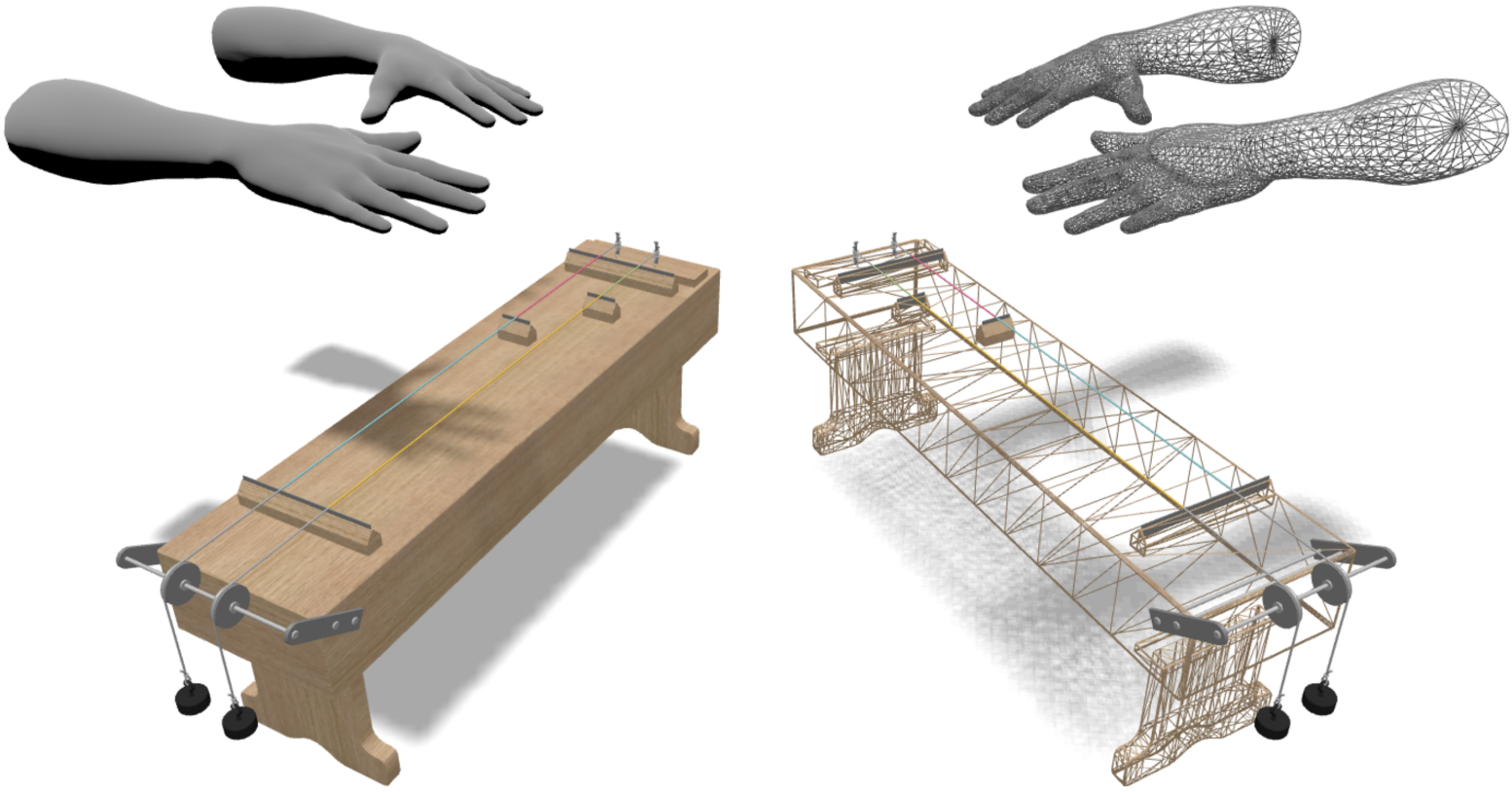
Computational models of multimodal interaction for music generation and music information retrieval

Kosmas Kritsis



Πανεπιστήμιο Πειραιώς
University of Piraeus

Υπολογιστικά μοντέλα πολυτροπικής διαδραστικότητας για μουσική δημιουργία και ανάκτηση μουσικής πληροφορίας

Κοσμάς Κρίτσης
Πατρώνυμο: Ευάγγελος
Αριθμός Μητρώου: ΠΛΔ1702

Διδακτορική διατριβή

Τμήμα Πληροφορικής
Σχολή Τεχνολογιών Πληροφορικής και Επικοινωνιών
Πανεπιστήμιο Πειραιώς
Ελλάδα, 2023

Computational models of multimodal interaction for music generation and music information retrieval

Kosmas Kritsis

A dissertation presented for the degree of
Doctor of Philosophy

Department of Informatics
School of Information and Communication Technologies
University of Piraeus
Greece, 2023

Υπολογιστικά μοντέλα πολυτροπικής διαδραστικότητας για μουσική δημιουργία και ανάκτηση μουσικής πληροφορίας

Η διατριβή εκπονήθηκε για την απονομή

Διδακτορικού Διπλώματος

από το Τμήμα Πληροφορικής
της Σχολής Τεχνολογιών Πληροφορικής και Επικοινωνιών
του Πανεπιστημίου Πειραιώς
στόν

Κοσμά Κρίτση

Τριμελής Συμβουλευτική Επιτροπή

Άγγελος Πικράκης	Βασίλειος Κατσούρος	Βασίλειος-Αιμίλιος Καμπουρόπουλος
Πανεπιστήμιο Πειραιώς *(Επιβλέπων)*	Ερευνητικό Κέντρο Αθηνά	Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Εγκρίθηκε την
14 Δεκεμβρίου 2023

_____	_____	_____
Άγγελος Πικράκης	**Βασίλειος Κατσούρος**	**Βασίλειος-Αιμίλιος Καμπουρόπουλος**
Επίκουρος Καθηγητής Πανεπιστήμιο Πειραιώς	Ερευνητής Α΄ Ερευνητικό Κέντρο Αθηνά	Καθηγητής Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
_____	_____	_____
Δημήτριος Αποστόλου	**Θεμιστοκλής Παναγιωτόπουλος**	**Διονύσιος Σωτηρόπουλος**
Καθηγητής Πανεπιστήμιο Πειραιώς	Καθηγητής Πανεπιστήμιο Πειραιώς	Επίκουρος Καθηγητής Πανεπιστήμιο Πειραιώς

