.; Clark, E.; and Gao, J. 2021. [Evaluation of Text Generation: A Survey](#). *Preprint*, arXiv:2006.14799.
- Čenkerová, Z. 2017. [Melodic Segmentation: Structure, Cognition, Algorithms](#). *Musicologica Brunensia*, 52(1):53–61.
- Chang, C.-J.; Lee, C.-Y.; and Yang, Y.-H. 2021. [Variable-Length Music Score Infilling via XLNet and Musically Specialized Positional Encoding](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 97–104. ISMIR.
- Chang, H.; Zhang, H.; Barber, J.; et al. 2023. [Muse: Text-To-Image Generation via Masked Generative Transformers](#). *Preprint*, arXiv:2301.00704.
- Chang, S.-H.; and Yu, N.-Y. 2022. [Computerized handwriting evaluation and statistical reports for children in the age of primary school](#). *Scientific Reports*, 12(1):15675.
- Chawla, K.; and Yang, D. 2020. [Semi-supervised Formality Style Transfer using Language Model Discriminator and Mutual Information Maximization](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2340–2354, Online. Association for Computational Linguistics.
- Chen, K.; Zhang, W.; Dubnov, S.; Xia, G.; and Li, W. 2019. [The Effect of Explicit Structure Encoding of Deep Neural Networks for Symbolic Music Generation](#). In *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)*, pages 77–84.
- Chen, T.-P.; and Su, L. 2018. [Functional Harmony Recognition of Symbolic Music Data with Multi-task Recurrent Neural Networks](#). In *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 90–97. ISMIR.
- Chen, T.-P.; and Su, L. 2019. [Harmony Transformer: Incorporating Chord Segmentation into Harmony Recognition](#). In *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pages 259–267. ISMIR.

- Chen, T.-P.; and Su, L. 2021. [Attend to chords: Improving harmonic analysis of symbolic music using transformer-based models](#). *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 4(1):1–13.
- Chen, Y.-H.; Huang, Y.-S.; Hsiao, W.-Y.; and Yang, Y.-H. 2020. [Automatic composition of guitar tabs by transformers and groove modeling](#). In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 756–763. ISMIR.
- Chiu, S.-C.; and Chen, M.-S. 2012. [A Study on Difficulty Level Recognition of Piano Sheet Music](#). In *2012 IEEE International Symposium on Multimedia*, pages 17–23.
- Cho, K.; van Merriënboer, B.; Gulcehre, C.; et al. 2014. [Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.
- Choi, K.; Fazekas, G.; and Sandler, M. 2016. [Text-based LSTM networks for automatic music composition](#). In *1st Conference on Computer Simulation of Musical Creativity*.
- Choi, K.; Fazekas, G.; Sandler, M.; and Cho, K. 2017. [Convolutional recurrent neural networks for music classification](#). In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2392–2396.
- Choi, K.; Park, J.; Heo, W.; Jeon, S.; and Park, J. 2021. [Chord Conditioned Melody Generation With Transformer Based Decoders](#). *IEEE Access*, 9:42071–42080.
- Chomsky, N. 1957. *Syntactic Structures*. De Gruyter Mouton, Berlin, New York.
- Chomsky, N. 1980. *Human Language and Other Semiotic Systems*, pages 429–440. Springer US, Boston, MA.
- Chou, Y.-H.; Chen, I.-C.; Ching, J.; Chang, C.-J.; and Yang, Y.-H. 2024. [MidiBERT-Piano: Large-scale Pre-training for Symbolic Music Classification Tasks](#). *Journal of Creative Music Systems*, 8(1).
- Chu, H.; Urtasun, R.; and Fidler, S. 2016. [Song From PI: A Musically Plausible Network for Pop Music Generation](#). *Preprint*, arXiv:1611.03477.
- Chuan, C.-H.; Agres, K.; and Herremans, D. 2020. [From context to concept: exploring semantic relationships in music with Word2Vec](#). *Neural Computing and Applications*, 32:1023–1036.
- Chung, J.; Ahn, S.; and Bengio, Y. 2017. [Hierarchical Multiscale Recurrent Neural Networks](#). In *International Conference on Learning Representations (ICLR)*.
- Chung, J.; Cho, K.; and Bengio, Y. 2016. [A Character-level Decoder without Explicit Segmentation for Neural Machine Translation](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1693–1703, Berlin, Germany. Association for Computational Linguistics.
- Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. [Empirical evaluation of gated recurrent neural networks on sequence modeling](#). In *NIPS 2014 Workshop on Deep Learning*.
- Cilibrasi, R.; Vitányi, P.; and Wolf, R. d. 2004. [Algorithmic Clustering of Music Based on String Compression](#). *Computer Music Journal*, 28(4):49–67.

- Clark, K.; Khandelwal, U.; Levy, O.; and Manning, C. D. 2019. [What Does BERT Look at? An Analysis of BERT's Attention](#). In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.
- Conklin, D.; and Witten, I. H. 1995. [Multiple viewpoint systems for music prediction](#). *Journal of New Music Research*, 24(1):51–73.
- Conneau, A.; Kruszewski, G.; Lample, G.; Barrault, L.; and Baroni, M. 2018a. [What you can cram into a single \$\&!#^*\$ vector: Probing sentence embeddings for linguistic properties](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.
- Conneau, A.; Rinott, R.; Lample, G.; et al. 2018b. [XNLI: Evaluating Cross-lingual Sentence Representations](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.
- Cooke, D. 1959. *The Language of Music*. Oxford University Press, New York.
- Cornelissen, B.; Zuidema, W.; and Burgoyne, J. A. 2020. [Mode classification and natural units in plainchant](#). In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 869–875. ISMIR.
- Corrêa, D. C.; Levada, A. L. M.; and da F. Costa, L. 2011. [Finding Community Structure in Music Genres Networks](#). In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 447–452. ISMIR.
- Corrêa, D. C.; and Rodrigues, F. A. 2016. [A survey on symbolic data-based music genre classification](#). *Expert Systems with Applications*, 60:190–210.
- Cosme-Clifford, N.; Symons, J.; Kapoor, K.; and White, C. W. 2023. [Musicological Interpretability in Generative Transformers](#). In *4th International Symposium on the Internet of Sounds*, pages 1–9.
- Cournut, J.; Bigo, L.; Giraud, M.; Martin, N.; and Régnier, D. 2021. [What are the most used guitar positions?](#) In *Proceedings of the 8th International Conference on Digital Libraries for Musicology, DLfM '21*, page 84–92, New York, NY, USA. Association for Computing Machinery.
- Couturier, L.; Bigo, L.; and Leve, F. 2022. [A Dataset of Symbolic Texture Annotations in Mozart Piano Sonatas](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 509–516. ISMIR.
- Crocker, R. 1966. *A History of Musical Style*. McGraw-Hill series in music. McGraw-Hill Book Company.
- Cuthbert, M. S.; and Ariza, C. 2010. [Music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data](#). In *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pages 637–642. ISMIR.
- Cífka, O.; Şimşekli, U.; and Richard, G. 2020. [Groove2Groove: One-Shot Music Style Transfer With Supervision From Synthetic Data](#). *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2638–2650.
- Dabre, R.; Chu, C.; and Kunchukuttan, A. 2020. [A Survey of Multilingual Neural Machine Translation](#). *ACM Computing Surveys*, 53(5).

- Dai, S.; Jin, Z.; Gomes, C.; and Dannenberg, R. 2021. [Controllable deep melody generation via hierarchical music structure representation](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 143–150. ISMIR.
- Dai, S.; Zhang, Z.; and Xia, G. G. 2018. [Music style transfer: A position paper](#). In *6th International Workshop on Musical Metacreation (MUME)*.
- Dai, Z.; Yang, Z.; Yang, Y.; et al. 2019. [Transformer-XL: Attentive Language Models beyond a Fixed-Length Context](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.
- Dalmazzo, D.; Déguernel, K.; and Sturm, B. L. T. 2024. [The Chordinator: Modeling Music Harmony by Implementing Transformer Networks and Token Strategies](#). In *Artificial Intelligence in Music, Sound, Art and Design*, pages 52–66, Cham. Springer Nature Switzerland.
- Dash, A.; and Agres, K. R. 2023. [AI-Based Affective Music Generation Systems: A Review of Methods, and Challenges](#). *Preprint*, arXiv:2301.06890.
- de Berardinis, J.; Meroño-Peñuela, A.; Poltronieri, A.; and Presutti, V. 2023. [ChoCo: a Chord Corpus and a Data Transformation Workflow for Musical Harmony Knowledge Graphs](#). *Scientific Data*, 10(1):641.
- de Gibert, O.; Nail, G.; Arefyev, N.; et al. 2024. [A New Massive Multilingual Dataset for High-Performance Language Technologies](#). In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1116–1128, Torino, Italia. ELRA and ICCL.
- Dempster, D. 1998. [Is there Even a Grammar of Music?](#) *Musicae Scientiae*, 2(1):55–65.
- Deng, J.; Dong, W.; Socher, R.; et al. 2009. [ImageNet: A large-scale hierarchical image database](#). In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.
- Deng, Q.; Yang, Q.; Yuan, R.; et al. 2024a. [ComposerX: Multi-Agent Symbolic Music Composition with LLMs](#). *Preprint*, arXiv:2404.18081.
- Deng, Z.; Ma, Y.; Liu, Y.; et al. 2024b. [MusiLingo: Bridging Music and Text with Pre-trained Language Models for Music Captioning and Query Response](#). In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3643–3655, Mexico City, Mexico. Association for Computational Linguistics.
- Dervakos, E.; Kotsani, N.; and Stamou, G. 2022. [Genre recognition from symbolic music with cnns: performance and explainability](#). *SN Computer Science*, 4(2):106.
- Deudon, M. 2018. [Learning semantic similarity in a continuous space](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31. Curran Associates, Inc.
- Deutsch, T.; Jasbi, M.; and Shieber, S. 2020. [Linguistic Features for Readability Assessment](#). In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–17, Seattle, WA, USA → Online. Association for Computational Linguistics.
- Devaney, J.; Arthur, C.; Condit-Schultz, N.; and Nisula, K. 2015. [Theme And Variation Encodings with Roman Numerals \(TAVERN\): A New Data Set for Symbolic Music Analysis](#). In *Proceedings of the 16th International Society for Music Information Retrieval Conference*, pages 728–734. ISMIR.

- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. [BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- D’Hooge, A.; Bigo, L.; and Déguernel, K. 2023. [Modeling Bends in Popular Music Guitar Tablatures](#). In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 741–748. ISMIR.
- Di, S.; Jiang, Z.; Liu, S.; et al. 2021. [Video Background Music Generation with Controllable Music Transformer](#). In *Proceedings of the 29th ACM International Conference on Multimedia, MM ’21*, pages 2037–2045, New York, NY, USA. Association for Computing Machinery.
- Dien, D.; Kiem, H.; and Van Toan, N. 2001. [Vietnamese Word Segmentation](#). In *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium*, volume 1, pages 749–756.
- Domingo, M.; García-Martínez, M.; Helle, A.; Casacuberta, F.; and Herranz, M. 2023. [How Much Does Tokenization Affect Neural Machine Translation?](#) In *Computational Linguistics and Intelligent Text Processing*, pages 545–554, Cham. Springer Nature Switzerland.
- Donahue, C.; Lee, M.; and Liang, P. 2020. [Enabling Language Models to Fill in the Blanks](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online. Association for Computational Linguistics.
- Donahue, C.; Mao, H. H.; Li, Y. E.; Cottrell, G.; and McAuley, J. 2019. [LakhNES: Improving Multi-instrumental Music Generation with Cross-domain Pre-training](#). In *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pages 685–692. ISMIR.
- Donahue, C.; Thickstun, J.; and Liang, P. 2022. [Melody transcription via generative pre-training](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 485–492. ISMIR.
- Dong, H.-W.; Chen, K.; Dubnov, S.; McAuley, J.; and Berg-Kirkpatrick, T. 2023. [Multitrack Music Transformer](#). In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.
- Dong, L.; Xu, S.; and Xu, B. 2018. [Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition](#). In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888.
- Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; et al. 2020. [An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale](#). In *International Conference on Learning Representations (ICLR)*.
- Douwes, C.; Bindi, G.; Caillon, A.; Esling, P.; and Briot, J.-P. 2023. [Is Quality Enough? Integrating Energy Consumption in a Large-Scale Evaluation of Neural Audio Synthesis Models](#). In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.
- Dowling, W. J.; and Fujitani, D. S. 1971. [Contour, Interval, and Pitch Recognition in Memory for Melodies](#). *The Journal of the Acoustical Society of America*, 49(2B):524–531.
- Downie, J. S. 1999. [Evaluating a simple approach to music information retrieval: Conceiving melodic n-grams as text](#). Faculty of Graduate Studies, University of Western Ontario London, Ont.
- Duan, W.; Yu, Y.; Zhang, X.; et al. 2023. [Melody Generation from Lyrics with Local Interpretability](#). *ACM Transactions on Multimedia Computing, Communications, and Applications*, 19(3).

- Ducros, J. 2012. *L'atonalisme. Et après ?* Collège de France.
- Eckstein, L.; and Reinfandt, C. 2006. *On Dancing about Architecture: Words and Music between Cultural Practice and Transcendence*. *Zeitschrift für Anglistik und Amerikanistik*, 54(1):1–8.
- Elkhatat, A. M.; Elsaid, K.; and Almeer, S. 2023. *Evaluating the efficacy of AI content detection tools in differentiating between human and AI-generated text*. *International Journal for Educational Integrity*, 19(1):17.
- Ens, J.; and Pasquier, P. 2020. *MMM : Exploring Conditional Multi-Track Music Generation with the Transformer*. Preprint, arXiv:2008.06048.
- Ens, J.; and Pasquier, P. 2021. *Building the MetaMIDI Dataset: Linking Symbolic and Audio Musical Data*. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 182–188. ISMIR.
- Farzaneh, M.; and Toroghi, R. M. 2020. *GGA-MG: Generative genetic algorithm for music generation*. Preprint, arXiv:2004.04687.
- Fernández, J. D.; and Vico, F. 2013. *AI methods in algorithmic composition: A comprehensive survey*. *Journal of Artificial Intelligence Research*, 48:513–582.
- Ferreira, P.; Limongi, R.; and Fávero, L. P. 2023. *Generating Music with Data: Application of Deep Learning Models for Symbolic Music Composition*. *Applied Sciences*, 13(7).
- Fornäs, J. 1997. *Text and music revisited*. *Theory, Culture & Society*, 14(3):109–123.
- Fradet, N.; Briot, J.-P.; Chhel, F.; El Fallah-Seghrouchni, A.; and Gutowski, N. 2021. *MidiTok: A Python package for MIDI file tokenization*. In *International Society for Music Information Retrieval Conference (ISMIR), Late-Breaking Demo Session*.
- Fradet, N.; Gutowski, N.; Chhel, F.; and Briot, J.-P. 2023a. *Byte Pair Encoding for Symbolic Music*. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2001–2020, Singapore. Association for Computational Linguistics.
- Fradet, N.; Gutowski, N.; Chhel, F.; and Briot, J.-P. 2023b. *Impact of Time and Note Duration Tokenizations on Deep Learning Symbolic Music Modeling*. In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 89–97. ISMIR.
- Frasconi, P.; Soda, G.; and Vullo, A. 2002. *Hidden markov models for text categorization in multi-page documents*. *Journal of Intelligent Information Systems*, 18:195–217.
- Fricke, L.; Gotham, M.; Ostermann, F.; and Vatolkin, I. 2024. *Adaptation and Optimization of AugmentedNet for Roman Numeral Analysis Applied to Audio Signals*. In *Artificial Intelligence in Music, Sound, Art and Design*, pages 146–161, Cham. Springer Nature Switzerland.
- Fu, Y.; Tanimura, Y.; and Nakada, H. 2023. *Improve symbolic music pre-training model using MusicTransformer structure*. In *2023 17th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pages 1–6.
- Gage, P. 1994. *A new algorithm for data compression*. *C Users Journal*, 12(2):23–38.
- Galassi, A.; Lippi, M.; and Torroni, P. 2021. *Attention in Natural Language Processing*. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10):4291–4308.