_____

Μάξιμος Καλιακάτσος-Παπακώστας
Αναπληρωτής Καθηγητής
Ελληνικό Μεσογειακό Πανεπιστήμιο

Computational models of multimodal interaction for music generation and music information retrieval

A dissertation presented for the degree of

Doctor of Philosophy

in the Department of Informatics
of the School of Information and Communication Technologies
at the University of Piraeus
By

Kosmas Kritsis

Supervising Committee

Angelos Pikrakis	Vasileios Katsouros	Vasileios-Aimilios Kampouropoulos
University of Piraeus *(Supervisor)*	ATHENA Research & Innovation Center	Aristotle University of Thessaloniki

Approved by
14th December 2023

Angelos Pikrakis	**Vasileios Katsouros**	**Vasileios-Aimilios Kampouropoulos**
Assistant Professor University of Piraeus	Research Director ATHENA Research & Innovation Center	Professor Aristotle University of Thessaloniki
Dimitrios Apostolou	**Themistoklis Panagiotopoulos**	**Dionysios Sotiropoulos**
Professor University of Piraeus	Professor University of Piraeus	Assistant Professor University of Piraeus

Maximos Kaliakatsos-Papakostas
Associate Professor
Hellenic Mediterranean University

". . . A purely materialistic art would be like a tree which is expected to bear fruit without flowering, and to sacrifice grace and beauty for mere utility. Those who learn here should from the beginning, assiduously avoid this spirit of utilitarianism. Our admiration for the Creator's handiwork should not be limited to those things He has provided us with for our daily needs, but should include all that is good and beautiful. It is these tender feelings of deep and silent admiration evoked from our hearts by the beauties of creation that should find adequate expression in the fine arts. . . ."

Haile Selassie I

Acknowledgements

I would like to express my sincere gratitude and appreciation to all those who have contributed to the completion of this PhD thesis. First and foremost, I am immensely grateful to my advisors, Prof. Aggelos Pikrakis and Dr. Vassilis Katsouros, for their invaluable guidance, unwavering support, and expertise throughout my research journey. Their encouragement, patience, and mentorship have been instrumental in shaping the direction of my work and pushing me to reach this point. I am truly fortunate to have had the opportunity to work under their supervision.

I would like to extend my heartfelt thanks to the members of my thesis committee, Emilios Cambouropoulos, Dimitris Apostolou, Themis Panayiotopoulos, Dionisios Sotiropoulos and Maximos Kaliakatsos-Papakostas, for their valuable insights, constructive feedback, and critical evaluation of my research. Their expertise and contributions have significantly strengthened the quality and rigor of this thesis.

My sincere appreciation also goes to the staff and faculty of Athena R.C. for providing a stimulating scientific environment and the necessary resources for carrying out my research. I am grateful for the opportunities I have had to engage in various fruitful projects and collaborations with fellow researchers, which have greatly enriched my analytical thinking and learning experience. I am indebted to my colleagues and labmates who have supported me in countless ways, whether through their technical expertise, intellectual discussions, or moral support. This work would not have been possible without all of you. I feel very lucky to have found so many friends who believed in me more than I did, who understood my passion and became prouder of my work than myself.

I would like to express my deepest appreciation to my family and friends for their unwavering love, encouragement, and understanding throughout this demanding endeavor. Their belief in my abilities has been a con-

stant source of motivation, and their presence in my life has given me the strength to overcome moral challenges.

I am also grateful to the participants who volunteered their time and contributed to the data collection process, as well as their willingness to participate and share their insights in the subjective evaluations, both being invaluable to the success of this research.

Lastly, I would like to acknowledge the financial support provided by the European Union's Horizon 2020 research and innovation programme under the Grant agreement No. 731861, as well as Greece and the European Union (European Social Fund—ESF) through the Operational Program "Human Resources Development, Education and Lifelong Learning 2014-2020" in the context of the project "Analysis and Processing of Motion and Sound Data for Real-Time Music Creation" under Grant MIS 5047232. Without their generous funding, this research would not have been possible. I am truly grateful for their investment in my academic pursuits.

Περίληψη

Η έρευνα στον τομέα της μουσικής αλληλεπίδρασης απαιτεί τη χρήση διεπιστημονικών μέσων κατανόησης βασισμένων σε δυναμικές αντιλήψεις. Με αυτήν την έννοια, οι κύριες προσεγγίσεις, οι εμπειρικές μελέτες, οι υποκειμενικές αξιολογήσεις και οι τεχνικές μοντελοποίησης που αναπτύσσονται σε αυτήν τη διδακτορική διατριβή ακολουθούν αυτές τις διεπιστημονικές αρχές. Τα συστήματά μας λαμβάνουν και αναλύουν διάφορες μορφές και επίπεδα πληροφορίας που σχετίζονται με τη μουσική, συμπεριλαμβανομένου του ακουστικού σήματος, των αισθητηριακών και σκελετικών δεδομένων, καθώς και διαφορετικών τύπων συμβολικών αναπαραστάσεων. Επομένως, σε αυτή τη διδακτορική διατριβή παρουσιάζουμε μια εκτενή εξερεύνηση των πεδίων της υπολογιστικής μουσικής, καθώς και της αυτόματης αναγνώρισης και σύνθεσης κίνησης, περιλαμβάνοντας μια σειρά από μεθοδολογίες, μοντέλα και εφαρμογές. Η έρευνα αποσκοπεί στη βελτίωση της κατανόησης αυτών των πεδίων και στην ανάπτυξη νέων προσεγγίσεων για την αντιμετώπιση των διάφορων προκλήσεων.

Μέσα από τις πολλαπλές πειραματικές προσεγγίσεις που αναπτύσσουμε, αναδύονται πολύτιμα συμπεράσματα για τις δυνατότητες και τις επιπτώσεις των διαφορετικών υπολογιστικών αρχιτεκτονικών, ιδιαίτερα αυτών που βασίζονται σε αναδρομικές και συνελικτικές συναρτήσεις. Συγκεκριμένα, η έρευνά μας ξεκινά με την αξιολόγηση υπολογιστικών μοντέλων για την αναγνώριση μουσικών κινήσεων, αναδεικνύοντας την ανωτερότητα των συνελικτικών μοντέλων, όπως οι βαθιές συνελικτικές αρχιτεκτονικές, όσον αφορά την ακρίβεια αναγνώρισης και τον χρόνο υπολογισμού. Βασιζόμενοι σε αυτά τα ευρήματα, αναπτύσσουμε μία διαδικτυακή εφαρμογή με στόχο τη μουσική αλληλεπίδραση σε πραγματικό χρόνο με εικονικά μουσικά όργανα, συνδυάζοντας τόσο συνελικτικές όσο και αναδρομικές αρχιτεκτονικές με σκοπό τη βελτίωση της εμπειρίας χρήστη. Επίσης, εξερευνούμε την αυτόματη σύνθεση χορευτικών κινήσεων με βάση το ακουστικό

σήμα, όπου βαθιές συνελικτικές αρχιτεκτονικές που ενσωματώνουν έναν εξαρτώμενο αυτοκωδικοποιητή, υπερτερούν σε σχέση με τα αναδρομικά μοντέλα στη δημιουργία ποικίλων και ρεαλιστικών ακολουθιών χορευτικών κινήσεων. Στη συνέχεια, επικεντρωνόμαστε στην προσομοίωση της αλληλεπίδρασης μεταξύ του ανθρώπινου σολίστα και του αυτόματου συνοδού, στο πλαίσιο μοντελοποίησης του τζαζ αυτοσχεδιασμού, αναδεικνύοντας τις προκλήσεις και τις προοπτικές των εγγενών προσεγγίσεων μηχανικής μάθησης στη μοντελοποίηση μουσικών αλληλεπιδράσεων. Τέλος, διερευνούμε την επίδραση των συμβολικών κωδικοποιήσεων στην αυτόματη παραγωγή μουσικής, τονίζοντας τη σημασία των μουσικών χαρακτηριστικών που πρέπει να αποτυπώνονται κατά τη σχεδίαση νέων προσεγγίσεων κωδικοποίησης, με σκοπό τη βελτιστοποίηση της δομής της παραγόμενης μουσικής.

Συνολικά, η έρευνά μας παρέχει πολύτιμες πληροφορίες για την απόδοση και τις δυνατότητες διαφορετικών υπολογιστικών αρχιτεκτονικών στην υπολογιστική μουσική παραγωγή και αλληλεπίδραση. Η επιτυχής ενσωμάτωση συνελικτικών και αναδρομικών μοντέλων καταδεικνύει την ικανότητά τους να μοντελοποιούν περίπλοκες μουσικές αλληλεπιδράσεις. Τονίζουμε τη σημασία της επιλογής της κατάλληλης υπολογιστικής αρχιτεκτονικής με βάση τους υποκειμενικούς στόχους, της συνθήκες και τους περιορισμούς που ορίζει το κάθε πρόβλημα προς διερεύνηση. Τα ευρήματά μας θέτουν τα θεμέλια για περαιτέρω έρευνα, ενθαρρύνοντας την εξερεύνηση προηγμένων αρχιτεκτονικών, μεγαλύτερων συνόλων δεδομένων, καθώς και την εφαρμογή τους σε επιπλέων ερευνητικά προβλήματα υπολογιστικής μουσικής παραγωγής και αλληλεπίδρασης, που δεν καλύφθηκαν στην παρούσα διατριβή. Με αυτόν τον τρόπο, προωθούμε νέες δυνατότητες δημιουργικής έκφρασης, συνεργασίας ανθρώπου-μηχανής και την πρόοδο της μουσικής τεχνολογίας στο σύνολό της.

Abstract

In this dissertation, we explore multiple aspects of computational music generation and interaction, addressing tasks such as musical gesture recognition, virtual instrument interaction, audio-driven dance motion synthesis, jazz improvisation accompaniment generation, and symbolic music encodings. Throughout our various studies described here, we gain valuable insights into the capabilities and implications of different computational architectures, particularly recurrent and convolutional models. Our research begins by evaluating computational models for musical gesture recognition, with subsequent experimentation revealing the superiority of convolutional models, such as deep convolutional architectures, in terms of recognition accuracy and computation time. Building upon these findings, we develop a web-based system for real-time interaction with virtual musical instruments, incorporating both convolutional and recurrent architectures to enhance the user experience. We also explore audio-driven dance motion synthesis, where deep convolutional architectures incorporating a conditional autoencoder with dilated causal highway gates, outperform recurrent models in generating diverse and realistic dance motion sequences. Next, in the context of jazz improvisation, we focus on simulating the interplay between human soloists and artificial accompanists, highlighting the challenges and prospects of implicit machine learning approaches in modeling musical interactions. Lastly, we investigate the impact of symbolic music encodings on automatic music generation, emphasizing the importance of careful encoding design characteristics in shaping the resulting musical structure.

Overall, our research provides valuable insights into the performance and potential of different computational architectures across various tasks in computational music generation and interaction. The successful integration of convolutional and recurrent models demonstrates their ability to model complex musical interactions. We emphasize the importance of

selecting the appropriate computational architecture based on the task-specific goals and constraints. Our findings lay the foundation for future research, encouraging further exploration of advanced architectures, larger datasets, and more diverse tasks to continue pushing the boundaries of computational music generation and interaction. By doing so, we can unlock new possibilities for creative expression, human-computer collaborations, and the advancement of music technology.

List of Publications

1. Kosmas Kritsis, Aggelos Gkiokas, Maximos Kaliakatsos-Papakostas, Vassilis Katsouros, and Aggelos Pikrakis. “Deployment of LSTMs for Real-Time Hand Gesture Interaction of 3D Virtual Music Instruments with a Leap Motion Sensor”. In: *Proceeding of the 15th Sound and Music Computing Conference (SMC2018), Limassol, Cyprus*. Limassol, Cyprus, July 2018, pp. 331–338. DOI: 10.5281/zenodo.1422601
2. Kosmas Kritsis, Maximos Kaliakatsos-Papakostas, Vassilis Katsouros, and Aggelos Pikrakis. “Deep Convolutional and LSTM Neural Network Architectures on Leap Motion Hand Tracking Data Sequences”. In: *2019 27th European Signal Processing Conference (EUSIPCO)*. 2019, pp. 1–5. DOI: 10.23919/EUSIPCO.2019.8902973
3. Kosmas Kritsis, Aggelos Gkiokas, Aggelos Pikrakis, and Vassilis Katsouros. “Attention-Based Multimodal Feature Fusion for Dance Motion Generation”. In: *Proceedings of the 2021 International Conference on Multimodal Interaction*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 763–767. ISBN: 9781450384810. DOI: 10.1145/3462244.3479961
4. Kosmas Kritsis, Aggelos Gkiokas, Aggelos Pikrakis, and Vassilis Katsouros. “Danceconv: Dance motion generation with convolutional networks”. In: *IEEE Access* 10 (2022), pp. 44982–45000. DOI: 10.1109/ACCESS.2022.3169782
5. Kosmas Kritsis, Theatina Kylafi, Maximos Kaliakatsos-Papakostas, Aggelos Pikrakis, and Vassilis Katsouros. “On the adaptability of recurrent neural networks for real-time jazz improvisation accompaniment”. In: *Frontiers in artificial intelligence* 3 (2021). ISSN: 2624-8212. DOI: 10.3389/frai.2020.508727
6. Manos Plitsis, Kosmas Kritsis, Maximos Kaliakatsos-Papakostas, Agge-