- Gan, Q.; Wang, S.; Wu, S.; and Zhu, J. 2025. [PianoMotion10M: Dataset and Benchmark for Hand Motion Generation in Piano Performance](#). In *International Conference on Learning Representations (ICLR)*.
- Garcia-Valencia, S. 2020. [Embeddings as representation for symbolic music](#). *Preprint*, arXiv:2005.09406.
- Gardner, M.; Grus, J.; Neumann, M.; et al. 2018. [AllenNLP: A Deep Semantic Natural Language Processing Platform](#). *Preprint*, arXiv:1803.07640.
- Gillick, J.; Roberts, A.; Engel, J.; Eck, D.; and Bamman, D. 2019. [Learning to Groove with Inverse Sequence Transformations](#). *Preprint*, arXiv:1905.06118.
- Giraud, M.; Groult, R.; Leguy, E.; and Levé, F. 2015. [Computational Fugue Analysis](#). *Computer Music Journal*, 39(2):77–96.
- Goel, K.; Vohra, R.; and Sahoo, J. K. 2014. [Polyphonic Music Generation by Modeling Temporal Dependencies Using a RNN-DBN](#). In *Artificial Neural Networks and Machine Learning – ICANN 2014*, pages 217–224, Cham. Springer International Publishing.
- Goodchild, M.; and McAdams, S. 2018. [Perceptual Processes in Orchestration](#). In *The Oxford Handbook of Timbre*. Oxford University Press.
- Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; et al. 2014. [Generative adversarial nets](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2672–2680, Cambridge, MA, USA. MIT Press.
- Gotham, M.; Micchi, G.; López, N. N.; and Sailor, M. 2023a. [When in Rome: A Meta-corpus of Functional Harmony](#). *Transactions of the International Society for Music Information Retrieval (TISMIR)*.
- Gotham, M.; Redbond, M.; Bower, B.; and Jonas, P. 2023b. [The “OpenScore String Quartet” Corpus](#). In *Proceedings of the 10th International Conference on Digital Libraries for Musicology, DLfM ’23*, pages 49–57, New York, NY, USA. Association for Computing Machinery.
- Gotham, M.; Tymoczko, D.; and Cuthbert, M. 2019. [The RomanText Format: A Flexible and Standard Method for Representing Roman Numerical Analyses](#). In *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pages 123–129. ISMIR.
- Gotham, M. R. H.; and Jonas, P. 2022. [The OpenScore Lieder Corpus](#). In *Music Encoding Conference Proceedings 2021*, pages 131–136. Humanities Commons.
- Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. [Knowledge distillation: A survey](#). *International Journal of Computer Vision*, 129:1789–1819.
- Gover, M.; and Zewi, O. 2022. [Music Translation: Generating Piano Arrangements in Different Playing Levels](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 36–43. ISMIR.
- Goyal, S.; Choudhury, A. R.; Raje, S.; et al. 2020. [PoWER-BERT: Accelerating BERT Inference via Progressive Word-vector Elimination](#). In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3690–3699. PMLR.
- Graham, S.; Berninger, V.; Weintraub, N.; and and, W. S. 1998. [Development of Handwriting Speed and Legibility in Grades 1–9](#). *The Journal of Educational Research*, 92(1):42–52.

- Grootendorst, M. 2022. [BERTopic: Neural topic modeling with a class-based TF-IDF procedure](#). Preprint, arXiv:2203.05794.
- Groves, R. 2013. [Automatic Harmonization Using a Hidden Semi-Markov Model](#). In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 9, pages 48–54.
- Guan, Y.; Zhao, J.; Qiu, Y.; Zhang, Z.; and Xia, G. 2018. [Melodic Phrase Segmentation By Deep Neural Networks](#). Preprint, arXiv:1811.05688.
- Guo, D.; Chen, B.; Lu, R.; and Zhou, M. 2020. [Recurrent Hierarchical Topic-Guided RNN for Language Generation](#). In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3810–3821. PMLR.
- Guo, R.; Herremans, D.; and Magnusson, T. 2019. [Midi Miner – A Python library for tonal tension and track classification](#). In *International Society for Music Information Retrieval Conference (ISMIR), Late-Breaking Demo Session*.
- Guo, R.; Simpson, I.; Kiefer, C.; Magnusson, T.; and Herremans, D. 2022. [MusIAC: An Extensible Generative Framework for Music Infilling Applications with Multi-level Control](#). In *Artificial Intelligence in Music, Sound, Art and Design*, pages 341–356, Cham. Springer International Publishing.
- Guo, Z.; Kang, J.; and Herremans, D. 2023. [A domain-knowledge-inspired music embedding space and a novel attention mechanism for symbolic music modeling](#). In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI), AAAI’23/IAAI’23/EAAI’23*. AAAI Press.
- Guthrie, D.; Allison, B.; Liu, W.; Guthrie, L.; and Wilks, Y. 2006. [A Closer Look at Skip-gram Modelling](#). In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC’06)*, Genoa, Italy. European Language Resources Association (ELRA).
- HaCohen-Kerner, Y. 2022. [Survey on profiling age and gender of text authors](#). *Expert Systems with Applications*, 199:117140.
- Hadjeres, G.; Pachet, F.; and Nielsen, F. 2017. [DeepBach: a Steerable Model for Bach Chorales Generation](#). In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1362–1371. PMLR.
- Hadjeres, G.; and Crestel, L. 2021. [The Piano Inpainting Application](#). Preprint, arXiv:2107.05944.
- Hadjeres, G.; and Nielsen, F. 2017. [Interactive Music Generation with Positional Constraints using Anticipation-RNNs](#). Preprint, arXiv:1709.06404.
- Han, S.; Ihm, H.; and Lim, W. 2023. [Systematic Analysis of Music Representations from BERT](#). Preprint, arXiv:2306.04628.
- Harris, Z. S. 1962. *String analysis of sentence structure*. Papers on formal linguistics ; no. 1. Mouton, The Hague.
- Haumann, N. T.; Vuust, P.; Bertelsen, F.; and Garza-Villarreal, E. A. 2018. [Influence of musical enculturation on brain responses to metric deviants](#). *Frontiers in neuroscience*, 12:218.
- Hawthorne, C.; Elsen, E.; Song, J.; et al. 2018. [Onsets and Frames: Dual-Objective Piano Transcription](#). In *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 50–57. ISMIR.

- Hawthorne, C.; Stasyuk, A.; Roberts, A.; et al. 2019. [Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset](#). In *International Conference on Learning Representations (ICLR)*.
- Hentschel, J.; Neuwirth, M.; and Rohrmeier, M. 2021. [The Annotated Mozart Sonatas: Score, Harmony, and Cadence](#). *Transactions of the International Society for Music Information Retrieval (TISMIR)*.
- Hentschel, J.; Rammos, Y.; Moss, F. C.; Neuwirth, M.; and Rohrmeier, M. 2024. [An Annotated Corpus of Tonal Piano Music from the Long 19th Century](#). *Empirical Musicology Review*, 18(1):84–95.
- Hepokoski, J.; and Darcy, W. 2006. *Elements of Sonata Theory: Norms, Types, and Deformations in the Late-Eighteenth-Century Sonata*. Oxford University Press.
- Hernandez-Olivan, C.; and Beltran, J. R. 2023. [Musicaiz: A python library for symbolic music generation, analysis and visualization](#). *SoftwareX*, 22:101365.
- Herremans, D.; and Chuan, C.-H. 2017. [Modeling musical context with Word2vec](#). In *International Workshop on Deep Learning and Music*.
- Herremans, D.; Chuan, C.-H.; and Chew, E. 2017. [A Functional Taxonomy of Music Generation Systems](#). *ACM Computing Surveys*, 50(5).
- Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. [GANs trained by a two time-scale update rule converge to a local nash equilibrium](#). In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17*, pages 6629–6640, Red Hook, NY, USA. Curran Associates Inc.
- Hieu Nguyen, N.; Nguyen, D. T.; and Nguyen, N. L.-T. 2025. [Vietnamese Words Are Not Constructed from Syllables: Rethinking the Role of Word Segmentation in Natural Language Processing for Vietnamese Texts](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(22):24069–24077.
- Higgins, K. M. 2012. *The Music between Us: Is Music a Universal Language?* University of Chicago Press.
- Hillewaere, R.; Manderick, B.; and Conklin, D. 2018. [Global Feature Versus Event Models for Folk Song Classification](#). In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, pages 729–734, Kobe, Japan. ISMIR.
- Hirai, T.; and Sawada, S. 2019. [Melody2vec: Distributed representations of melodic phrases based on melody segmentation](#). *Journal of Information Processing*, 27:278–286.
- Hirata, K.; Tojo, S.; and Hamanaka, M. 2022. [Music as Formal Language](#). In *Music, Mathematics and Language: The New Horizon of Computational Musicology Opened by Information Science*, pages 51–78, Singapore. Springer Nature Singapore.
- Hochreiter, S.; and Schmidhuber, J. 1997. [Long Short-Term Memory](#). *Neural Computation*, 9(8):1735–1780.
- Hsiao, W.-Y.; Liu, J.-Y.; Yeh, Y.-C.; and Yang, Y.-H. 2021. [Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 178–186.
- Huang, A.; Dinculescu, M.; Vaswani, A.; and Eck, D. 2018. [Visualizing music self-attention](#). In *Proc. NeurIPS Workshop on Interpretability and Robustness in Audio, Speech, and Language*, page 1.

- Huang, C.-Z. A.; Duvenaud, D.; and Gajos, K. Z. 2016. [ChordRipple: Recommending Chords to Help Novice Composers Go Beyond the Ordinary](#). In *Proceedings of the 21st International Conference on Intelligent User Interfaces, IUI '16*, pages 241–250, New York, NY, USA. Association for Computing Machinery.
- Huang, C.-Z. A.; Vaswani, A.; Uszkoreit, J.; et al. 2019. [Music Transformer](#). In *International Conference on Learning Representations*.
- Huang, C.-R.; and Xue, N. 2012. [Words without Boundaries: Computational Approaches to Chinese Word Segmentation](#). *Language and Linguistics Compass*, 6(8):494–505.
- Huang, H.-H.; Sun, C.-T.; and Chen, H.-H. 2010. [Classical Chinese Sentence Segmentation](#). In *CIPS-SIGHAN Joint Conference on Chinese Language Processing*.
- Huang, Y.-S.; and Yang, Y.-H. 2020. [Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions](#). In *Proceedings of the 28th ACM International Conference on Multimedia, MM '20*, pages 1180–1188, New York, NY, USA. Association for Computing Machinery.
- Huang, Z.; Liang, D.; Xu, P.; and Xiang, B. 2020. [Improve Transformer Models with Better Relative Position Embeddings](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3327–3335, Online. Association for Computational Linguistics.
- Hung, H.-T.; Ching, J.; Doh, S.; et al. 2021. [EMOPIA: A Multi-Modal Pop Piano Dataset For Emotion Recognition and Emotion-based Music Generation](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 318–325, Online. ISMIR.
- Huron, D. 1989. [Characterizing musical textures](#). In *International Computer Music Conference (ICMC 1989)*, pages 131–134.
- Huron, D.; et al. 1996. [The melodic arch in Western folksongs](#). *Computing in Musicology*, 10:3–23.
- Ioffe, S.; and Szegedy, C. 2015. [Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift](#). In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France. PMLR.
- Irani, H.; and Metsis, V. 2025. [Positional Encoding in Transformer-Based Time Series Models: A Survey](#). *Preprint*, arXiv:2502.12370.
- Jackendoff, R. 2009. [Parallels and Nonparallels between Language and Music](#). *Music Perception: An Interdisciplinary Journal*, 26(3):195–204.
- Jacovi, A.; and Goldberg, Y. 2020. [Towards Faithfully Interpretable NLP Systems: How Should We Define and Evaluate Faithfulness?](#) In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.
- Jain, S.; and Wallace, B. C. 2019. [Attention is not Explanation](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.
- Jamshidi, F.; Pike, G.; Das, A.; and Chapman, R. 2024. [Machine Learning Techniques in Automatic Music Transcription: A Systematic Survey](#). *Preprint*, arXiv:2406.15249.
- Jaques, N.; Gu, S.; Bahdanau, D.; et al. 2017. [Tuning recurrent neural networks with reinforcement learning](#). *5th International Conference on Learning Representations (ICLR) Workshop track*.

- Jasleen Kaur, J. R. S. 2014. [Emotion Detection and Sentiment Analysis in Text Corpus: A Differential Study with Informal and Formal Writing Styles](#). *International Journal of Computer Applications*, 101(9):1–9.
- Jauhiainen, T.; Lui, M.; Zampieri, M.; Baldwin, T.; and Lindén, K. 2019. [Automatic language identification in texts: a survey](#). *Journal of Artificial Intelligence Research*, 65(1):675–682.
- Jelodar, H.; Wang, Y.; Rabbani, M.; et al. 2021. [A NLP framework based on meaningful latent-topic detection and sentiment analysis via fuzzy lattice reasoning on youtube comments](#). *Multimedia Tools and Applications*, 80:4155–4181.
- Jeong, D.; Kwon, T.; Kim, Y.; Lee, K.; and Nam, J. 2019a. [VirtuosoNet: A Hierarchical RNN-based System for Modeling Expressive Piano Performance](#). In *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pages 908–915, Delft, The Netherlands. ISMIR.
- Jeong, D.; Kwon, T.; Kim, Y.; and Nam, J. 2019b. [Graph Neural Network for Music Score Data and Modeling Expressive Piano Performance](#). In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3060–3070. PMLR.
- Jhamtani, H.; Gangal, V.; Hovy, E.; and Nyberg, E. 2017. [Shakespearizing Modern Language Using Copy-Enriched Sequence to Sequence Models](#). In *Proceedings of the Workshop on Stylistic Variation*, pages 10–19, Copenhagen, Denmark. Association for Computational Linguistics.
- Ji, S.; Yang, X.; and Luo, J. 2023. [A Survey on Deep Learning for Symbolic Music Generation: Representations, Algorithms, Evaluations, and Challenges](#). *ACM Computing Surveys*, 56(1).
- Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; et al. 2023. [Mistral 7B](#). *Preprint*, arXiv:2310.06825.
- Jiang, J.; Xia, G.; and Berg-Kirkpatrick, T. 2020a. [Discovering Music Relations with Sequential Attention](#). In *Proceedings of the 1st Workshop on NLP for Music and Audio (NLP4MusA)*, pages 1–5, Online. Association for Computational Linguistics.
- Jiang, J.; Xia, G. G.; Carlton, D. B.; Anderson, C. N.; and Miyakawa, R. H. 2020b. [Transformer VAE: A Hierarchical Model for Structure-Aware and Interpretable Music Representation Learning](#). In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 516–520.
- Jin, C.; Tie, Y.; Bai, Y.; Lv, X.; and Liu, S. 2020. [A style-specific music composition neural network](#). *Neural Processing Letters*, 52:1893–1912.
- Jin, D.; Jin, Z.; Hu, Z.; Vechtomova, O.; and Mihalcea, R. 2022. [Deep Learning for Text Style Transfer: A Survey](#). *Computational Linguistics*, 48(1):155–205.
- Ju, Y.; Margot, S.; McKay, C.; Dahn, L.; and Fujinaga, I. 2020. [Automatic figured bass annotation using the new bach chorales figured bass dataset](#). In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 640–646. ISMIR.
- Ju, Z.; Lu, P.; Tan, X.; et al. 2022. [TeleMelody: Lyric-to-Melody Generation with a Template-Based Two-Stage Method](#). *Preprint*, arXiv:2109.09617.
- Jurafsky, D. 2000. *Speech & language processing*. Prentice Hall PTR.
- Kang, J.; Poria, S.; and Herremans, D. 2023. [Video2Music: Suitable Music Generation from Videos using an Affective Multimodal Transformer model](#). *Preprint*, arXiv:2311.00968.

- Karystinaios, E.; and Widmer, G. 2022. [Cadence Detection in Symbolic Classical Music using Graph Neural Networks](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 917–924. ISMIR.
- Karystinaios, E.; and Widmer, G. 2023. [Roman Numeral Analysis With Graph Neural Networks: Onset-Wise Predictions From Note-Wise Features](#). In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 597–604. ISMIR.
- Katharopoulos, A.; Vyas, A.; Pappas, N.; and Fleuret, F. 2020. [Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention](#). In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165. PMLR.
- Keerti, G.; Vaishnavi, A.; Mukherjee, P.; et al. 2022. [Attentional networks for music generation](#). *Multimedia Tools and Applications*, 81(4):5179–5189.
- Kermarec, M.; Bigo, L.; and Keller, M. 2022. [Improving Tokenization Expressiveness With Pitch Intervals](#). In *International Society for Music Information Retrieval Conference (ISMIR), Late-Breaking Demo Session*.
- Kessler, B.; Nunberg, G.; and Schutze, H. 1997. [Automatic Detection of Text Genre](#). In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 32–38, Madrid, Spain. Association for Computational Linguistics.
- Khayrallah, H.; and Koehn, P. 2018. [On the Impact of Various Types of Noise on Neural Machine Translation](#). In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia. Association for Computational Linguistics.
- Kilgour, K.; Zuluaga, M.; Roblek, D.; and Sharifi, M. 2019. [Fréchet Audio Distance: A Metric for Evaluating Music Enhancement Algorithms](#). *Preprint*, arXiv:1812.08466.
- Kim, N.; Patel, R.; Poliak, A.; et al. 2019. [Probing What Different NLP Tasks Teach Machines about Function Word Comprehension](#). In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 235–249, Minneapolis, Minnesota. Association for Computational Linguistics.
- Kingma, D. P.; and Welling, M. 2013. [Auto-encoding variational bayes](#). In *International Conference on Learning Representations (ICLR)*.
- Klein, J.; and Jacobsen, T. 2012. [Music is not a Language: Re-interpreting empirical evidence of musical 'syntax'](#).
- Kong, Q.; Li, B.; Chen, J.; and Wang, Y. 2020. [GiantMIDI-Piano: A large-scale midi dataset for classical piano music](#). *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 1(5):87–98.
- Koops, H. V.; De Haas, W. B.; Burgoyne, J. A.; et al. 2019. [Annotator subjectivity in harmony annotations of popular music](#). *Journal of New Music Research*, 48(3):232–252.
- Kosta, K.; Lu, W. T.; Medeot, G.; and Chanquion, P. 2022. [A deep learning method for melody extraction from a polyphonic symbolic music representation](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 756–763. ISMIR.