los Pikrakis, and Vassilis Katsouros. “Towards a Classification and Evaluation of Symbolic Music Encodings for RNN Music Generation”. In: *Proceedings of the 1st Joint Conference on AI Music Creativity* (Stockholm, Sweden). Stockholm, Sweden: AIMC, Oct. 2020, p. 8. DOI: 10.5281/zenodo.4285410

Contents

List of Figures

List of Tables

Listings

Acronyms

Adam Adaptive Moment estimation. 38, 45, 81, 115, 154

AI Artificial Intelligence. 24, 133, 168

ARMA Autoregressive Moving Average. 106

BLSTM Bidirectional Long Short-Term Memory. 38, 39

BPM Beats per Minute. 142, 143, 157, 161

CAU Choreographic Action Unit. 69

CDE Collision Detection Engine. 56, 57, 58

CNN Convolutional Neural Network. 14, 15, 17, 19, 25, 26, 28, 29, 34, 35, 36, 37, 45, 46, 47, 48, 65, 67, 69, 71, 73, 74, 85, 86, 100, 168, 171, 172, 173

CPU Central Processing Unit. 38, 45, 116

DCHC Dilated Causal Highway Convolution. 73, 74, 75, 76, 78, 81, 85, 86, 100, 172, 173

DOF Degree Of Freedom. 23

DTW Dynamic Time Warping. 29, 52

FCRBM Factored Conditional Restricted Boltzmann Machines. 69

FFT Fast Fourier Transform. 72

FID Fréchet Inception Distance. 85, 86, 89, 90, 93, 98

FLMP Fuzzy Logical Model of Perception. 4, 5, 6

FOV field of view. 32, 58

FPS frames per second. 32, 72, 87, 88, 93, 97

GAN Generative Adversarial Network. 14, 64, 70, 105, 107

GMM Gaussian Mixture Model. 24, 52

GPU Graphics Processing Unit. 39, 45, 61, 66, 116

GRE Gesture Recognition Engine. 56, 58

GRU Gated Recurrent Unit. 69, 70, 90, 108

GUI Graphical User Interface. 13, 59, 116

HAR Human Activity Recognition. 29

HCI Human-Computer Interaction. 13, 14, 15, 23, 30, 49, 51

HMD Head-Mounted Display. 52, 54

HMM Hidden Markov Model. 24, 29, 52, 106, 108

HTML HyperText Markup Language. 49, 54

IE Interaction Engine. 56, 58

IOI Inter-onset Interval. 157

KDE Kernel Density Estimation. 157

KLD Kullback–Leibler Divergence. 157, 158, 162

KNN K-Nearest Neighbor. 28

LMA Laban Movement Analysis. 67

LSTM Long Short-Term Memory. 17, 18, 25, 26, 29, 34, 35, 36, 37, 38, 39, 43, 45, 46, 47, 48, 66, 67, 69, 70, 85, 86, 100, 106, 107, 108, 114, 115, 116, 117, 128, 130, 137, 138, 147, 154, 156, 165, 166, 171, 172, 173

MDN Mixture Density Network. 67

MIDI Musical Instrument Digital Interface. 66, 105, 113, 114, 133, 134, 139, 142, 145, 146, 151, 162, 168

MIR Music Information Retrieval. 135, 137, 139

ML Machine Learning. 24, 52, 110, 128, 135, 137, 172

MLP Multilayer Perceptron. 28

MoCap Motion Capture. 23, 50, 52, 64, 66, 67, 69

NC Note Count. 157

NLH Note Length Histogram. 157

NLP Natural Language Processing. 133

NLTM Note Length Transition Matrix. 157, 162, 166

OA Overlapping Area. 157, 158, 162

PC Pitch Count. 157

PCH Pitch Class Histogram. 157, 158

PCTM Pitch Class Transition Matrix. 157, 162

PDF Probability Density Function. 157, 158, 162

PR Pitch Range. 157

PS Pitch Shift. 157

RAM random-access memory. 38, 45

ReLU Rectified Linear Unit. 36, 45, 76

RNN Recurrent Neural Network. 14, 15, 17, 18, 25, 29, 35, 37, 64, 65, 66, 67, 69, 70, 71, 73, 74, 80, 83, 90, 100, 106, 107, 114, 115, 136, 137, 138, 140, 154, 156, 162, 171, 172, 173

SDK software development kit. 32, 38, 50

seq2seq sequence-to-sequence. 67, 154

STD standard deviation. 43

STEAM Science, Technology, Engineering, Arts and Mathematics. 26, 50, 59

SVM Support Vector Machine. 27, 28, 107

TTS text-to-speech. 63

VAE variational autoencoder. 70, 89, 90, 108, 138

VGG Visual Geometry Group. 69

VR Virtual Reality. 23, 49, 52, 54

Part I

Preface

CHAPTER 1

Introduction

1.1 Human Interaction and Multimodality

Our daily interactions require us to process information from multiple sensory modalities, such as hearing, sight, touch, smell, and taste, in order to deduce emotions and intentions. Simple day-to-day decisions and actions are often the result of evaluating non-verbal cues, such as facial expressions and body gestures, as well as verbal cues, like vocal tone and inflection. A well-known example of human multimodal integration is the so called McGurk effect, which demonstrates how humans use both visual and auditory information to perceive speech [1, 2].

The study of this phenomenon involved recording a voice articulating one consonant and combining it with a video of a face articulating a different consonant. Despite the acoustic speech signal being easily recognizable on its own, when combined with incongruent visual speech, it was usually perceived as a different consonant, especially an auditory */ba/* paired with a visual */ga/* often resulted in the percept */da/* (see Figure 1.1). This illusion has been replicated numerous times and has led to an abundance of research, mainly due to its striking demonstration of multisensory integration. Various theories have been proposed to interpret this observation.

The initial hypothesis put forth by McGurk and MacDonald was that visible speech drives the perception of the position of articulation, while audible speech determines the perception of voicing [4]. Various researchers have characterized the McGurk effect as solely a fusion effect because it involves

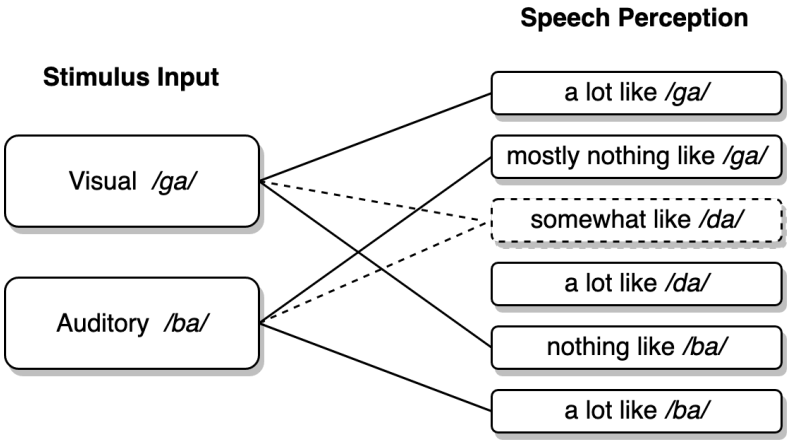


Figure 1.1: The classic example of the McGurk effect is that an auditory /ba/ paired with a visual /ga/ often produces the percept /da/ [3].

the integration of auditory and visual information resulting in the perception of a third consonant [5–7]. However, this definition fails to acknowledge that other incongruent audio-visual stimuli can produce different types of percepts. For instance, the reverse combination of the consonants (i.e. auditory */ga/* and visual */ba/*) is perceived as */bagba/*, where the visual and auditory components oscillate one after the other [4].

To this end, a line of studies focused on how individuals perceive both the acoustic and visual components of the stimulus, aiming to explain perceptual processing independently of motor behavior [8–11]. Especially, this concept has been extensively explored in great detail by Massaro and colleagues [12–17], proposing a prototypical pattern recognition approach called the Fuzzy Logical Model of Perception (FLMP), where the accuracy of identifying the unisensory components designate audiovisual speech perception.

Figure 1.2 depicts the three key processes involved in the FLMP: evaluation, integration, and decision. During the evaluation process, each source of information is transformed into psychological values indicating the degree of support for various alternatives. These values are then integrated to provide an overall degree of support for each vocabulary alternative. The decision operation maps these integrated outputs to an appropriate alternative. The model is based on several assumptions:

- each source of information is independently evaluated to determine

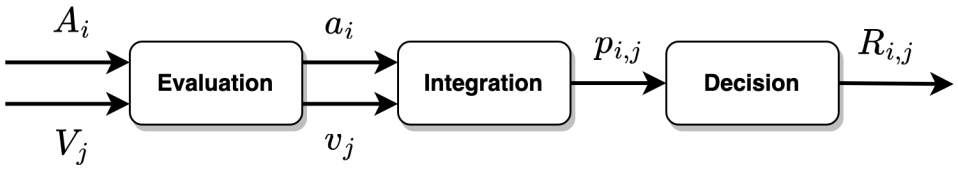


Figure 1.2: Schematic diagram of the three operations assumed by the FLMP [14]. The sources of information are represented by uppercase letters. Auditory information is represented by A_i, and visual information is represented by V_j. The evaluation process transforms these sources of information into psychological values (indicated by lowercase letters a_i and v_j). These sources are then integrated to give an overall degree of support for a given alternatively $p_{i,j}$. The decision operation maps this value into some response, such as a discrete function or a rating, R_{ij}.

its continuous degree of specification for various alternatives;

- the sources of information are evaluated independently of each other;
- the sources of information are integrated to provide an overall continuous degree of support for each alternative; and
- the relative degree of support among the various alternatives determines perceptual identification and interpretation.

To ensure a consistent metric for the degree of match of each feature, fuzzy-truth values are employed [12]. These values offer a natural representation of the degree of match in cases where multiple sources of information are present. For instance, in the paradigm of Figure 1.1, by assuming the truth values as "a lot like" $= 0.9$, "somewhat like" $= 0.7$, "mostly nothing like" $= 0.3$ and "nothing like" $= 0.1$, it is evident that */da/* would have almost twice as much support as the other options (i.e. using multiplicative integration of the FLMP, support for */ga/* $= 0.9 * 0.3 = 0.27$, support for */ba/* $= 0.1 * 0.9 = 0.09$, while support for /da/ $= 0.7 * 0.7 = 0.49$).

Overall, the FLMP predicts that the contribution of one source of information to performance increases as the ambiguity of the other available sources of information increases. This means that when one source of information is unclear or ambiguous, the brain relies more on other available sources to make sense of the information. For example, in a noisy environment where speech is difficult to hear, visual cues such as lip movements become more important in understanding speech. Similarly, when

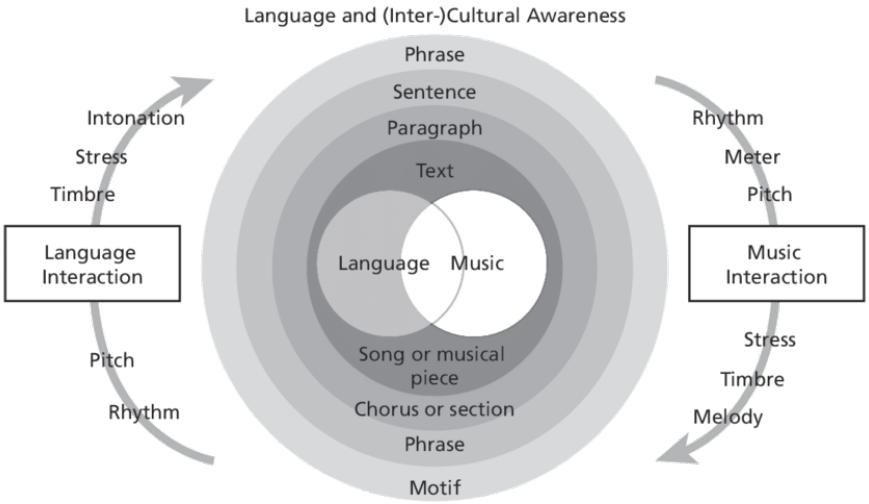


Figure 1.3: Parallels between elements and structures of language and music [23, 24].

the visual information is unclear or ambiguous, the brain relies more on the auditory information to perceive speech. The FLMP provides a framework for understanding how the brain integrates and weighs different sources of information to perceive speech [17].