- Kovaleva, O.; Romanov, A.; Rogers, A.; and Rumshisky, A. 2019. [Revealing the Dark Secrets of BERT](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.
- Krausz, M. 2019. [Rightness and reasons: Interpretation in cultural practices](#). Cornell University Press.
- Krogh, A.; and Hertz, J. 1991. [A Simple Weight Decay Can Improve Generalization](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 4. Morgan-Kaufmann.
- Krol, S. J.; Llano, M. T.; and McCormack, J. 2022. [Towards the generation of musical explanations with GPT-3](#). In *International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar)*, pages 131–147. Springer.
- Kudo, T. 2018. [Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.
- Kuhl, P. K.; Tsao, F.-M.; and Liu, H.-M. 2003. [Foreign-language experience in infancy: Effects of short-term exposure and social interaction on phonetic learning](#). *Proceedings of the National Academy of Sciences*, 100(15):9096–9101.
- Kumar, A.; and Sarmiento, P. 2023. [From Words to Music: A Study of Subword Tokenization Techniques in Symbolic Music Generation](#). *Preprint*, arXiv:2304.08953.
- Kumar, H.; and Ravindran, B. 2019. [Polyphonic Music Composition with LSTM Neural Networks and Reinforcement Learning](#). *Preprint*, arXiv:1902.01973.
- Kupiec, J. 1992. [Robust part-of-speech tagging using a hidden Markov model](#). *Computer Speech & Language*, 6(3):225–242.
- Lahnala, A.; Kambhatla, G.; Peng, J.; et al. 2021. [Chord Embeddings: Analyzing What They Capture and Their Role for Next Chord Prediction and Artist Attribute Prediction](#). In *Artificial Intelligence in Music, Sound, Art and Design*, pages 171–186, Cham. Springer International Publishing.
- Latif, S.; Zaidi, A.; Cuayahuitl, H.; et al. 2023. [Transformers in Speech Processing: A Survey](#). *Preprint*, arXiv:2303.11607.
- Lazzari, N.; Poltronieri, A.; and Presutti, V. 2023. [Pitchclass2vec: Symbolic Music Structure Segmentation with Chord Embeddings](#). *Preprint*, arXiv:2303.15306.
- Le, D.-V.-T.; Giraud, M.; Levé, F.; and Maccarini, F. 2022. [A Corpus Describing Orchestral Texture in First Movements of Classical and Early-Romantic Symphonies](#). In *Proceedings of the 9th International Conference on Digital Libraries for Musicology, DLfM '22*, pages 27–35, New York, NY, USA. Association for Computing Machinery.
- Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. [Gradient-based learning applied to document recognition](#). *Proceedings of the IEEE*, 86(11):2278–2324.
- Lee, H.; Kim, T.; Kang, H.; et al. 2022. [ComMU: Dataset for Combinatorial Music Generation](#). *Advances in Neural Information Processing Systems (NeurIPS)*, 35:39103–39114.
- Lee, K. J. M.; Ens, J.; Adkins, S.; et al. 2025. [The GigaMIDI Dataset with Features for Expressive Music Performance Detection](#). *Transactions of the International Society for Music Information Retrieval (TISMIR)*.

- Lee, Y.-S.; Papineni, K.; Roukos, S.; Emam, O.; and Hassan, H. 2003. [Language Model Based Arabic Word Segmentation](#). In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 399–406, Sapporo, Japan. Association for Computational Linguistics.
- Lenz, J.; and Mani, A. 2024. [PerTok: Expressive Encoding and Modeling of Symbolic Musical Ideas and Variations](#). In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pages 981–988. ISMIR.
- Lerdahl, F. 2012. [Musical syntax and its relation to linguistic syntax](#). Collège de France.
- Lerdahl, F.; and Jackendoff, R. S. 1996. *A Generative Theory of Tonal Music*. MIT press, Cambridge, Massachusetts, USA.
- Lewis, M.; Liu, Y.; Goyal, N.; et al. 2019. [BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension](#). *Preprint*, arXiv:1910.13461.
- Li, J.; Yang, L.; Tang, M.; et al. 2024. [The Music Maestro or The Musically Challenged, A Massive Music Evaluation Benchmark for Large Language Models](#). *Preprint*, arXiv:2406.15885.
- Li, S.; and Sung, Y. 2023a. [MelodyDiffusion: Chord-Conditioned Melody Generation Using a Transformer-Based Diffusion Model](#). *Mathematics*, 11(8).
- Li, S.; and Sung, Y. 2023b. [MRBERT: Pre-Training of Melody and Rhythm for Automatic Music Generation](#). *Mathematics*, 11(4):798.
- Li, S.; and Sung, Y. 2023c. [Transformer-Based Seq2Seq Model for Chord Progression Generation](#). *Mathematics*, 11(5).
- Li, W.; Luo, H.; Lin, Z.; et al. 2023a. [A Survey on Transformers in Reinforcement Learning](#). *Transactions on Machine Learning Research*.
- Li, Y.; and Yang, T. 2018. [Word Embedding for Understanding Natural Language: A Survey](#). In *Guide to Big Data Applications*, pages 83–104, Cham. Springer International Publishing.
- Li, Y.; Li, S.; and Fazekas, G. 2023b. [An Comparative Analysis of Different Pitch and Metrical Grid Encoding Methods in the Task of Sequential Music Generation](#). *Preprint*, arXiv:2301.13383.
- Li, Y.; Li, S.; and Fazekas, G. 2023c. [Pitch Class and Octave-Based Pitch Embedding Training Strategies for Symbolic Music Generation](#). In *Proceedings of the 16th International Symposium on Computer Music Multidisciplinary Research (CMMR)*, pages 86–97, Tokyo, Japan. Zenodo.
- Liang, H.; Lei, W.; Chan, P. Y.; et al. 2020a. [PiRhDy: Learning Pitch-, Rhythm-, and Dynamics-aware Embeddings for Symbolic Music](#). In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, pages 574–582, New York, NY, USA. Association for Computing Machinery.
- Liang, X.; Zhao, Z.; Zeng, W.; et al. 2024. [PianoBART: Symbolic Piano Music Generation and Understanding with Large-Scale Pre-Training](#). *Preprint*, arXiv:2407.03361.
- Liang, Y.; Duan, N.; Gong, Y.; et al. 2020b. [XGLUE: A New Benchmark Dataset for Cross-lingual Pre-training, Understanding and Generation](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018, Online. Association for Computational Linguistics.
- Lidov, D. 1997. [Our time with the druids: What \(and how\) we can recuperate from our obsession with segmental hierarchies and other “tree structures”](#). *Contemporary Music Review*, 16(4):1–28.
- Liebel, L.; and Körner, M. 2018. [Auxiliary Tasks in Multi-task Learning](#). *Preprint*, arXiv:1805.06334.

- Lin, Z.; Chen, J.; Tang, B.; et al. 2024a. [Multi-View MidiVAE: Fusing Track- and Bar-View Representations for Long Multi-Track Symbolic Music Generation](#). In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 941–945.
- Lin, Z.-S.; Kuo, Y.-C.; Hung, T.-Y.; et al. 2024b. [S3: A Symbolic Music Dataset for Computational Music Analysis of Symphonies](#). In *International Society for Music Information Retrieval Conference (ISMIR), Late-Breaking Demo Session*.
- Liu, C.-H.; and Ting, C.-K. 2017. [Computational Intelligence in Music Composition: A Survey](#). *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(1):2–15.
- Liu, J.; Dong, Y.; Cheng, Z.; et al. 2022. [Symphony Generation with Permutation Invariant Language Model](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 551–558. ISMIR.
- Liu, Q.; Kusner, M. J.; and Blunsom, P. 2020. [A Survey on Contextual Embeddings](#). *Preprint*, arXiv:2003.07278.
- Liu, Y.; Cao, J.; Liu, C.; Ding, K.; and Jin, L. 2024. [Datasets for Large Language Models: A Comprehensive Survey](#). *Preprint*, arXiv:2402.18041.
- Liu, Y.; Ott, M.; Goyal, N.; et al. 2019. [RoBERTa: A Robustly Optimized BERT Pretraining Approach](#). *Preprint*, arXiv:1907.11692.
- Liumei, Z.; Fanzhi, J.; Jiao, L.; Gang, M.; and Tianshi, L. 2021. [K-means clustering analysis of Chinese traditional folk music based on midi music textualization](#). In *2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP)*, pages 1062–1066.
- Liutkus, A.; Cífka, O.; Wu, S.-L.; et al. 2021. [Relative Positional Encoding for Transformers with Linear Complexity](#). In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 7067–7079. PMLR.
- Loiseau, G.; Keller, M.; and Bigo, L. 2021. [What Musical Knowledge Does Self-Attention Learn?](#) In *Proceedings of the 2nd Workshop on NLP for Music and Spoken Audio (NLP4MusA)*, pages 6–10, Online. Association for Computational Linguistics.
- Lousseief, E.; and Sturm, B. 2019. [Mahlernet: Unbounded orchestral music with neural networks](#). In *the Nordic Sound and Music Computing Conference 2019 and the Interactive Sonification Workshop 2019*, pages 57–63.
- Lu, P.; Xu, X.; Kang, C.; et al. 2023. [MuseCoco: Generating Symbolic Music from Text](#). *Preprint*, arXiv:2306.00110.
- Luo, J.; Yang, X.; and Herremans, D. 2024. [BandControlNet: Parallel Transformers-based Steerable Popular Music Generation with Fine-Grained Spatiotemporal Features](#). *Preprint*, arXiv:2407.10462.
- Luo, J.; Yang, X.; Ji, S.; and Li, J. 2020. [MG-VAE: Deep Chinese Folk Songs Generation with Specific Regional Styles](#). In *Proceedings of the 7th Conference on Sound and Music Technology (CSMT)*, pages 93–106, Singapore. Springer Singapore.
- Lv, A.; Tan, X.; Lu, P.; et al. 2023. [GETMusic: Generating Any Music Tracks with a Unified Representation and Diffusion Framework](#). *Preprint*, arXiv:2305.10841.
- Lyu, Q.; Apidianaki, M.; and Callison-Burch, C. 2024. [Towards Faithful Model Explanation in NLP: A Survey](#). *Computational Linguistics*, 50(2):657–723.

- Ma, Y.; Øland, A.; Ragni, A.; et al. 2024. [Foundation Models for Music: A Survey](#). *Preprint*, arXiv:2408.14340.
- Madaan, A.; Setlur, A.; Parekh, T.; et al. 2020. [Politeness Transfer: A Tag and Generate Approach](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1869–1881, Online. Association for Computational Linguistics.
- Madjiheurem, S.; Qu, L.; and Walder, C. 2016. [Chord2vec: Learning musical chord embeddings](#). In *Constructive machine learning workshop at NIPS*.
- Makris, D.; Agres, K. R.; and Herremans, D. 2021. [Generating Lead Sheets with Affect: A Novel Conditional seq2seq Framework](#). In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.
- Makris, D.; Kaliakatsos-Papakostas, M.; Karydis, I.; and Kermanidis, K. L. 2019. [Conditional neural sequence learners for generating drums’ rhythms](#). *Neural Computing and Applications*, 31(6):1793–1804.
- Makris, D.; Zixun, G.; Kaliakatsos-Papakostas, M.; and Herremans, D. 2022. [Conditional drums generation using compound word representations](#). In *International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar)*, pages 179–194. Springer.
- Malandro, M. E. 2024. [Composer’s Assistant 2: Interactive Multi-Track MIDI Infilling with Fine-Grained User Control](#). *Preprint*, arXiv:2407.14700.
- Mann, A. 1987. *The study of fugue*. Courier Corporation.
- Mao, H. H.; Shin, T.; and Cottrell, G. 2018. [DeepJ: Style-Specific Music Generation](#). In *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, pages 377–382.
- Margulis, E. H. 2013. [Repetition and emotive communication in music versus speech](#). *Frontiers in Psychology*, 4:167.
- Masada, K.; and Bunesco, R. C. 2017. [Chord Recognition in Symbolic Music Using Semi- Markov Conditional Random Fields](#). In *Proceedings of the 18th International Society for Music Information Retrieval Conference*, pages 272–278. ISMIR.
- McCallum, A.; and Li, W. 2003. [Early results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons](#). In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 188–191.
- McClary, S. 2002. *Feminine endings: Music, gender, and sexuality*. U of Minnesota Press.
- McCulloch, W. S.; and Pitts, W. 1943. [A logical calculus of the ideas immanent in nervous activity](#). *The bulletin of mathematical biophysics*, 5:115–133.
- Medeot, G.; Cherla, S.; Kosta, K.; et al. 2018. [StructureNet: Inducing Structure in Generated Melodies](#). In *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 725–731. ISMIR.
- Mejova, Y. 2009. [Sentiment analysis: An overview](#). *University of Iowa, Computer Science Department*, 5:1–34.
- Melechovsky, J.; Roy, A.; and Herremans, D. 2024. [MidiCaps: A Large-Scale MIDI Dataset With Text Captions](#). In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pages 858–865. ISMIR.

- Merriam, A. P.; and Merriam, V. 1964. *The anthropology of music*. Northwestern University Press.
- Meyer, L. B. 1973. *Explaining Music: Essays and Explorations*, dgo - digital original, 1 edition. University of California Press.
- Micchi, G.; Gotham, M.; and Giraud, M. 2020. [Not All Roads Lead to Rome: Pitch Representation and Model Architecture for Automatic Harmonic Analysis](#). *Transactions of the International Society for Music Information Retrieval*.
- Micchi, G.; Kosta, K.; Medeot, G.; and Chanquion, P. 2021. [A deep learning method for enforcing coherence in Automatic Chord Recognition](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 443–451. ISMIR.
- Michel, P.; Levy, O.; and Neubig, G. 2019. [Are Sixteen Heads Really Better than One?](#) In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32. Curran Associates, Inc.
- Mielke, S. J.; Alyafeai, Z.; Salesky, E.; et al. 2021. [Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP](#). *Preprint*, arXiv:2112.10508.
- Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. [Efficient Estimation of Word Representations in Vector Space](#). *Preprint*, arXiv:1301.3781.
- Min, L.; Jiang, J.; Xia, G.; and Zhao, J. 2023. [Polyffusion: A Diffusion Model for Polyphonic Score Generation With Internal and External Controls](#). In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 231–238. ISMIR.
- Mogren, O. 2016. [C-RNN-GAN: A continuous recurrent neural network with adversarial training](#). In *Constructive Machine Learning Workshop (CML) at NIPS 2016*, page 1.
- Montavon, G.; Binder, A.; Lapuschkin, S.; Samek, W.; and Müller, K.-R. 2019. [Layer-Wise Relevance Propagation: An Overview](#), pages 193–209. Springer International Publishing, Cham.
- Muhamed, A.; Li, L.; Shi, X.; et al. 2021. [Symbolic Music Generation with Transformer-GANs](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 408–417.
- Müller, M. 2015. *Fundamentals of music processing: Audio, analysis, algorithms, applications*, volume 5. Springer.
- Nakamura, T.; Takamichi, S.; Tanji, N.; Fukayama, S.; and Saruwatari, H. 2023. [jaCappella Corpus: A Japanese a Cappella Vocal Ensemble Corpus](#). In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.
- Nápoles López, N.; Arthur, C.; and Fujinaga, I. 2019. [Key-Finding Based on a Hidden Markov Model and Key Profiles](#). In *Proceedings of the 6th International Conference on Digital Libraries for Musicology, DLFM '19*, pages 33–37, New York, NY, USA. Association for Computing Machinery.
- Narmour, E. 1990. *The analysis and cognition of basic melodic structures: The implication-realization model*. University of Chicago Press.
- Nasar, Z.; Jaffry, S. W.; and Malik, M. K. 2021. [Named Entity Recognition and Relation Extraction: State-of-the-Art](#). *ACM Computing Surveys*, 54(1).
- Nattiez, J.-J. 1990. *Music and discourse: Toward a semiology of music*. Princeton University Press.
- Neves, P. L. T.; Fornari, J.; and Florindo, J. B. 2022. [Generating music with sentiment using Transformer-GANs](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 717–725. ISMIR.