Language can be also considered as an embodied system that involves physical expression through gestures [18]. Even though speech is the dominant form of communication, manual gestures still accompany it. These visuomanual properties of language are evident in various forms, including sign language, handwriting and even typing [19]. Therefore, the manual actions associated with co-speech gestures are closely connected to the language system. Consequently, the occurrence of speech along with co-speech gestures exemplifies the interaction between language and action [20]. This communication model shares common characteristics with other domains in which multimodal human perception and action intersect, such as musical interaction [21, 22]. Both music and language are universal human abilities that share many similarities, including their acoustics, structure, and frequent use in social settings [23–25]. Therefore, it can be predicted that they are processed and understood similarly. An increasing body of research supports this prediction, indicating that music and language are indeed processed and understood in similar ways.

As depicted in Figure 1.3, most similarities between music and language are found in terms of acoustics and structure [23, 24]. Rhythm, for instance,

establishes a regular beat in music and maintains a definite syllable rate in speech, such as in speech tempo [26]. In music, pitch advances the melody, while in language it underlies prosody in expression and conversation. Both music and language consist of numerous repeated units with an infinite number of combinations, often organized hierarchically and frequently performed in coordination with others [27]. Both systems are therefore rule-based and can be conveyed in written form. Furthermore, they are found in all known cultures [28, 29] and are often combined, such as in singing [30].

1.2 Musical Interaction

Music involves the integration of various human cognitive, perceptual, motor, and emotional abilities [31]. However, music cognition works differently from language, even though the roots of both processes might be the same [32]. In particular, music has a “floating intentionality”, lacking the clear referentiality that allows to refer to musical experiences by means of language description [33]. Unlike many human activities that utilize only specific parts of the brain, research using imaging techniques has shown that both playing and listening to music involve coordinated activity in multiple brain regions, including the prefrontal, motor, sensory, visual, and auditory cortices, as well as the corpus callosum, which connects the two hemispheres of the brain, and various structures that govern memory, emotion, and balance [34, 35].

However, human engagement with music is not solely a product of human intellectual ability. A valuable perspective on the musical communication chain was presented by Richard Moore in his book Elements of Computer Music [36]. Although his view is rooted in traditional musical concepts, it provides an excellent starting point for understanding the interdisciplinary issues that arise when studying human-music interactions. As illustrated in Figure 1.4, the communication chain is a loop of interconnected signals (data) and processes (transformations of the signals) that encompasses all the elements involved in composing, performing, and perceiving music. This loop results from the intersection of the musical knowledge, which is primarily mental and based on our cultural tradition, and the physical knowledge, in which the laws of physics have a greater influence.

Starting at the top of the loop, the composer uses perceptual inputs and personal musical background to create a symbolic representation that con-

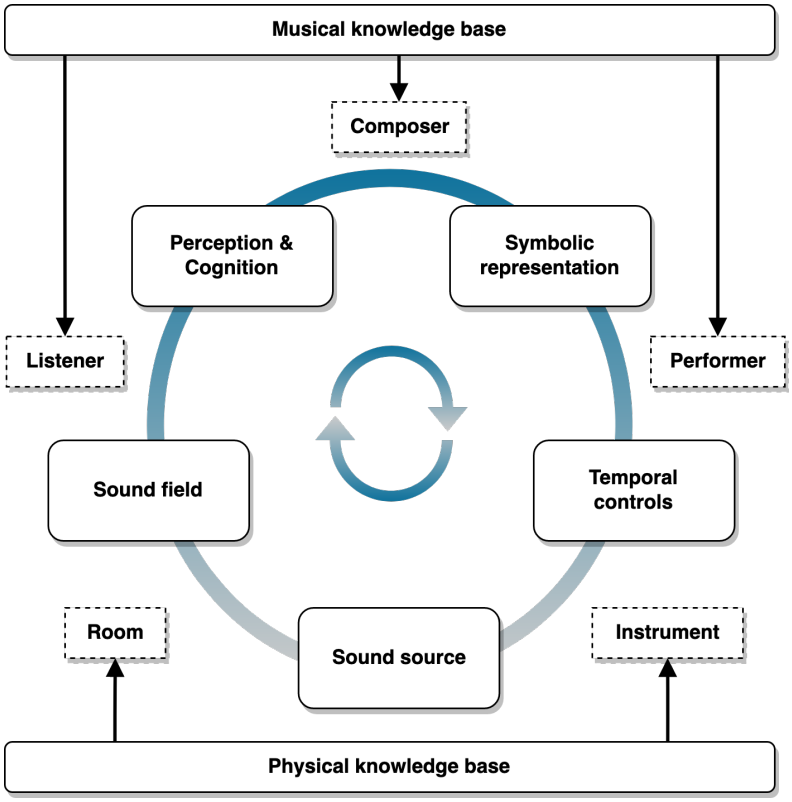


Figure 1.4: Musical communication chain involves musical data information (boxes on circle) that exist in multiple modalities and processors (dashed boxes), which may be human beings, machines, or a combination of both [36].

veys a musical idea. Music composition, similar to other forms of art, entails a multifaceted and intricate creative process. While this process can differ greatly based on individual preferences [37], musicological analysis and interviews with novice and professional composers, indicate that there are common creativity approaches, characterized by either divergent or convergent thinking [38]. Divergent thinking aims to generate a wide range of new and different ideas, often with a “chance of discovery” in mind [39]. Conversely, convergent thinking is geared towards “seeking the suitable”, involving the evaluation of ideas and adapting the selected few to better serve the creative purpose [39]. These two processes often intertwine multiple times during the compositional process, as musical sub-problems arise and get resolved. From this symbolic representation, a performer produces gestures or temporal controls to operate a musical instrument, relying on

shared musical backgrounds. The instrument is a physical object that can produce sound from the performer's gestures. The sound source produced by the instrument then creates a sound field in a room that a listener perceives. Finally, the listener processes the acoustic signal that enters their ear, using previous perceptual experiences to have a perceptual and cognitive experience. The composer then closes the loop by incorporating their own perceptual and cognitive experiences into the musical creative decisions. Globally speaking, music perception can be described as a process that involves three stages [40], namely:

- the extraction of basic patterns from an audio signal,
- the analysis of these patterns by means of the existing structures that reside in memory (also called "knowledge" or mental representation), and
- associated experiences (e.g. tension, relaxation, affect, emotion).