- Noh, S.-H. 2021. [Analysis of Gradient Vanishing of RNNs and Performance Comparison](#). *Information*, 12(11).
- nostalgebraist. 2020. [Interpreting GPT: the logit lens](#). *Less-Wrong*.
- Nápoles López, N. 2022. [Automatic Roman Numeral Analysis in Symbolic Music Representations](#). PhD Thesis, McGill University.
- Nápoles López, N.; Gotham, M.; and Fujinaga, I. 2021. [AugmentedNet: A Roman Numeral Analysis Network with Synthetic Training Examples and Additional Tonal Tasks](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 404–411, Online. ISMIR.
- Ogihara, M.; and Li, T. 2008. [N-Gram Chord Profiles for Composer Style Representation](#). In *Proceedings of the 9th International Conference on Music Information Retrieval*, pages 671–676. ISMIR.
- Oore, S.; Simon, I.; Dieleman, S.; Eck, D.; and Simonyan, K. 2018. [This time with feeling: Learning expressive musical performance](#). *Neural Computing and Applications*, 32:955–967.
- Ott, M.; Edunov, S.; Baevski, A.; et al. 2019. [fairseq: A Fast, Extensible Toolkit for Sequence Modeling](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.
- Ozaki, Y.; de Heer Kloots, M.; Ravignani, A.; and Savage, P. E. 2023. [Cultural evolution of music and language](#).
- Palmer, D. D. 2000. [Tokenisation and Sentence Segmentation](#). *Handbook of natural language processing*, page 11.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. [Bleu: a Method for Automatic Evaluation of Machine Translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Park, K.; Lee, J.; Jang, S.; and Jung, D. 2020. [An Empirical Study of Tokenization Strategies for Various Korean NLP Tasks](#). In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 133–142, Suzhou, China. Association for Computational Linguistics.
- Park, S.; Choi, E.; Kim, J.; and Nam, J. 2024. [Mel2Word: A Text-Based Melody Representation for Symbolic Music Analysis](#). *Music & Science*, 7.
- Pasquier, P.; Ens, J.; Fradet, N.; et al. 2025. [MIDI-GPT: A Controllable Generative Model for Computer-Assisted Multitrack Music Composition](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(2):1474–1482.
- Payne, C. 2019. [Musesnet](#). *OpenAI Blog*.
- Pearce, M. T. 2018. [Statistical learning and probabilistic prediction in music cognition: mechanisms of stylistic enculturation](#). *Annals of the New York Academy of Sciences*, 1423(1):378–395.
- Peng, Y.; Yan, S.; and Lu, Z. 2019. [Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets](#). In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy. Association for Computational Linguistics.
- Perotti, J. I.; and Billoni, O. V. 2020. [On the emergence of Zipf’s law in music](#). *Physica A Stat. Mech. Appl.*, 549:124309.

- Peters, J. 2016. *Fundamentals of Writing Four-part Harmony*. CreateSpace Independent Publishing Platform.
- Poliak, A. 2020. [A survey on Recognizing Textual Entailment as an NLP Evaluation](#). In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 92–109, Online. Association for Computational Linguistics.
- Pollastri, E.; and Simoncelli, G. 2001. [Classification of melodies by composer with hidden Markov models](#). In *International Conference on WEB Delivering of Music*, pages 88–95.
- Pulvermüller, F.; and Assadollahi, R. 2007. [Grammar or serial order?: Discrete combinatorial brain mechanisms reflected by the syntactic mismatch negativity](#). *Journal of cognitive neuroscience*, 19(6):971–980.
- Qin, Y.; Xie, H.; Ding, S.; et al. 2022. [Bar transformer: a hierarchical model for learning long-term structure and generating impressive pop music](#). *Applied Intelligence*, 53(9):10130–10148.
- Qiu, L.; Li, S.; and Sung, Y. 2021. [DBTMPE: Deep Bidirectional Transformers-Based Masked Predictive Encoder Approach for Music Genre Classification](#). *Mathematics*, 9(5).
- Qu, X.; Bai, Y.; Ma, Y.; et al. 2025. [MuPT: A Generative Symbolic Music Pretrained Transformer](#). In *International Conference on Learning Representations (ICLR)*.
- Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. [Improving language understanding by generative pre-training](#). *OpenAI*.
- Radford, A.; Wu, J.; Child, R.; et al. 2019. [Language Models are Unsupervised Multitask Learners](#). *OpenAI*.
- Raffel, C. 2016. [Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching](#).
- Raffel, C.; and Ellis, D. P. 2014. [Intuitive analysis, creation and manipulation of MIDI data with pretty_midi](#). In *International Society for Music Information Retrieval Conference (ISMIR), Late-Breaking Demo Session*, pages 84–93.
- Raffel, C.; Shazeer, N.; Roberts, A.; et al. 2020. [Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer](#). *J. Mach. Learn. Res.*, 21(1).
- Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. [SQuAD: 100,000+ Questions for Machine Comprehension of Text](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.
- Ramesh, A.; Pavlov, M.; Goh, G.; et al. 2021. [Zero-Shot Text-to-Image Generation](#). In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR.
- Randles, C.; and Sullivan, M. 2013. [How composers approach teaching composition: Strategies for music teachers](#). *Music Educators Journal*, 99(3):51–57.
- Reich, U.; and Rohrmeier, M. 2024. [Beats in Time Across Music and Language](#). *OSF Preprints*.
- Reimers, N.; and Gurevych, I. 2019. [Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks](#). *Preprint*, arXiv:1908.10084.
- Ren, Y.; He, J.; Tan, X.; et al. 2020. [Popmag: Pop music accompaniment generation](#). In *Proceedings of the 28th ACM international conference on multimedia*, pages 1198–1206.

- Retkowski, J.; Stępnia, J.; and Modrzejewski, M. 2025. [Frechet Music Distance: A Metric For Generative Symbolic Music Evaluation](#). In *Workshop Artificial Intelligence for Music at AAAI 2025*, Philadelphia, United States.
- Rimsky-Korsakov, N. 1964. *Principles of orchestration*. Courier Corporation.
- Roads, C.; and Wieneke, P. 1979. [Grammars as Representations for Music](#). *Computer Music Journal*, 3(1):48–55.
- Roberts, A.; Engel, J.; Raffel, C.; Hawthorne, C.; and Eck, D. 2018. [A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music](#). In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4364–4373. PMLR.
- Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1986. [Learning representations by back-propagating errors](#). *Nature*, 323(6088):533–536.
- Ryu, J.; Dong, H.-W.; Jung, J.; and Jeong, D. 2024. [Nested Music Transformer: Sequentially Decoding Compound Tokens in Symbolic Music and Audio Generation](#). In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pages 588–595. ISMIR.
- Sailor, M. 2024. [RNBert: Fine-Tuning a Masked Language Model for Roman Numeral Analysis](#). In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pages 814–821. ISMIR.
- Sanh, V.; Debut, L.; Chaumond, J.; and Wolf, T. 2020. [DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter](#). *Preprint*, arXiv:1910.01108.
- Sarmiento, P.; Kumar, A.; Chen, Y.-H.; et al. 2023a. [GTR-CTRL: Instrument and Genre Conditioning for Guitar-Focused Music Generation with Transformers](#). In *Artificial Intelligence in Music, Sound, Art and Design*, pages 260–275, Cham. Springer Nature Switzerland.
- Sarmiento, P.; Kumar, A.; Xie, D.; et al. 2023b. [ShredGP: Guitarist Style-Conditioned Tablature Generation with Transformers](#). In *Proceedings of the 16th International Symposium on Computer Music Multidisciplinary Research*, pages 112–121. Zenodo.
- Sarmiento, P. P.; Kumar, A.; Carr, C.; et al. 2021. [DadaGP: A Dataset of Tokenized GuitarPro Songs for Sequence Models](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 610–617. ISMIR.
- Schachter, C. 1987. [Analysis by Key: Another Look at Modulation](#). *Music Analysis*, 6(3):289–318.
- Schaffrath, H. 1995. [The Essen Folksong Collection](#). In *Center for Computer Assisted Research in the Humanities*.
- Schoenberg, A.; Strang, G.; and Stein, L. 1999. *Fundamentals of Musical Composition*. Faber & Faber.
- Schuster, M.; and Nakajima, K. 2012. [Japanese and Korean voice search](#). In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.
- Schwab, S.; Avanzi, M.; Goldman, J.-P.; et al. 2012. [An acoustic study of penultimate accentuation in three varieties of French](#). In *Proceedings of speech prosody*, pages 266–269.
- Sears, D. R. W.; Arzt, A.; Frostel, H.; Sonnleitner, R.; and Widmer, G. 2017. [Modeling Harmony with Skip-Grams](#). In *Proceedings of the 18th International Society for Music Information Retrieval Conference*, pages 332–338. ISMIR.

- Sears, D. R. W.; and Widmer, G. 2021. [Beneath \(or beyond\) the surface: Discovering voice-leading patterns with skip-grams](#). *Journal of Mathematics and Music*, 15(3):209–234.
- Sennrich, R.; Haddow, B.; and Birch, A. 2016. [Neural Machine Translation of Rare Words with Subword Units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Sentürk, S.; and Chordia, P. 2011. [Modeling Melodic Improvisation in Turkish Folk Music Using Variable-Length Markov Models](#). In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 269–274. ISMIR.
- Sergeant, D. C.; and Himonides, E. 2016. [Gender and Music Composition: A Study of Music, and the Gendering of Meanings](#). *Frontiers in Psychology*, Volume 7 - 2016.
- Serra-Peralta, M.; Serrà, J.; and Corral, Á. 2021. [Heaps' law and vocabulary richness in the history of classical music harmony](#). *EPJ Data Science*, 10(1):40.
- Sevilla, J.; Heim, L.; Ho, A.; et al. 2022. [Compute Trends Across Three Eras of Machine Learning](#). In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.
- Shapiro, P.; and Duh, K. 2018. [BPE and CharCNNs for Translation of Morphology: A Cross-Lingual Comparison and Analysis](#). *Preprint*, arXiv:1809.01301.
- Shaw, P.; Uszkoreit, J.; and Vaswani, A. 2018. [Self-Attention with Relative Position Representations](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.
- Shen, Z.; Yang, L.; Yang, Z.; and Lin, H. 2023. [More Than Simply Masking: Exploring Pre-training Strategies for Symbolic Music Understanding](#). In *Proceedings of the 2023 ACM International Conference on Multimedia Retrieval, ICMR '23*, pages 540–544, New York, NY, USA. Association for Computing Machinery.
- Shih, Y.-J.; Wu, S.-L.; Zalkow, F.; Müller, M.; and Yang, Y.-H. 2023. [Theme Transformer: Symbolic Music Generation With Theme-Conditioned Transformer](#). *IEEE Transactions on Multimedia*, 25:3495–3508.
- Shu, Y.; Xu, H.; Zhou, Z.; van den Hengel, A.; and Liu, L. 2024. [MuseBarControl: Enhancing Fine-Grained Control in Symbolic Music Generation through Pre-Training and Counterfactual Loss](#). *Preprint*, arXiv:2407.04331.
- Skierś, P.; Łazarski, M.; Kopeć, M.; and Modrzejewski, M. 2024. [Symbotunes: unified hub for symbolic music generative models](#). In *International Society for Music Information Retrieval Conference (ISMIR), Late-Breaking Demo Session*. ISMIR.
- Spencer, P.; and Temko, P. M. 1994. [A practical approach to the study of form in music](#). Waveland Press.
- Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. [Dropout: A Simple Way to Prevent Neural Networks from Overfitting](#). *Journal of Machine Learning Research*, 15(56):1929–1958.
- Stamatatos, E.; Fakotakis, N.; and Kokkinakis, G. 1999. [Automatic Authorship Attribution](#). In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 158–164, Bergen, Norway. Association for Computational Linguistics.

- Steedman, M. J. 1984. [A Generative Grammar for Jazz Chord Sequences](#). *Music Perception*, 2(1):52–77.
- Stefani, G. 1987. [Melody: A Popular Perspective](#). *Popular Music*, 6(1):21–35.
- Sturm, B. L.; Santos, J. F.; Ben-Tal, O.; and Korshunova, I. 2016. [Music transcription modelling and composition using deep learning](#). *Preprint*, arXiv:1604.08723.
- Su, J.; Zeng, J.; Xiong, D.; et al. 2018. [A Hierarchy-to-Sequence Attentional Neural Machine Translation Model](#). *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(3):623–632.
- Su, Y.; Han, R.; Wu, X.; Zhang, Y.; and Li, Y. 2022. [Folk melody generation based on CNN-BiGRU and Self-Attention](#). In *2022 4th International Conference on Communications, Information System and Computer Engineering (CISCE)*, pages 363–368.
- Sulun, S.; Davies, M. E. P.; and Viana, P. 2022. [Symbolic Music Generation Conditioned on Continuous-Valued Emotions](#). *IEEE Access*, 10:44617–44626.
- Suzuki, M. 2022. [Score Transformer: Generating Musical Score from Note-level Representation](#). In *Proceedings of the 3rd ACM International Conference on Multimedia in Asia, MMAAsia '21*, New York, NY, USA. Association for Computing Machinery.
- Tang, J.; Wiggins, G.; and Fazekas, G. 2023. [Reconstructing Human Expressiveness in Piano Performances with a Transformer Network](#). *Preprint*, arXiv:2306.06040.
- Tao, H.; Tong, S.; Zhao, H.; et al. 2019. [A Radical-Aware Attention-Based Model for Chinese Text Classification](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):5125–5132.
- Tay, Y.; Tran, V. Q.; Ruder, S.; et al. 2022. [Charformer: Fast character transformers via gradient-based subword tokenization](#). In *International Conference on Learning Representations (ICLR)*.
- Temperley, D. 1999. [What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered](#). *Music Perception: An Interdisciplinary Journal*, 17(1):65–100.
- Temperley, D. 2004. *The cognition of basic musical structures*. MIT press.
- Tian, J.; Li, Z.; Li, J.; and Wang, P. 2024. [N-gram Unsupervised Compoundation and Feature Injection for Better Symbolic Music Understanding](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(14):15364–15372.
- Touvron, H.; Martin, L.; Stone, K.; et al. 2023. [Llama 2: Open Foundation and Fine-Tuned Chat Models](#). *Preprint*, arXiv:2307.09288.
- Trieu, N.; and Keller, R. 2018. [JazzGAN: Improvising with generative adversarial networks](#). In *6th International Workshop on Musical Metacreation (MUME)*.
- Tschuggnall, M.; Stamatatos, E.; Verhoeven, B.; et al. 2017. [Overview of the author identification task at PAN-2017: style breach detection and author clustering](#). In *Working Notes Papers of the CLEF 2017 Evaluation Labs*, pages 1–22.
- Turian, J.; Shier, J.; Khan, H. R.; et al. 2022. [Hear: Holistic evaluation of audio representations](#). In *NeurIPS 2021 Competitions and Demonstrations Track*, pages 125–145. PMLR.
- Turker, M.; Dirik, A.; and Yanardag, P. 2022. [MIDISpace: Finding Linear Directions in Latent Space for Music Generation](#). In *Proceedings of the 14th Conference on Creativity and Cognition, C&C '22*, pages 420–427, New York, NY, USA. Association for Computing Machinery.

- Uc-Cetina, V.; Navarro-Guerrero, N.; Martin-Gonzalez, A.; Weber, C.; and Wermter, S. 2023. [Survey on reinforcement learning for language processing](#). *Artificial Intelligence Review*, 56(2):1543–1575.
- Valenti, A.; Carta, A.; and Bacciu, D. 2020. [Learning Style-Aware Symbolic Music Representations by Adversarial Autoencoders](#). *Preprint*, arXiv:2001.05494.
- Van Der Merwe, A.; and Schulze, W. 2011. [Music Generation with Markov Models](#). *IEEE MultiMedia*, 18(3):78–85.
- Van Kranenburg, P.; de Bruin, M.; Grijp, L. P.; and Wiering, F. 2014. [The Meertens tune collections](#). *Meertens Online Reports*, 2014(1).
- Vaswani, A.; Shazeer, N.; Parmar, N.; et al. 2017. [Attention is All you Need](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30. Curran Associates, Inc.
- Vercoe, B. L. 2001. [Folk music classification using hidden Markov models](#). In *International Conference on Artificial Intelligence*, volume 6.
- Vila, L. C.; Sturm, B. L. T.; Casini, L.; and Dalmazzo, D. 2025. [The AI Music Arms Race: On the Detection of AI-Generated Music](#). *Transactions of the International Society for Music Information Retrieval*.
- Voita, E.; Talbot, D.; Moiseev, F.; Sennrich, R.; and Titov, I. 2019. [Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.
- von Rütte, D.; Biggio, L.; Kilcher, Y.; and Hofmann, T. 2023. [FIGARO: Controllable Music Generation using Learned and Expert Features](#). In *International Conference on Learning Representations (ICLR)*.
- Vásquez, M. A. V.; Baelemans, M.; Driedger, J.; Zuidema, W.; and Burgoyne, J. A. 2023. [Quantifying the Ease of Playing Song Chords on the Guitar](#). In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 725–732. ISMIR.
- Waite, E. 2016. [Generating Long-Term Structure in Songs and Stories](#).
- Walder, C. 2016. [Modelling Symbolic Music: Beyond the Piano Roll](#). In *Proceedings of The 8th Asian Conference on Machine Learning*, volume 63 of *Proceedings of Machine Learning Research*, pages 174–189, The University of Waikato, Hamilton, New Zealand. PMLR.
- Waller, D. 2010. [Language literacy and music literacy: A pedagogical asymmetry](#). *Philosophy of Music Education Review*, 18(1):26–44.
- Wallin, N. L.; Merker, B.; and Brown, S. 2001. *The origins of music*. MIT press.
- Wang, A.; Singh, A.; Michael, J.; et al. 2018. [GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding](#). In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.
- Wang, B.; Wang, A.; Chen, F.; Wang, Y.; and Kuo, C.-C. J. 2019. [Evaluating word embedding models: methods and experimental results](#). *APSIPA Transactions on Signal and Information Processing*, 8:19.
- Wang, L.; Zhao, Z.; Liu, H.; et al. 2023. [A Review of Intelligent Music Generation Systems](#). *Preprint*, arXiv:2211.09124.