Nevertheless, this is a traditional view of the music communication chain, and it is evident that scientific and technological developments have significantly impacted this loop. One critical alteration has been the increased flexibility of some processes, as they can now be accomplished not only by human beings or mechanical devices, but also by machines and software algorithms. For instance, a physical room may be entirely skipped in the communication chain, as we can attend to live performances though online streaming services [41], or a performer may be also the composer, creating sounds directly with a computer without the traditional performer-instrument interaction [42]. The most significant impact has been the incorporation of new creative possibilities by developing computational models for specific processes involved in the chain [43]. However, performance on both traditional and digital musical instruments requires precise control of sensory-motor loops in real-time, including not only note onset but also pitch and timbre, that involves muscular manipulation at a millisecond scale.

To this end, it is clearly visible that music is also a highly embodied activity that involves intricate, synchronized movements by both performers and listeners in response to the sonic environment [44]. This is motivated by the fact that many people do not engage with music in terms of narrative reflections or interpretations of the music's intentions, as evidenced

by the non-verbal communication in musical social behaviors (e.g. concerts) [45]. Music serves not only as a socially embodied practice but also possesses a complex abstract aspect. Many people do not have the necessary background in music analysis, history, and culture to understand and project subjective experience onto a linguistic narrative of cultural meanings. However, this does not necessarily mean that they are barred from making sense of music. Instead, it can be observed that people often listen to music for its direct corporeal value, such as for relief after a stressful day, improving mood, or distracting from repetitive tasks. People seek music for its capacity to create behavioral resonance and its effects on mood [46]. During these activities, most people tend to engage with music in a corporeal way rather than a cerebral way [44].

This corporeal engagement with music broadens the perspective of what musical communication is about and forms the basis for a range of interactions with music. In addition to interactions based on linguistic or verbal narrative descriptions and interactions based on symbolic or visual signs with information stored in lists, such as scores and tables containing descriptions of musical properties, we should also consider other forms of interactions with music. These may include interactions based on mimetic skills or rehearsed action scenarios [47], such as playing a musical instrument, interactions based on goal-directed gestures that are highly culture-dependent but do not require highly developed skills, such as symbolic gestures, and interactions based on direct episodic action sequences that involve responses based on our emotive, affective, and expressive capabilities [48]. Nonverbal expressions, such as interactive bodily movements in a musical context, depend on converting acoustical motion patterns into motor motion patterns. The motor mode represents the manifestation of the initial auditory experience and can be viewed as a portrayal of the original music through bodily movements [46].

In other words, the theory of embodied music cognition proposes that the body plays a crucial role in the music communication chain. This is achieved through corporeal intentionality, which involves translating the physical properties of music, such as frequency and amplitude, into a mental representation of objects with specific qualities, goals, and intentions. The action-perception couplings of a musician, which are developed through embodied skills, knowledge, and experience, facilitate a pre-reflective attunement between the musician and the musical world during perfor-

mance [49]. By using learned schemas of action-perception couplings to transform sound patterns into corporeal articulations, such as gestures and musical patterns, a musician can resonate with the music in a corporeal way and develop a deep understanding of its meaning [50]. This experiential, corporeal basis for understanding music is known as embodied interaction with music, in which the musician directly engages with the musical environment they create while playing.

Although the study of gestures in music has a long history in musicology [51] and research on interactive music systems [52], recent efforts have focused on understanding their relationship to music [53]. Gestures play a role in encoding and decoding musical expressions [54]. During performance, "expression-supporting" gestures encode expressive qualities in the music. During listening, "expression-responding" gestures decode these qualities. Both cases assume that expression can be transferred between gesture and music. In addition, gestural shapes can relate to timing, particularly in the context of dance. Dancing provides various means of acquiring bodily knowledge, such as through kinesthetic sensing in the muscles, perception via other senses, concentration on internal stimuli within the body, association of movement qualities with emotions, and interaction with other dancers while moving through space. Through the act of dancing, all of these aspects of sensory and bodily experiences are synchronized and interrelated [55]. Repetitive gestures are constrained within a timing framework dictated by the musical meter, and movement choreography is based on a basic shape that reappears over set time periods. Elements of the basic shape are linked to timing due to the overall timing framework set by the music.

CHAPTER 2

Scope

2.1 Challenges in musical interaction and generation

Although humans have been interacting with the natural world through multiple senses, conventional Human-Computer Interaction (HCI) has mainly focused on unimodal communication [56]. This means that information or data is conveyed primarily through a single mode or channel, such as typing text with a keyboard for input and visualizing the output on a screen. Despite the fact that most computer interactions have been multimodal to some extent (combining typed text with switches, buttons, mouse movement and clicks, along with various visual and auditory output signals), the predominant model for interactive computing has been based on a single primary channel for data input and a different primary channel for data output. The concept of multimodal interfaces refers to interactive systems that aim to capitalize on the innate communication abilities of humans through speech, touch, gesture, facial expressions, and other modalities. Such interfaces incorporate advanced techniques for recognizing and categorizing patterns, leading to more sophisticated HCI systems [57]. Although they are not expected to completely replace conventional desktop and Graphical User Interface (GUI)-based applications, multimodal interfaces are gaining significance due to hardware and software advancements, their potential to offer benefits to users, and their compatibility with the growing prevalence of mobile computing [58].

In this dissertation, the term “musical interaction” is used specifically to refer to music and multimodal HCI. Music interaction can pose unique challenges for HCI researchers and designers, due to the intricate and demanding nature of musical activities and their multimodal connection with various human capabilities [53]. However, it also provides a valuable source of inspiration for generating new ideas and techniques for musical HCI. Moreover, emerging forms of musical interaction have the potential to revolutionize music in various domains, which can have significant implications for musicians, educators, learners, and anyone interested in exploring music in a deeper way. Theories from fields such as mathematics, computational neuroscience, and cognitive psychology have been applied to illuminate aspects of musical practice and aesthetic and subjective musical judgment.