- Wang, Y.; Wu, S.; Hu, J.; et al. 2025. [NotaGen: Advancing Musicality in Symbolic Music Generation with Large Language Model Training Paradigms](#). *Preprint*, arXiv:2502.18008.
- Wang, Z.; Chen, K.; Jiang, J.; et al. 2020a. [POP909: A pop-song dataset for music arrangement generation](#). In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 38–45, Montreal, Canada. ISMIR.
- Wang, Z.; Wang, D.; Zhang, Y.; and Xia, G. 2020b. [Learning interpretable representation for controllable polyphonic music generation](#). In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 662–669. ISMIR.
- Wang, Z.; and Xia, G. 2021. [MuseBERT: Pre-training Music Representation for Music Understanding and Controllable Generation](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 722–729. ISMIR.
- Wang, Z.; Zhang, Y.; Zhang, Y.; et al. 2020c. [PianoTree VAE: Structured representation learning for polyphonic music](#). In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 368–375, Montreal, Canada. ISMIR.
- Wankhade, M.; Rao, A. C. S.; and Kulkarni, C. 2022. [A survey on sentiment analysis methods, applications, and challenges](#). *Artificial Intelligence Review*, 55(7):5731–5780.
- Ward, J. 2019. *The student's guide to cognitive neuroscience*. Routledge.
- Wenzek, G.; Lachaux, M.-A.; Conneau, A.; et al. 2019. [CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data](#). *Preprint*, arXiv:1911.00359.
- Wiedemann, G.; Remus, S.; Chawla, A.; and Biemann, C. 2019. [Does BERT Make Any Sense? Interpretable Word Sense Disambiguation with Contextualized Embeddings](#). *Preprint*, arXiv:1909.10430.
- Wiegrefe, S.; and Pinter, Y. 2019. [Attention is not not Explanation](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.
- Wingstedt, J.; Brändström, S.; and Berg, J. 2010. [Narrative music, visuals and meaning in film](#). *Visual Communication*, 9(2):193–210.
- Wolf, T.; Debut, L.; Sanh, V.; et al. 2020. [Transformers: State-of-the-Art Natural Language Processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- Wołkowicz, J.; and Kešelj, V. 2012. [Analysis of Important Factors for Measuring Similarity of Symbolic Music Using n-gram-Based, Bag-of-Words Approach](#). In *Advances in Artificial Intelligence*, pages 230–241, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Wołkowicz, J.; Kulka, Z.; and Kešelj, V. 2008. [N-gram-based approach to composer recognition](#). *Archives of Acoustics*, 33(1):43–55.
- Wong, K.-F.; Li, W.; Xu, R.; and Zhang, Z.-s. 2022. [Introduction to Chinese natural language processing](#). Springer Nature.
- Wu, J.; Hu, C.; Wang, Y.; Hu, X.; and Zhu, J. 2020a. [A Hierarchical Recurrent Neural Network for Symbolic Melody Generation](#). *IEEE Transactions on Cybernetics*, 50(6):2749–2757.

- Wu, S.; and Sun, M. 2023. [Exploring the Efficacy of Pre-trained Checkpoints in Text-to-Music Generation Task](#). In *The AAAI-23 Workshop on Creative AI Across Modalities*.
- Wu, S.; Wang, Y.; Li, X.; Yu, F.; and Sun, M. 2024. [MelodyT5: A Unified Score-to-Score Transformer for Symbolic Music Processing](#). *Preprint*, arXiv:2407.02277.
- Wu, S.; Wang, Y.; Yuan, R.; et al. 2025. [CLaMP 2: Multimodal Music Information Retrieval Across 101 Languages Using Large Language Models](#). In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 435–451, Albuquerque, New Mexico. Association for Computational Linguistics.
- Wu, S.; Yu, D.; Tan, X.; and Sun, M. 2023. [CLaMP: Contrastive Language-Music Pre-Training for Cross-Modal Symbolic Music Information Retrieval](#). In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 157–165. ISMIR.
- Wu, S.-L.; and Yang, Y.-H. 2020. [The jazz transformer on the front line: Exploring the shortcomings of AI-composed music through quantitative measures](#). In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 142–149. ISMIR.
- Wu, S.-L.; and Yang, Y.-H. 2023a. [Compose & Embellish: Well-Structured Piano Performance Generation via A Two-Stage Approach](#). In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.
- Wu, S.-L.; and Yang, Y.-H. 2023b. [MuseMorphose: Full-Song and Fine-Grained Piano Music Style Transfer With One Transformer VAE](#). *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1953–1967.
- Wu, X.; Wang, C.; and Lei, Q. 2020b. [Transformer-XL Based Music Generation with Multiple Sequences of Time-valued Notes](#). *Preprint*, arXiv:2007.07244.
- Xu, W.; McAuley, J.; Dubnov, S.; and Dong, H.-W. 2023. [Equipping Pretrained Unconditional Music Transformers with Instrument and Genre Controls](#). In *2023 IEEE International Conference on Big Data (BigData)*, pages 4512–4517.
- Xue, N. 2003. [Chinese Word Segmentation as Character Tagging](#). In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*, pages 29–48.
- Yang, L.-C.; and Lerch, A. 2020. [On the evaluation of generative models in music](#). *Neural Computing and Applications*, 32(9):4773–4784.
- Yang, R.; Wang, D.; Wang, Z.; et al. 2019a. [Deep Music Analogy Via Latent Representation Disentanglement](#). *Preprint*, arXiv:1906.03626.
- Yang, Z.; Dai, Z.; Yang, Y.; et al. 2019b. [XLNet: Generalized Autoregressive Pretraining for Language Understanding](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, Vancouver, Canada. Curran Associates, Inc.
- Yannakakis, G. N.; and Martínez, H. P. 2015. [Ratings are Overrated!](#) *Frontiers in ICT*, 2.
- Yeh, Y.-C.; Hsiao, W.-Y.; Fukayama, S.; et al. 2021. [Automatic melody harmonization with triad chords: A comparative study](#). *Journal of New Music Research*, 50(1):37–51.
- Young, G.; and Roens, S. 2022. [Form, texture, style, and harmonic language](#). In *Insights into music composition*, pages 46–55. Routledge.

- Yu, B.; Lu, P.; Wang, R.; et al. 2022. [Museformer: Transformer with fine-and coarse-grained attention for music generation](#). *Advances in Neural Information Processing Systems (NeurIPS)*, 35:1376–1388.
- Yu, Y.; Srivastava, A.; and Canales, S. 2021. [Conditional LSTM-GAN for Melody Generation from Lyrics](#). *ACM Transactions on Multimedia Computing, Communications, and Applications*, 17(1).
- Yuan, R.; Lin, H.; Wang, Y.; et al. 2024. [ChatMusician: Understanding and Generating Music Intrinsically with LLM](#). *Preprint*, arXiv:2402.16153.
- Zbikowski, L. M. 2009. [Music, language, and multimodal metaphor](#). *Multimodal metaphor*, pages 359–381.
- Zeiler, M. D.; and Fergus, R. 2014. [Visualizing and Understanding Convolutional Networks](#). In *Computer Vision – ECCV 2014*, pages 818–833, Cham. Springer International Publishing.
- Zeng, M.; Tan, X.; Wang, R.; et al. 2021. [MusicBERT: Symbolic Music Understanding with Large-Scale Pre-Training](#). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 791–800, Online. Association for Computational Linguistics.
- Zhang, H.; Karystinaios, E.; Dixon, S.; Widmer, G.; and Cancino-Chacón, C. E. 2023. [Symbolic Music Representations for Classification Tasks: A Systematic Evaluation](#). In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 848–858. ISMIR.
- Zhang, H.; Tang, J.; Rafee, S. R.; et al. 2022. [ATEPP: A Dataset of Automatically Transcribed Expressive Piano Performance](#). In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 446–453. ISMIR.
- Zhang, L.; and Callison-Burch, C. 2023. [Language Models are Drummers: Drum Composition with Natural Language Pre-Training](#). In *The AAAI-23 Workshop on Creative AI Across Modalities*.
- Zhang, L.; and Jiang, F. 2021. [Visualizing Symbolic Music via Textualization: An Empirical Study on Chinese Traditional Folk Music](#). In *Mobile Multimedia Communications*, pages 647–662, Cham. Springer International Publishing.
- Zhang, N. 2020. [Learning Adversarial Transformer for Symbolic Music Generation](#). *IEEE Transactions on Neural Networks and Learning Systems*, 34(4):1754–1763.
- Zhang, Y.; Wang, Z.; Wang, D.; and Xia, G. 2020. [BUTTER: A Representation Learning Framework for Bi-directional Music-Sentence Retrieval and Generation](#). In *Proceedings of the 1st Workshop on NLP for Music and Audio (NLP4MusA)*, pages 54–58, Online. Association for Computational Linguistics.
- Zhao, H.; Chen, H.; Yang, F.; et al. 2024a. [Explainability for Large Language Models: A Survey](#). *ACM Transactions on Intelligent Systems and Technology*, 15(2).
- Zhao, J.; Taniar, D.; Adhinugraha, K.; Baskaran, V. M.; and Wong, K. 2023a. [Multi-mmlg: a novel framework of extracting multiple main melodies from MIDI files](#). *Neural Computing and Applications*, 35(30):22687–22704.
- Zhao, J.; Wong, K.; Baskaran, V. M.; Adhinugraha, K.; and Taniar, D. 2023b. [Computational Music: Analysis of Music Forms](#). In *Computational Science and Its Applications – ICCSA 2023: 23rd International Conference, Athens, Greece, July 3–6, 2023, Proceedings, Part I*, pages 366–384, Berlin, Heidelberg. Springer-Verlag.
- Zhao, J.; and Xia, G. 2021. [AccoMontage: Accompaniment Arrangement via Phrase Selection and Style Transfer](#). In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 833–840. ISMIR.

- Zhao, J.; Xia, G.; and Wang, Y. 2023c. [Q&A: Query-Based Representation Learning for Multi-Track Symbolic Music re-Arrangement](#). In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI-23)*.
- Zhao, J.; Xia, G.; Wang, Z.; and Wang, Y. 2024b. [Structured Multi-Track Accompaniment Arrangement via Style Prior Modelling](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Zhao, Z. 2024. [Adversarial-MidiBERT: Symbolic Music Understanding Model Based on Unbias Pre-training and Mask Fine-tuning](#). *Preprint*, arXiv:2407.08306.
- Zhou, J.; Zhu, H.; and Wang, X. 2023. [Choir Transformer: Generating Polyphonic Music with Relative Attention on Transformer](#). *Preprint*, arXiv:2308.02531.
- Zhou, Z.; Wu, Y.; Wu, Z.; et al. 2024. [Can LLMs "Reason" in Music? an Evaluation of LLMs' Capability of Music Understanding and Generation](#). In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pages 103–110. ISMIR.
- Zhu, H.; Liu, Q.; Yuan, N. J.; et al. 2018. [XiaoIce Band: A Melody and Arrangement Generation Framework for Pop Music](#). In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18*, pages 2837–2846, New York, NY, USA. Association for Computing Machinery.
- Zhu, X.; Li, J.; Liu, Y.; Ma, C.; and Wang, W. 2023a. [A Survey on Model Compression for Large Language Models](#). *Preprint*, arXiv:2308.07633.
- Zhu, Y.; Baca, J.; Rekabdar, B.; and Rawassizadeh, R. 2023b. [A Survey of AI Music Generation Tools and Models](#). *Preprint*, arXiv:2308.12982.
- Zixun, G.; Makris, D.; and Herremans, D. 2021. [Hierarchical Recurrent Neural Networks for Conditional Melody Generation with Long-term Structure](#). In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Appendix A

NLP tools for MIR: representations

The following tables illustrate [Chapter 4: Representations of symbolic music as sequences](#).

- [Table A.1](#) gives an overview and descriptions of event-based tokenization based on **elementary tokens** ([Section 4.1.2.1](#)).
- [Table A.2](#) gives an overview and descriptions of event-based tokenization based on **composite tokens** ([Section 4.1.2.2](#)).

A companion Github repository with latest representations is available at <https://github.com/dinhviettoanle/survey-music-nlp>.

Table A.1: Overview of event-based tokenization strategies based on *elementary* tokens. The “alphabet” describes the types of atomic elements constituting the alphabet with their type. The “data” corresponds to the type of music considered by the indicated article. It also specifies whether the tokenization is score- or performance-based.

Tokenization	Alphabet (<i>Atomic elements</i>)	Grouping	Vocab. size	Data
ABC notation (Sturm et al., 2016)	Text alphabet	Bar patching (Wu et al., 2023)	N/A	Monophonic (Score)
SMT-ABC (<i>MuPT</i>) (Qu et al., 2025)	Text alphabet	BPE (Qu et al., 2025)	N/A	Multi-track (Score)
Interleaved ABC notation (Wang et al., 2025)	Text alphabet	Bar patching (Wang et al., 2025)	N/A	Multi-track (Score)
MEI-based tags (Acosta et al., 2022)	Text alphabet	–	769	Monophonic (Score)
Park et al. (2024) (<i>Mel2Word</i>)	<Pitch-interval> (<i>integer</i>) <Time-shift> (<i>music time</i>)	BPE (Park et al., 2024)	30	Monophonic (Score)
MIDI-like (Oore et al., 2018)	<Note-ON> (<i>MIDI value</i>) <Note-OFF> (<i>MIDI value</i>) <Time-shift> (<i>absolute time</i>) <Velocity> (<i>integer</i>)	BPE (Kumar and Sarmento, 2023) (Zhang et al., 2023) Unigram (Kumar and Sarmento, 2023)	388	Piano (Performance)
LakhNES (Donahue et al., 2019)	<NoteON-[trk]> (<i>MIDI value</i>) <NoteOFF-[trk]> (<i>MIDI value</i>) <Time-shift> (<i>absolute time</i>)	–	630	Multi-track (Performance)
REMI (Huang and Yang, 2020)	<Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Velocity> (<i>integer</i>) <Chord> (<i>class</i>) <Bar> <Position> (<i>music time</i>)	BPE (Fradet et al., 2023a) (Kumar and Sarmento, 2023) (Zhang et al., 2023) Unigram (Kumar and Sarmento, 2023)	332	Piano (Score)
REMI+ (von Rütte et al., 2023)	REMI alphabet + features: <Instrument> (<i>class</i>) <Time-Signature> (<i>class</i>) <Tempo> (<i>integer</i>)	–	N/A	Multi-track (Score)

Table A.1: (Continued) Overview of event-based tokenization strategies based on *elementary* tokens.

Tokenization	Alphabet (<i>Atomic elements</i>)	Grouping	Vocab. size	Data
Lee et al. (2022) (ComMU)	REMI alphabet + metadata: <Instrument> (<i>class</i>) <Key> (<i>class</i>) <Time-Signature> (<i>class</i>) <BPM> (<i>integer</i>) <Min/Max-velocity> (<i>integer</i>) <Nb-of-bars> (<i>number</i>) <Pitch-range> (<i>class</i>) <Rhythm> (<i>class</i>)	–	728	Multi-track (Score)
Gover and Zewi (2022) (MusIAC)	REMI alphabet + control info: <Occupation> (<i>class</i>) <Density> (<i>class</i>) <Tensile-train> (<i>class</i>) <Cloud diameter> (<i>class</i>) <Polyphony> (<i>class</i>)	–	360	Multi-track (Score)
Gover and Zewi (2022)	<Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Bar> <Position> (<i>music time</i>) <Hand> (<i>class</i>)	–	N/A	Piano (Score)
Wu and Yang (2023b) (MuseMorphose)	<Durat.-[trk]> (<i>music time</i>) <Pitch-[trk]> (<i>MIDI value</i>) <Bar> <Velocity-[trk]> (<i>integer</i>) <Position> (<i>music time</i>) <Tempo> (<i>integer</i>)	–	3440	Multi-track (Score)
MultiTrack (Ens and Pasquier, 2020)	<Start-piece> <Start-track> / <End-track> <Instrument> (<i>class</i>) <Start-bar> / <End-bar> <Start-fill> / <End-fill> <Density-level> (<i>integer</i>) <Note-ON/OFF> (<i>MIDI value</i>) <Time-shift> (<i>absolute time</i>)	–	440	Multi-track (Performance)

Table A.1: (Continued) Overview of event-based tokenization strategies based on *elementary* tokens.

Tokenization	Alphabet (<i>Atomic elements</i>)	Grouping	Vocab. size	Data
MMR (<i>SymphonyNet</i>) (Liu et al., 2022)	<Start-score> / <End-score> <Start-bar> / <End-bar> <Change-track> <Position> (<i>integer</i>) <Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Chord> (<i>class</i>)	BPE (Liu et al., 2022)	N/A	Multi-track (Score)
TSD (Fradet et al., 2023a)	<Pitch> (<i>MIDI value</i>) <Velocity> (<i>integer</i>) <Rest> (<i>absolute time</i>) <Duration> (<i>absolute time</i>) <Time-shift> (<i>absolute time</i>) <Program> (<i>class</i>)	BPE (Fradet et al., 2023a)	249	Multi-track (Performance)
Structured (Hadjeres and Crestel, 2021)	<Pitch> (<i>MIDI value</i>) <Velocity> (<i>integer</i>) <Duration> (<i>absolute time</i>) <Time-shift> (<i>absolute time</i>)	–	428	Piano (Performance)
PerTok (Lenz and Mani, 2024)	<Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Velocity> (<i>integer</i>) <Bar> <Time-shift> (<i>absolute time</i>) <Micro-Shift> (<i>absolute time</i>)	–	196	Piano (Performance)
Li et al. (2023c)	<Pitch-class> (<i>class</i>) <Octave> (<i>integer</i>) <Duration> (<i>music time</i>) <Bar> (<i>integer</i>) <Position> (<i>music time</i>) <Velocity> (<i>integer</i>)	–	N/A	Monophonic (Score)

Table A.1: (Continued) Overview of event-based tokenization strategies based on *elementary* tokens.

Tokenization	Alphabet (<i>Atomic elements</i>)	Grouping	Vocab. size	Data
Chen et al. (2020)	<Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Velocity> (<i>integer</i>) <Bar> (<i>integer</i>) <Position> (<i>music time</i>) <Grooving> (<i>class</i>) <String> (<i>integer</i>) <Fret> (<i>integer</i>) <Technique> (<i>class</i>)	–	231	Guitar (Tablatures)
Sarmiento et al. (2021) (<i>DadaGP</i>)	<start>/<end> <Wait> (<i>integer</i>) <Instr:note> (<i>MIDI value</i>) <Effect> (<i>class</i>) <Drums:note> (<i>MIDI value</i>) <String> (<i>integer</i>) <Fret> (<i>integer</i>)	BPE (Kumar and Sarmiento, 2023) Unigram (Kumar and Sarmiento, 2023)	2140	Guitar (Tablatures)
Suzuki (2022) (<i>Score Transformer</i>)	<Staff> <Barline> <Clef> (<i>class</i>) <Key-signature> (<i>class</i>) <Time-signature> (<i>class</i>) <Voice> <Rest> <Pitch> (<i>class</i>) <Duration> (<i>class</i>) <Stem-direction> (<i>class</i>) <Beams> (<i>class</i>) <Tie> (<i>class</i>)	–	N/A	Piano (Score)

Table A.2: Overview of event-based tokenization strategies based on *composite* tokens. The “musical features” column describes the components of the vectors considered as tokens, in terms of musical attribute. The “embedded object” denotes the manner these musical features are grouped together to form the super-token, including fixed-size vectors or based on event families.

Tokenization	Musical features	Super-token nature	Data
Luo et al. (2020) (MG-VAE)	<Pitch> (<i>class</i>) <Interval> (<i>number</i>) <Rhythm> (<i>class</i>)	Homogeneous	Monophonic
Zhang (2020)	<Pitch> (<i>integer</i>) <Velocity> (<i>integer</i>) <Program> (<i>class</i>)	Homogeneous	Multi-track
PiRhDy (Liang et al., 2020a)	<Chroma> (<i>class</i>) <Octave> (<i>integer</i>) <Velocity> (<i>integer</i>) <Inter-onset-interval> (<i>music time</i>) <Note-state> (<i>class</i>)	Homogeneous	Multi-track
Zixun et al. (2021)	<Pitch> (<i>one-hot</i>) <Duration> (<i>one-hot</i>) <Bar> (<i>one-hot</i>) <Current-chord> (<i>one-hot</i>) <Next-chord> (<i>one-hot</i>)	Homogeneous	Lead sheet
Octuple (Zeng et al., 2021)	<Time-signature> (<i>class</i>) <Tempo> (<i>integer</i>) <Bar> (<i>integer</i>) <Position> (<i>music time</i>) <Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Velocity> (<i>integer</i>) <Instrument> (<i>class</i>)	Homogeneous	Multi-track
Dong et al. (2023) (MMT)	<Type> (<i>class</i>) <Beat> (<i>integer</i>) <Position> (<i>music time</i>) <Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Instrument> (<i>class</i>)	Homogeneous	Multi-track

Table A.2: (Continued) Overview of event-based tokenization strategies based on *composite* tokens.

Tokenization	Musical features	Super-token nature	Data
Dalmazzo et al. (2024) (Chordinator)	<Chord-root> (class) <Chord-nature> (class) <Chord-extensions> (class) <Slash-chord> (boolean) <MIDI-array> (multi-hot)	Homogeneous	Chord sequences
Wang and Xia (2021) (MuseBERT)	<Onset> (music time) <Pitch> (MIDI value) <Duration> (music time) + factorized properties	Homogeneous	Multi-track
MuMIDI (Ren et al., 2020)	<Bar> <Position> (music time) <Tempo> (integer) <Track> (class) <Chord> (class) <Pitch/Drum> (MIDI value) <Velocity> (integer) <Duration> (music time)	Family-based	Multi-track
Compound Word (Hsiao et al., 2021)	<Family> (class) <Time-signature> (class) <Bar> (integer) <Beat> (music time) <Chord> (class) <Tempo> (integer) <Pitch> (MIDI value) <Duration> (music time) <Velocity> (integer)	Family-based	Piano
Di et al. (2021)	<Type> (class) <Beat> (integer) <Strenth> (class) <Pitch> (MIDI value) <Duration> (music time) <Instrument> (integer)	Family-based	Multi-track

Table A.2: (Continued) Overview of event-based tokenization strategies based on *composite* tokens.

Tokenization	Musical features	Super-token nature	Data
Makris et al. (2022)	Encoder input: <Onset> (<i>number</i>) <Duration> (<i>music time or none</i>) <Group> (<i>class</i>) <Type> (<i>class</i>) <Value> (<i>any - depends on type</i>) Decoder output: <Onset> (<i>number</i>) <Drums> (<i>integer</i>)	Family-based	Enc.: Multi-track Dec.: Drums
Unsupervised Compound Word (Tian et al., 2024)	<Family> (<i>class</i>) <Time-signature> (<i>class</i>) <Bar> (<i>integer</i>) <Beat> (<i>music time</i>) <Chord> (<i>class</i>) <Tempo> (<i>integer</i>) <Pitch> (<i>MIDI value</i>) <Duration> (<i>music time</i>) <Velocity> (<i>integer</i>)	Family-based + learning	Piano
REMI_Track (Luo et al., 2024)	<Instrument> (<i>class</i>) <Position> (<i>music time</i>) <Bar> Learned grouping: <BPE> and/or <Pitch> (<i>MIDI value</i>) and/or <Velocity> (<i>integer</i>) and/or <Duration> (<i>music time</i>)	Heterogeneous + learning	Multi-track

Appendix B

NLP tools for MIR: models

The following tables illustrate [Chapter 5: NLP-based models for symbolic music processing](#).

- [Table B.1](#) gives an overview and descriptions of **recurrent models** for symbolic music processing ([Section 5.2.2](#)).
- [Table B.2](#) gives an overview and descriptions of Transformer-based models trained as **end-to-end** models for symbolic music processing ([Section 5.3](#)).
- [Table B.3](#) gives an overview and descriptions of Transformer-based models trained as **pre-trained** and **fine-tuned** models for symbolic music processing ([Section 5.3](#)).

A companion Github repository with latest models is available at <https://github.com/dinhviettoanle/survey-music-nlp>.

Table B.1: Recurrent models applied to symbolic music. Models are grouped based on their recurrent unit type (RNN, LSTM, GRU). Precisions indicated in the *Representation* column depict the specific adaptations brought to an initial tokenization strategy. The last column indicates if the code has been publicly released.

Model	Architecture	Data	Representation	Tasks	Code
Recurrent Neural Network (RNN)					
RNN-RBM (Boulanger-Lewandowski et al., 2012)	RBM + RNN	Multi-track	Time-slice (piano roll)	Free generation	✗
RNN-DBN (Goel et al., 2014)	RBM + DBN + RNN	Multi-track	Time-slice (piano roll)	Free generation	✗
Long-Short Term Memory (LSTM)					
Folk-RNN (Sturm et al., 2016)	LSTM	Monophonic	ABC notation	Free generation	✓
C-RNN-GAN (Mogren, 2016)	GAN + Bi-LSTM	Multi-track	Pitch + duration + time-shift + velocity (composite tokens)	Free generation	✓
Song from Pi (Chu et al., 2016)	Hierarchical + LSTM	Multi-track	Custom features (composite tokens)	Free generation (melody, chord, drum generation)	✗
Melody/Attention-RNN (Waite, 2016)	LSTM (+ Attention)	Monophonic	Note-ON / Note-OFF	Priming	✓
DeepBach (Hadjeres et al., 2017)	Bi-LSTM	4-part chorales	Time-slice-based	Harmonization Free generation	✓
Anticipation-RNN (Hadjeres and Nielsen, 2017)	LSTM	Monophonic	Pitch + duration (time-slice-based)	Infilling	✓
JamBot (Brunner et al., 2017)	LSTM	Multi-track	Time-slice (piano roll)	Chord generation Chord-conditioned gen.	✓

Table B.1: (Continued) Recurrent models applied to symbolic music.

Model	Architecture	Data	Representation	Tasks	Code
Note-RNN / RL Tuner (Jaques et al., 2017)	LSTM (+ RL)	Monophonic	Note-ON / Note-OFF	Free generation	✓
PerformanceRNN (Oore et al., 2018)	LSTM	Piano	MIDI-like	Expressive perform. gen.	✓
Chen and Su (2018)	Bi-LSTM	Piano	Time-slice (piano roll)	Roman Numeral Analysis	✓
StructureNet (Medeot et al., 2018)	LSTM	Monophonic	Custom features (composite tokens)	Free generation	✗
Music-VAE (Roberts et al., 2018)	VAE + LSTM	Monophonic	MIDI-like	Samples interpolation Free generation	✓
JazzGAN (Trieu and Keller, 2018)	GAN + LSTM	Lead sheet	Pitch + duration + chord (event-based)	Chord-conditioned gen.	✗
DeepJ (Mao et al., 2018)	Biaxial LSTM	Piano	Time-slice (piano roll)	Free generation Style embedding analysis	✓
Chen et al. (2019)	Bi-LSTM	Lead sheet	Time-slice (piano roll)	Chord-conditioned gen.	✗
Makris et al. (2019)	LSTM (Drums) Feed-forward (Context)	Multi-track	Drums: (event-based) Context: (time-slice)	Drums accompaniment generation	✗
MahlerNet (Lousseief and Sturm, 2019)	VAE + Bi-LSTM	Multi-track	Event-based	Samples interpolation	✓
GrooVAE (Gillick et al., 2019)	VAE + Bi-LSTM	Drums	Time-slice (drumroll)	Drum Infilling Tap2Drum Humanization	✓

Table B.1: (Continued) Recurrent models applied to symbolic music.

Model	Architecture	Data	Representation	Tasks	Code
Wu et al. (2020a)	Hierarchical + LSTM	Monophonic	Note-ON / Note-OFF	Structure-conditioned gen.	✗
VirtuosoNet (Jeong et al., 2019a)	Hierarchical + VAE + Bi-LSTM + Attention	Piano	Custom features (composite tokens)	Expressive perform. gen.	✓
Amadeus (Kumar and Ravindran, 2019)	LSTM + RL	Piano	Pitch + duration (event-based)	Free generation	✗
MuseAE (Valenti et al., 2020)	Adversarial Autoencoder + LSTM	Multi-track	Time-slice (piano roll)	Samples interpolation Embedding analysis	✓
Jin et al. (2020)	LSTM + RL	Multi-track	Time-slice (piano roll)	Free generation	✗
GGA-MG (Farzaneh and Toroghi, 2020)	Bi-LSTM + GA	Monophonic	ABC Notation	Free generation	✓
Yu et al. (2021)	GAN + LSTM	Monophonic	Pitch + duration (event-based)	Lyrics-conditioned gen.	✓
CM-HRNN (Zixun et al., 2021)	Hierarchical + LSTM	Lead sheet	Pitch + duration + chord + bar (composite tokens)	Chord-conditioned gen.	✓
Keerti et al. (2022)	Bi-LSTM + Attention	Monophonic	Pitch + duration (event-based)	Sequence reconstruction	✗
LStoM (Kosta et al., 2022)	Bi-LSTM	Piano	Custom features (event-based)	Melody extraction	✓
Turker et al. (2022)	VAE + LSTM	Piano	Note-ON / Note-OFF	Sequence reconstruction Latent space analysis	✗
Gated Recurrent Unit (GRU)					
MIDI-VAE (Brunner et al., 2018)	VAE + GRU	Multi-track	Time-slice (piano roll)	Style transfer Samples interpolation	✓

Table B.1: (Continued) Recurrent models applied to symbolic music.

Model	Architecture	Data	Representation	Tasks	Code
XiaoIce Band (Zhu et al., 2018)	GRU + Attention	Multi-track	Pitch + duration + chord (event-based)	Chord-conditioned gen. Arrangement generation	✗
Songwriter (Bao et al., 2019)	GRU + Attention	Monophonic	Pitch + duration (event-based)	Lyrics-conditioned gen.	✗
Yang et al. (2019a)	VAE + bi-GRU	Lead sheet	Time-slice (piano roll) + chords (chromagram)	Melody contour / chord-conditioned gen.	✓
BUTTER (Zhang et al., 2020)	VAE + GRU	Monophonic	Time-slice (piano roll)	Text-based query Music captioning Text-conditioned gen.	✓
Kong et al. (2020)	Bi-GRU	Piano	Time-slice (piano roll)	Composer classification	✓
MG-VAE (Luo et al., 2020)	VAE + GRU	Monophonic	Pitch + interval + duration (event-based)	Free generation	✗
PianoTree-VAE (Wang et al., 2020c)	VAE + bi-GRU	Piano / Multi-track	Time-slice (piano roll) MIDI-like	Samples interpolation Free generation Embedding analysis	✓
Su et al. (2022)	Bi-GRU + CNN + Attention	Monophonic	Pitch + duration (time-slice-based)	Free generation	✗

Table B.2: *End-to-end* Transformer-based models applied to symbolic music: such models are directly trained on specific tasks. Models are grouped by architecture. Details indicated in the *Representation* column depict the specific adaptations brought to an initial tokenization strategy. The last column indicates the code availability.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
Encoder-only architecture					
MTBert (Zhao et al., 2023b)	BERT (no pre-training)	4-part chorales	Interval + duration (event-based)	Fugue form analysis	✗
Decoder-only architecture					
Music Transformer (Huang et al., 2019)	Tf. decoder (+ <i>Relative attention</i>)	Piano / Choral	MIDI-like	Priming Harmonization	✓
Chen et al. (2020)	Transformer-XL	Guitar tabs	REMI-derived (Tablatures)	Free tabs gen.	✗
Pop Music Transformer (Huang and Yang, 2020)	Transformer-XL	Piano	REMI	Free generation Priming	✓
Jazz Transformer (Wu and Yang, 2020)	Transformer-XL	Lead sheet	REMI-derived (Chords)	Free generation	✓
PopMAG (Ren et al., 2020)	Transformer-XL	Multi-track	MuMIDI	Accompaniment gen.	✗
Wu et al. (2020b)	Transformer-XL	Piano	MIDI-like-derived (composite tokens)	Free generation	✗
Di et al. (2021)	Tf. decoder	Multi-track	CPWord-derived (Rhythm family)	Video-to-music	✓
Chang et al. (2021)	XLNet (+ <i>Relative bar enc.</i>)	Piano	Compound Word	Infilling	✓
Compound Word Tf. (Hsiao et al., 2021)	Linear Tf. decoder	Piano	Compound Word	Free generation Priming	✓

Table B.2: (Continued) *End-to-end* Transformer-based models applied to symbolic music.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
Sarmento et al. (2021)	Transformer-XL	Guitar tabs + multi-track	DadaGP	Metadata-conditioned gen.	✓
Sulun et al. (2022)	Music Transformer	Multi-track	MIDI-like	Emotion-conditioned gen.	✓
ComMU (Lee et al., 2022)	Transformer-XL	Multi-track	REMI + metadata	Metadata-conditioned gen. Multi-track combination	✓
SymphonyNet (Liu et al., 2022)	Linear Tf. (+ <i>3-D positional encoding</i>)	Orchestral	MMR	Free generation / Priming Chord-conditioned gen.	✓
Li et al. (2023c)	Transformer-XL	Lead sheet	REMI-derived (pitch class)	Free generation	✗
Multitrack Music Tf. (Dong et al., 2023)	Tf. decoder	Orchestral	MMT	Free generation / Priming Instr.-conditioned gen.	✓
GTR-CTRL (Sarmento et al., 2023a)	Transformer-XL	Guitar tabs + multi-track	DadaGP	Instr.-conditioned gen. Genre-conditioned gen.	✗
ShredGP (Sarmento et al., 2023b)	Transformer-XL	Guitar tabs	DadaGP	Style-conditioned gen.	✗
Choir Transformer (Zhou et al., 2023)	Tf. decoder (+ <i>Relative attention</i>)	4-part chorales	Chord + pitch (event-based)	Harmonization	✓
Guo et al. (2023)	Tf. encoder (+ <i>Fundamental music embd.</i> + <i>RIPO attention</i>)	Monophonic	FME	Priming	✓
Compose & Embellish (Wu and Yang, 2023a)	Tf. decoder	Piano	REMI	Lead sheet priming Accomp refinement	✓
RHEPP-Transformer (Tang et al., 2023)	Tf. decoder	Piano	Octuple	Expressive performance gen.	✓
Angioni et al. (2023)	Tf. encoder	Multi-track	TSD-like	Style classification	✓

Table B.2: (Continued) *End-to-end* Transformer-based models applied to symbolic music.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
Chordinator (Dalmazzo et al., 2024)	minGPT (no pre-training)	Chords	Custom chord features (+ MIDI array)	Chord generation	✓
Nested Music Tf. (Ryu et al., 2024)	Tf. decoder (+ <i>Auto-reg. CPWord decoding</i>)	Multi-track	Compound Word	Free generation	✓
Encoder-decoder architecture					
Transformer-VAE (Jiang et al., 2020b)	Tf. encoder-decoder	Monophonic	Pitch + duration (time-slice-based)	Priming	✗
Harmony Transformer (Chen and Su, 2021)	Tf. encoder-decoder	Piano	Piano roll time-slices	Roman Numeral Analysis	✓
Makris et al. (2021)	Tf. encoder-decoder	Lead sheet	Enc.: bar features Dec.: chord + pitch + dur.	Emotion-conditioned gen.	✓
Liutkus et al. (2021)	Performer (+ <i>Stochastic positional enc.</i>)	Multi-track	REMI / MIDI-like-derived (multi-track)	Free generation Groove continuation	✓
Gover and Zewi (2022)	BART	Piano	REMI-derived (hands tokens)	Arrangement generation	✗
Museformer (Yu et al., 2022)	Tf. encoder-decoder (+ <i>Fine-/coarse-grained attn.</i> + <i>Bar selection</i>)	Multi-track	REMI	Free generation	✓
Theme Transformer (Shih et al., 2023)	Tf. encoder-decoder (+ <i>Theme-aligned pos. enc.</i>)	Multi-track	REMI-derived (theme tokens)	Theme-conditioned gen.	✓
FIGARO (von Rütte et al., 2023)	Tf. encoder-decoder	Multi-track	REMI+	Controllable generation	✓
MuseMorphose (Wu and Yang, 2023b)	Tf. enc + Transformer-XL (+ <i>In-attention conditioning</i>)	Piano	REMI-derived (multi-track)	Style transfer Controllable generation	✓

Table B.2: (Continued) *End-to-end* Transformer-based models applied to symbolic music.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
Zhao et al. (2024b)	Tf. encoder-decoder (+ <i>Instrument embedding</i>)	Multi-track	Piano roll time-slices	Accompaniment gen.	✓
TeleMelody (Ju et al., 2022)	Tf. encoder-decoder	Monophonic	Bar + position + pitch + duration	Lyrics-to-melody	✓
MuseCoco (Lu et al., 2023)	Text2Attr.: BERT Attr2Music: Linear Tf.	Multi-track	REMI	Text-to-MIDI	✓
Multi-view MidiVAE (Lin et al., 2024a)	Tf. encoder-decoder	Multi-track	Octuple	Free generation	✗
MelodyT5 (Wu et al., 2024)	T5	Monophonic	ABC notation	Melody generation Melody harmonization Melody segmentation	✓
Composer’s Assistant 2 (Malandro, 2024)	T5	Multi-track	REMI+-derived (text format)	Infilling Controllable generation	✗
BandControlNet (Luo et al., 2024)	Tf. encoder-decoder (+ <i>Structure enhanced self-attention</i>)	Multi-track	REMI_Track	Controllable generation	✓
Text2midi (Bhandari et al., 2025)	FlanT5 + Tf. decoder	Multi-track	REMI+	Text-to-MIDI	✓
Model combinations					
Zhang (2020)	Generator: Tf. decoder Discriminator: Tf. encoder	Multi-track	MIDI-like-derived (composite tokens)	Free generation	✗
Transformer-GAN (Muhammed et al., 2021)	Generator: Tf.-XL Discriminator: BERT	Piano	MIDI-like	Free generation	✓
Dai et al. (2021)	Encoder: Tf. encoder Decoder: LSTM	Piano	Pitch + rhythm (event-based)	Structure-conditioned gen. Chord conditioned gen.	✗

Table B.2: (Continued) *End-to-end* Transformer-based models applied to symbolic music.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
Choi et al. (2021)	Chord enc.: Bi-LSTM Rhythm dec.: Tf. decoder Pitch dec.: Tf. decoder	Lead sheet	Pitch + rhythm + chord (time-slice-based)	Chord-conditioned gen.	✓
Bar Transformer (Qin et al., 2022)	Bi-LSTM - Tf. decoder	Lead sheet	Bar + position + melody + chord (time-slice-based)	Free generation	✗
Makris et al. (2022)	Bi-LSTM - Tf. decoder	Multi-track	CPWord-derived	Drums accomp. generation	✓
Neves et al. (2022)	Generator: Linear Tf. Discriminator: Linear Tf. (+ <i>Local prediction map</i>)	Piano	REMI	Emotion-conditioned gen.	✓
Q&A (Zhao et al., 2023c)	PianoTree-VAE Tf. decoder (+ <i>Instrument embedding</i>)	Multi-track	Piano roll time-slices	Accompaniment gen.	✓
Duan et al. (2023)	Generator: Tf. encoder Discriminator: LSTM	Monophonic	Pitch + duration + rest (event-based)	Lyrics-to-melody	✗
Video2Music (Kang et al., 2023)	GRU + Tf. encoder-decoder	Multi-track	MIDI-like	Video-to-music	✓

Table B.3: *Pre-trained* Transformer-based models applied to symbolic music: such models are pre-trained and then fine-tuned on downstream tasks.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
Encoder-only architecture					
MuseBERT (Wang and Xia, 2021)	BERT (+ <i>Generalized relative pos. enc.</i>)	Piano	MuseBERT repr.	Controllable generation Chord analysis Accomp. refinement	✓
MidiBERT-Piano (Chou et al., 2024)	BERT	Piano	REMI Compound Word	Melody extraction Velocity prediction Composer classification Emotion classification	✓
MusicBERT (Zeng et al., 2021)	RoBERTa (+ <i>Bar-level masking</i>)	Multi-track*	Octuple	Melody completion Accompiment suggest. Genre classification Style classification	✓
DBTMPE (Qiu et al., 2021)	Tf. encoder	Multi-track	Pitch combinations + duration (event-based)	Style classification	✗
MRBERT (Li and Sung, 2023b)	BERT (+ <i>Melody/rhythm cross attention</i>)	Lead sheet	Pitch + duration (event-based)	Free generation Infilling Chord analysis	✗
SoloGPBERT (Sarmiento et al., 2023b)	BERT	Guitar tabs	DadaGP	Guitar player classif.	✗
Shen et al. (2023)	MidiBERT-Piano (+ <i>Quad-attribute masking</i> + <i>Key prediction pre-training tasks</i>)	Multi-track	CPWord simplified	Melody extraction Velocity prediction Composer classification Emotion classification	✗

Table B.3: (Continued) *Pre-trained* Transformer-based models applied to symbolic music.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
CLaMP (Wu et al., 2023)	Text enc.: DistilRoBERTa Music enc.: BERT	Lead sheet	ABC notation-derived	Text-based semantic search Music recommendation Genre classification Emotion classification Composer classification	✓
Adversarial MidiBERT (Zhao, 2024)	BERT (+ <i>Adversarial pre-training</i>)	Piano	Octuple	Melody extraction Velocity prediction Composer classification Emotion classification	✓
Decoder-only architecture					
LakhNES (Donahue et al., 2019)	Transformer-XL	Multi-track	MIDI-like	Free generation	✓
Musenet (Payne, 2019)	GPT-2 (+ <i>Timing embedding</i> + <i>Structural embedding</i>)	Multi-track*	MIDI-like	Priming	✗
MMM (Ens and Pasquier, 2020)	GPT-2	Multi-track	MultiTrack repr.	Free generation Inpainting Priming Controllable generation	✓
Angioni et al. (2023)	GPT-2	Multi-track	TSD-like	Priming	✓
Zhang and Callison-Burch (2023)	GPT-3	Drums	Drumroll time-slices	Priming	✓
ChatMusician (Yuan et al., 2024)	Llama-2	Monophonic	ABC Notation	Text-to-ABC	✓
ComposerX (Deng et al., 2024a)	GPT-4	Monophonic	ABC notation	Text-to-ABC	✓

Table B.3: (Continued) *Pre-trained* Transformer-based models applied to symbolic music.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
MuseBarControl (Shu et al., 2024)	Linear Tf. (+ <i>Auxiliary task pre-adaptation</i>)	Piano	REMI	Controllable generation Chord-conditioned gen.	✗
MIDI-GPT (Pasquier et al., 2025)	GPT-2	Multi-Track	Multi-Track repr.	Free generation Infilling Priming Controllable generation	✓
NotaGen (Wang et al., 2025)	GPT-2 (+ <i>CLaMP-DPO</i>)	Multi-track	ABC notation	Free generation Style-conditioned gen.	✓
Encoder-decoder architecture					
MusIAC (Guo et al., 2022)	Tf. encoder-decoder	Multi-track	REMI	Infilling Controllable generation	✓
Li and Sung (2023c)	Tf. encoder-decoder	Lead sheet	Pitch + duration (event-based)	Harmony analysis Chord generation	✗
Fu et al. (2023)	MusicBERT + Music Tf.	Multi-track	Octuple	Melody completion Accompaniment suggest. Melody extraction Emotion classification	✗
Multi-MMLG (Zhao et al., 2023a)	XLNet + MuseBERT	Multi-track	CPWord-derived	Melody extraction	✗
PianoBART (Liang et al., 2024)	BART (+ <i>Multi-level object masking</i>)	Piano	Octuple	Priming Velocity prediction Melody extraction Composer classification Emotion classification	✓

Table B.3: (Continued) *Pre-trained* Transformer-based models applied to symbolic music.

Model	Base model (+ <i>MIR mechanism</i>)	Data	Representation	Tasks	Code
Comparative studies					
Ferreira et al. (2023)	GRU, Performance-RNN GPT-2, Music Tf., MuseNet (Tf. decoders)	Piano	MIDI-like	Free generation	✓
Wu and Sun (2023)	BERT (Tf. encoder) GPT-2 (Tf. decoder) BART (Tf. enc.-dec.)	Lead sheet	ABC notation	Text-to-ABC	✓

Tf.: *Transformer* | Enc.: *Encoder* | Dec.: *Decoder* | Pos. enc.: *Positional Encoding* | (*) These datasets are not publicly available.

Appendix C

Layer-wise relevance propagation for multi-hot time-slice representation

In [Section 7.4.3](#), we aim at applying Layer-wise Relevance Propagation (LRP) to a masked language model in order to evaluate if harmonic analysis characteristics are encoded through attention head relevance. Though, the input format of the model relies on a time-slice representation, through multi-hot vectors. Mathematically, the application of LRP is not direct because it must be back-propagated from a *single* logit value from the output.

In the following, we will consider this set of three multi-hot vectors outputs as a single multi-hot vector, for simplicity. To this end, instead of back-propagating LRP from each single value of this multi-hot vector, we consider summing the values of this multi-hot vector from which LRP is back-propagated. We show that these two processes are equivalent (*i.e.* we want that the relevance of each single value of the multi-hot vector corresponds to its logit). In practice, such LRP back-propagation from S allows for a single LRP back-propagation instead of performing n independent back-propagations.

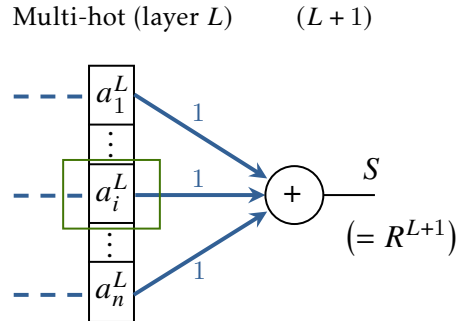


Figure C.1: LRP for multi-hot representation. The back-propagation is performed from the sum of all the values from the final multi-hot output vector, and is equivalent to back-propagate from each single value of the multi-hot vector.

Proof. Following [Figure C.1](#), we consider a model with L layers, with an output layer being a multi-hot vector $[a_1^L, \dots, a_n^L] \in \mathbb{R}^n$. We define S as the sum of these values:

$$S = \sum_{j=1}^n a_j^L \quad (\text{C.1})$$

From a model perspective, this sum can be considered as an $(L+1)^{\text{th}}$ layer with only one output and sums its inputs. In terms of weights, this means:

$$\forall k \in \{1, \dots, n\}, \omega_k^{(L,L+1)} = 1 \quad (\text{C.2})$$

Considering S as being the value from which LRP is back-propagated means:

$$R^{L+1} = S \quad (\text{C.3})$$

We want to derive the relevance R_i^L of the i^{th} element of the multi-hot vector. Following [Equation \(7.1\)](#), it is defined by:

$$\begin{aligned} R_i^L &= \sum_k \frac{a_i^L \omega_{ik}^{(L,L+1)}}{\sum_j a_j^L \omega_{jk}^{(L,L+1)}} R_k^{L+1} \\ &= \frac{a_i^L \omega_i^{(L,L+1)}}{\sum_j a_j^L \omega_j^{(L,L+1)}} R^{L+1} &> \text{the } L+1 \text{ layer has only one output.} \\ &= \frac{a_i^L \omega_i^{(L,L+1)}}{\sum_j a_j^L \omega_j^{(L,L+1)}} S &> \text{using Equation (C.3).} \\ &= \frac{a_i^L}{\sum_j a_j^L} S &> \text{using Equation (C.2).} \\ R_i^L &= a_i^L &> \text{using Equation (C.1).} \end{aligned}$$

Back-propagating LRP from S is therefore equivalent to back-propagating from each single element of the multi-hot vector.

□

Appendix D

Résumé étendu en français

Le domaine du Traitement automatique du langage naturel (TALN) a connu des avancées majeures durant ces dernières années, notamment grâce à la popularité des applications basées sur des grands modèles de langage. Cela se traduit notamment par des applications grand public tels que ChatGPT. Motivé par ces avancées et par les nombreux liens existants en musique et langage naturel, le domaine de l'extraction automatique d'informatique musicale, ou *Music Information Retrieval (MIR)*, a peu à peu consacré une part croissante de ses sujets de recherche sur l'utilisation de ces modèles. Ils sont notamment utilisés en analyse ou génération automatique de musique, aussi bien dans le domaine audio que dans le domaine symbolique.

Dans cette thèse, nous nous intéressons au contenu musical représenté sous forme symbolique, par exemple, sous forme de partition. Dans une certaine mesure, celle-ci est comparable au texte, mais garde également ses spécificités. Cela soulève alors la question de l'*adaptation* de ces outils de TALN pour traiter des données musicales. Dans cette thèse, nous détaillons les aspects dans lesquels les méthodes de TALN peuvent être appliquées en musique: tâches, représentations séquentielles et modèles, et nous présentons alors des contributions techniques dans chacun de ces trois aspects. Cette thèse soutient néanmoins que, malgré les similitudes entre musique et langage, les outils du TALN doivent avant tout servir d'*inspiration* pour la recherche en musique symbolique. Celle-ci doit être guidée en priorité par des problématiques musicales, plutôt que par une application directe ou systématique des méthodes issues du TALN.

Chapitre 2 – Des parallèles entre musique et langage naturel

Musique et langage naturel sont souvent considérés comme des moyens de communication, mais avec différentes visées. Le langage a pour but de transmettre des idées et des concepts, alors que la musique est plus souvent associée à l'affect et aux émotions. Alors que le langage peut également transmettre des émotions, la

question du *sens* de la musique demeure un sujet de débat récurrent et de longue date. Leur déclinaison sous une forme auditive et écrite peuvent également engendrer des réactions cognitives similaires, notamment en termes d'attente grammaticale ou sémantique dans le langage, ou d'attente liée à l'harmonie en musique.

En considérant musique et texte comme des données, ceux-ci partagent également des spécificités. Ils sont notamment tous les deux basés sur des représentations hiérarchiques, avec plusieurs niveaux de segmentation (par exemple, lettres, mots, paragraphes en texte ; note, motif, phrase musicale en musique). En revanche, la dimension temporelle de la musique, la polyphonie et le polymorphisme d'un symbole musical demeurent spécifiques à ce domaine et ne trouvent pas d'équivalent direct en texte. Cette hiérarchie se déploie également à haut niveau, avec notamment des règles grammaticales ou des structures textuelles, de la même manière que la musique tonale est régie par les règles d'harmonie et des formes musicales, tels que la fugue ou la forme sonate.

Partie I : Une vue d'ensemble des méthodes de traitement du langage naturel en informatique musicale

La première contribution de cette thèse est une présentation organisée des méthodes de TALN pour le traitement de représentations symboliques de la musique. Celle-ci est organisée autour de trois axes : *tâches*, *représentations séquentielles* et *modèles*.

Chapitre 3 – Tâches en TALN et MIR : similarités et spécificités

Les similarités entre ces deux domaines se retrouvent dans leurs tâches. En suivant les paradigmes d'apprentissage sur des données qui sont aujourd'hui les plus répandues, ces tâches peuvent s'organiser autour de données *annotées* (*i.e.* où chaque partition ou texte d'un corpus est annoté par des labels), ou *non-annotées* (*i.e.* des partitions ou des textes bruts).

Les tâches s'appuyant sur des **données annotées** comprennent notamment des tâches opérant sur des *séquences entières* de musique ou de texte. On peut notamment aisément tracer des parallèles entre des tâches de classification d'auteur ou de compositeur, ou de reconnaissance de sentiment ou d'émotion. En revanche, les tâches qui opèrent au niveau du jeton (*token*) – par exemple un mot ou une note – font ressortir les différences entre texte et musique. Par exemple, une tâche d'analyse harmonique ne trouve pas d'équivalent en texte, au même titre que la labellisation des classes grammaticale de mot en texte.

Des tâches utilisant des corpus de **données non-annotées** comprennent notamment des tâches de *clustering* ou de *segmentation* de phrases textuelles / musicales.

Aujourd’hui, les tâches les plus dominantes en TALN et MIR sont des tâches portant sur la *génération automatique* de contenu textuel ou musical. La génération de musique ou de texte peuvent se rapporter à des contextes similaires, tels que la génération à partir d’une instruction générative (*prompt*) ou le transfert de style. Des tâches restent particulières à chacun des domaines tels que la génération d’accompagnement en musique – qui suppose l’existence de la notion d’harmonie et/ou de rythme – ou le résumé de texte – qui présume de la notion de sémantique.

Chapitre 4 – Représentations séquentielles de la musique et du texte

Alors que le texte peut se représenter de manière séquentielle de manière relativement naturelle, une telle représentation séquentielle pour la musique est moins directe. Par exemple, la notion de polyphonie, induite par la simultanéité de certaines notes, viendrait directement à l’encontre de l’unidimensionnalité d’une séquence. Pourtant, de multiples **tokenizations** musicales – terme emprunté au TALN pour signifier la représentation séquentielle d’un contenu musical ou textuel – ont été proposées.

L’échelle temporelle étant une caractéristique fondamentale en musique, les *tokenizations* basées sur des **tranches de temps** segmentent la séquence musicale en tranches de longueurs temporelles égales. Celles-ci permettent notamment de généraliser les *piano rolls* – une représentation matricielle de la musique – sous une forme séquentielle.

Ce sont, en revanche, les *tokenizations* basées sur les **événements** qui se sont largement répandues. Cela est notamment dû au nombre important de fichiers MIDI disponibles aujourd’hui. Ces événements sont définis par le choix d’un *alphabet* et d’une possible *stratégie de groupement* statistique ou par apprentissage. Le choix de l’alphabet détermine les éléments de la partition à représenter ainsi que leur forme de représentation. Par exemple, la tokenization basée sur le protocole MIDI définit une note comme étant une suite d’événements de notes appuyées puis relâchées, et le temps est représenté via des écarts temporels relatifs entre ces événements. Au contraire, REMI encode le temps de façon absolue de manière similaire aux partitions, comme étant un temps musical au sein d’une mesure.

Le choix d’événements pour encoder la musique introduit nécessairement une *séquentialité artificielle*. En effet, une note, caractérisée par plusieurs uniques propriétés, doit être encodée comme une suite d’événements, et la polyphonie doit être approximée sous la forme d’une séquence unidimensionnelle. Pour palier à la séquentialité au sein d’une même note, des *tokens composites* permettent de regrouper toutes les caractéristiques d’une même note au sein d’un même *token*.

Ces *tokens* peuvent alors être plongés dans un espace, via des *embeddings* qui peuvent être construits ou appris par un modèle, et utilisés comme entrée d’un modèle traitant des données représentées sous forme de séquence de *tokens*.

Chapitre 5 – Modèles adaptés du TALN pour la musique symbolique

Avant l’essor de l’apprentissage profond, des modèles dérivés du TALN ont déjà été utilisés pour traiter des informations musicales symboliques tels que les grammaires génératives, les chaînes de Markov cachées ou les modèles récurrents. En revanche, les performances du modèle Transformeur dans des tâches de TALN ont par la suite grandement motivé l’adaptation de ce modèle en MIR.

Les modèles utilisés en MIR basés sur le Transformeur peuvent être vus sous plusieurs prismes, notamment leur paradigme d’entraînement, leur architecture ou bien les adaptations de ses mécanismes internes pour la musique.

Le **paradigme d’entraînement** – de bout-en-bout (*end-to-end*) ou via un pré-entraînement suivi d’un réglage fin (*fine-tuning*), tels que BERT ou GPT– peut donner lieu un modèle spécifique à une tâche particulière, ou bien un modèle « comprenant » la musique pouvant être ensuite spécialisé sur une tâche.

L’**architecture** du modèle comprend notamment des *encodeurs*, *décodeurs*, ou bien une association *encodeur-décodeur*. L’architecture basée sur uniquement des encodeurs est généralement privilégiée pour des tâches d’analyse. Les deux autres architectures permettent de réaliser des tâches allant de l’analyse harmonique à des tâches de génération automatique de musique. Enfin, les *modèles multimodaux* permettent d’intégrer d’autres types de données, tels que du texte ou de la vidéo, et les mettre en relation avec la musique. Ce cadre permet également d’utiliser la puissance des grands modèles de langage textuels (*Large Language Model*) pour traiter des données musicales symboliques encodées sous forme de texte, via notamment la Notation ABC.

Pour répondre aux particularités de la musique, plusieurs ajustements des **mécanismes internes** du Transformeur ont été proposés. Par exemple, la couche d’encodage de la position (*positional encoding*) a notamment été étudiée de sorte à prendre en compte la polyphonie ou les relations entre les attributs musicaux. Le *mécanisme d’attention* a également pu être adapté, par exemple en prenant en compte la répétitivité en musique. Enfin, dans le cadre des modèles pré-entraînés, différentes tâches de pré-entraînement spécifiquement musicales ont été utilisées tels que l’analyse harmonique.

Partie II: Contributions techniques

La seconde partie de cette thèse expose les contributions techniques, en suivant cette organisation en trois axes.

Chapitre 6 – Expressivité des représentations séquentielles de la musique

Dans ce chapitre, nous nous intéressons à l’impact de l’expressivité d’une *tokenization* basée sur des événements dans les performances de modèles d’analyse automatique. Tout d’abord, nous étudions l’expressivité de l’alphabet, avec le choix de *tokens* encodant les intervalles musicaux au lieu des hauteurs absolues des notes. Nous étudions ensuite l’expressivité d’un mécanisme de groupement, *Byte-Pair Encoding* (BPE), appliqué à différents niveaux de polyphonie.

Tokenization basée sur les intervalles musicaux – On mémorise généralement une mélodie à travers son contour mélodique plutôt qu’à partir de ses hauteurs de note exactes. Pourtant, les *tokenizations* basées sur les événements utilisent le plus souvent des *tokens* <Pitch> indiquant la hauteur absolue de la note. Nous avons alors exploré l’impact de l’utilisation d’une tokenization basée sur des *tokens* d’intervalles. Nous proposons notamment une méthode formelle pour construire de telles tokenizations, en se basant sur le choix de l’encodage des intervalles entre notes successives ou simultanées, et le choix d’un sous-ensemble de notes de référence.

Nos résultats montrent qu’une telle tokenization peut améliorer les performances de modèles sur diverses tâches, et améliorer l’explicabilité du modèle, bien que les choix permettant de construire la tokenization jouent un rôle important. En dépit de ces résultats, nous avons restreint notre étude à des modèles légers pour isoler l’effet de la *tokenization*. Des modèles plus profonds pourraient toutefois être capables d’apprendre eux-mêmes la notion d’intervalle.

Analyse de BPE pour la musique monophonique et polyphonique – *Byte-Pair Encoding* (BPE) est un algorithme permettant de regrouper statistiquement les paires de tokens les plus récurrents. Il est notamment utilisé en TALN pour construire des sous-mots (*subwords*) en enrichissant le vocabulaire initial de *super-tokens*. Notre étude se focalise plus spécifiquement sur l’analyse des *super-tokens* musicaux et l’effet de BPE sur une tâche de segmentation en phrase musicale.

Nous observons alors que les *super-tokens* musicaux appris par BPE ont des caractéristiques différentes par rapport au langage naturel, et diffèrent selon le niveau de polyphonie considéré. Par ailleurs, ceux-ci peuvent être interprétés musicalement, et ont une place particulière vis-à-vis des positions de fin de phrase musicale. Des résultats quantitatifs sur cette tâche de détection de fin de phrase musicale montrent alors que le nombre de regroupement via BPE impacte la performance d’un modèle, selon le niveau de polyphonicité des données. Cet hyper-paramètre sensible et le temps alloué à l’entraînement d’un *tokenizer* via BPE peut potentiellement dissuader son utilisation en pratique, au profit d’une *tokenization* plus commune.

Chapitre 7 – Une exploration des mécanismes de modèles basés sur l’attention

Bien que les modèles de TALN appliqués en MIR peuvent aboutir à des performances satisfaisantes sur certaines tâches, leur fonctionnement interne reste souvent inexploré. Nous nous focalisons alors *l’explicabilité d’un modèle* entraîné sur une tâche d’analyse harmonique fonctionnelle, à travers l’étude de son mécanisme d’attention.

Analyse harmonique fonctionnelle – L’analyse harmonique fonctionnelle est une tâche pouvant être modélisée comme une tâche de classification de *token*. Ces labels sont alors des annotations d’harmonie, tels que la tonalité, le degré de l’accord ou encore son renversement. Nous proposons alors un modèle Transformeur dont l’architecture est basée sur des encodeurs, prenant en entrée une *tokenization* en tranches de temps, et entraîné pour réaliser cette tâche de multi-classification. Ce modèle est entraîné de bout-en-bout et obtient des performances proches de l’état de l’art.

Analyse du mécanisme d’attention – Nous nous intéressons plus particulièrement au comportement du mécanisme d’attention inhérent au modèle Transformeur que nous avons proposé. Pour cela, une première analyse se focalise sur les coefficients d’attention bruts. Nous modélisons la portée de l’attention comme étant le nombre de tokens précédents ou suivants auxquels le modèle « porte attention ». En analysant cette quantité, le modèle accorde plus d’attention au futur qu’au passé.

Nous étudions ensuite la contribution des têtes d’attention dans la sous-tâche de détection de tonalité en utilisant un outil nommé *Layer-wise Relevance Propagation*. En TALN, cet outil a notamment servi à améliorer l’explicabilité des modèles textuels, en attribuant une importance à chaque token dans le résultat généré pour une tâche donnée. Nous observons alors que, pour un modèle entraîné uniquement sur cette sous-tâche, des têtes d’attentions différentes sont spécialisées sur différentes tonalités. Par ailleurs, il apparaît qu’en dépit de l’entraînement sur une sous-tâche de détection de tonalité, le comportement des têtes d’attention est également corrélé avec plus ou moins d’importance à d’autres éléments d’analyse harmonique.

Enfin, l’analyse des contributions des têtes d’attention dans un modèle uniquement pré-entraîné ne révèle pas de présence claire d’informations liées à l’analyse harmonique. Autrement dit, contrairement à la grammaire qui apparaît dans des modèles pré-entraînés en TALN, le concept d’harmonie fonctionnelle semble être trop abstrait pour être appris de façon implicite à travers une tâche de pré-entraînement générique, et n’est alors pas capturé par notre protocole.

Chapitre 8 – Un cas d’adaptation des méthodes de TALN pour une tâche de génération de musique

Enfin, nous proposons une *tâche* de ré-orchestration. Nous développons pour cela un modèle dérivé du TALN entraîné sur cette tâche de génération automatique de musique symbolique.

Ré-orchestration – Nous formalisons la tâche de ré-orchestration comme étant une tâche de transfert de style instrumental et textural conservant la mélodie. En d’autres termes, nous nous concentrons sur une tâche visant à transformer une pièce musicale de référence en changeant son instrumentation et sa texture. Aujourd’hui, les modèles de génération automatique de musique symbolique réalisent cette tâche en considérant le plus souvent des ensembles d’instruments fixes, et outrepassent la fidélité mélodique entre la pièce d’origine et ré-orchestrée.

METEOR – Nous proposons alors METEOR, un auto-encoder variationnel basé sur un modèle Transformeur entraîné sur cette tâche de ré-orchestration. Celui-ci permet une contrôlabilité texturale et instrumentale, et assurant une fidélité mélodique. Son architecture est basée sur *MuseMorphose*, originellement entraîné sur une tâche de transfert de style pour le piano. METEOR réalise une contrôlabilité de texture au niveau des mesures via un conditionnement de son espace latent. Le modèle réalise les contrôlabilités texturales et instrumentales et assure une fidélité mélodique via des *tokens* de contrainte. Au-delà de la ré-orchestration, ce modèle peut être converti en un modèle d’orchestration à partir de *lead sheet* (mélodie et accords) sans entraînement supplémentaire.

Nous évaluons alors le modèle via des métriques quantitatives, montrant que notre modèle atteint des performances de l’état-de-l’art sur la tâche de ré-orchestration. Nous montrons également que la fidélité mélodique peut nuire à la contrôlabilité et que le choix de l’instrumentation peut impacter les performances. Une étude utilisateur est ensuite réalisée, montrant également que METEOR est préféré par rapport à deux autres modèles sur la tâche de ré-orchestration, et atteint des résultats similaires à l’état de l’art sur la tâche d’orchestration de *lead sheet*. À l’instar d’autres modèles de génération, METEOR présente toutefois des limites en ce qui concerne la jouabilité de la musique produite, qui n’est pas toujours garantie.

Chapitre 9 – Discussions, perspectives et conclusions

En dépit des performances atteintes empiriquement par les méthodes de TALN traitant des données musicales symboliques, leur application *directe* soulève des interrogations. Par exemple, la quantité de données textuelles est bien supérieure à celle des données musicales symboliques, alors que les modèles Transformeurs sont conçus pour être entraînés sur des corpus de grande taille. En revanche, les pratiques dans le domaine du TALN peuvent inspirer de nouvelles recherches en

MIR, notamment par le développement de modèles plus légers ou explicables, ou la mise en place de *benchmarks* pour l'évaluation et la comparaison de modèles.

Ainsi, nous proposons dans cette thèse une *structuration d'une piste de recherche en MIR* basée sur trois éléments: la *tâche* visée, la *représentation* de la musique utilisée, et le *modèle* implémenté. L'identification et la justification de ces trois niveaux sont essentielles pour évaluer la pertinence d'une contribution en MIR. Aussi, de la même manière que les modèles à bruit statistique (*diffusion models*) – initialement développés pour des images – se développent de plus en plus en musique symbolique, les outils de TALN doivent avant être considérés une *boîte à outils* dont le MIR peut s'inspirer. Autrement dit, la recherche en MIR devrait avant tout être motivée par des problématiques *musicales*, plus que par l'application directe, systématique ou forcée d'outils issus de domaines mis en parallèle avec la musique.