Some key topics of research in musical HCI include the role of body motion and gestures in musical communication (embodied interaction), and the use of automatic music generation systems to augment musical experiences and creativity [59]. In other words, automatic music generation refers to the use of computer algorithms and machine learning techniques to create music automatically, with or without direct input from human composers or performers. On the other hand, embodied musical interaction maps physical movements to musical features by utilizing gestures to manipulate musical content in real-time. One approach to achieve this is by utilizing sensors and motion capturing technology to detect body or hand movements, which can be translated into musical actions. Alternatively, physical interfaces like instruments or controllers can also be used to create and manipulate music [60, 61].

There are various approaches proposed in the literature for automatic music generation, ranging from rule-based systems that use pre-defined musical patterns and structures to generate new pieces, to deep learning models that analyze existing musical data and use that information to generate new compositions [62]. Deep learning models have shown great promise in generating musical content that is both creative and convincing to human listeners. These models are typically trained on large datasets of existing music, and use various techniques such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Generative Adversarial Networks (GANs) to generate new pieces of music [63]. One of the main challenges in deep learning models for music generation is to ensure that the generated music is both coherent and musically interest-

ing [64]. This requires the model to have a deep understanding of musical structure, including melody, harmony, and rhythm. Another challenge is to incorporate human feedback into the generation process. While deep learning models can generate music autonomously, they may not always produce music that is appropriate for a particular context [65]. Incorporating feedback from human listeners or musicians can help to guide the generation process and ensure that the output is of proper quality. Finally, there is also a challenge in creating models that are able to interact with human musicians in real-time [66]. This requires the computational model to be able to process and respond to musical input in a way that is both timely and musically meaningful.

Similarly, incorporating deep learning models into embodied musical HCI systems poses several challenges. One significant hurdle is the development of computational systems capable of processing multimodal data, such as audio and motion tracking, to fully capture the complexity of embodied musical experiences. Deep learning models may struggle to analyze the nuances of musical expression and the subjective experiences of listeners [53]. In addition, there is a need for large, diverse datasets to effectively train deep learning models, which is challenging and time-consuming. Especially in the context of embodied musical interaction, this process often requires specialized equipment and expertise [67]. Lastly, interpreting and explaining the results of deep learning models in embodied musical interaction presents another obstacle [68]. These models can be highly complex and opaque, making it difficult to understand how they generate their predictions and recommendations.

2.2 Goals and Contributions

This dissertation tries to address the challenges of deep learning models in musical interactive systems, from both technical and aesthetic perspectives. We implement several computational models based on RNN and CNN architectures, and evaluate their performance on different musical tasks, including automatic music generation, real-time jazz accompaniment, musical gesture recognition and audio-driven dance motion synthesis. In this regard, our research tries to answer the following questions:

- Can we leverage raw data from motion capture sensors to train a musi-

cal gesture recognition system, considering the great amount of temporal information of such gestures, and their subtle and short duration?

- Can we use such mid-air gestures to interact with virtual music instruments that simulate real-world instruments, considering the absence of physical constraints (e.g. specific tangible instrument body or string positions)?
- How can we enhance the performance of gesture recognition systems, considering the great diversity in the way a single gesture is performed by different or even individual users?
- How effective are pretrained pose detection algorithms in constructing skeletal training data?
- Can deep learning models be trained to analyze the cross-modal interconnection between auditory stimuli and dancing motion?
- What can we deduce from the different approaches to encode symbolic musical data?
- Is it possible to categorize them systematically?
- What are the advantages and drawbacks for each one, especially regarding their impact on the form of a generated musical piece?
- Can the proposed deep learning systems capture the static harmonic information of a given chart in a dynamic environment?
- How well do different computational models enable the musical interactive systems to adapt to changing limitations that are reliant on human input?
- Can the proposed computational models be implemented in real-time settings, considering their computational complexity and robustness?
- Can we visualize the feature learning process and reason about the decisions made by the computational models?

Thus far, the approaches developed in the literature present minimal controllability and usually generate static predictions with low variability, compared to the methods discussed herein, proposing dynamic changes to the system's responses, which are depended on the human intentions and input. Consequently, the broad scope of this research has enabled us to

achieve several contributions, which can be summarized as follows:

- Implemented a multimodal capturing system to record and analyze Leap Motion sensorial data intuitively, based on a scalable and user-friendly web interface, which can be adopted to the needs of any hand motion analysis task[1].
- Collected and released a dataset involving 8 classes (total 1200 samples) of musical gestures that belong to 2 instrumental interaction families, namely, tapping and plucking[2].
- Developed a series of deep learning models towards recognizing dynamic hand gestures with the Leap Motion sensor in the context of musical performance with virtual instruments, simulating percussive and stringed based interactions.
- The gesture recognition models were integrated and deployed as part of the iMuSciCA web platform, validating their real-time effectiveness[3].
- Proposed a multimodal deep CNN architecture capable of generating novel dance motion sequences of arbitrary length, based on an autoregressive curriculum learning training scheme, to deal with prediction error accumulation during inference.
- Designed informative visualizations of the attention-based multimodal feature fusion method, helping to enhance the comprehension of the generated dance motion sequences.
- Validated the effectiveness of pretrained pose detection models as a data collection methodology and released a novel dataset containing paired music and 2D skeletal pose sequences[4].
- Proposed a taxonomy of monophonic symbolic music encodings that accounts for common musical characteristics.
- Developed an automatic music generation system based on a Long Short-Term Memory (LSTM) RNN architecture that was trained on a diverse dataset of musical pieces, by re-encoding them in various for-

[1] https://github.com/kosmasK/air-writing-recognition
[2] https://zenodo.org/record/1260336
[3] https://workbench.imuscica.eu
[4] https://github.com/kosmasK/multimodal-dance-generation

mats[5].

- Evaluated empirically the impact of the different music encodings on the output of the system, supported further by visual explanations based on depictions of the corresponding latent representations learned by the computational model.
- Developed a two stage multi-layered system based on LSTM RNN architectures to generate real-time music accompaniment, capable to adapt to the intentions of the human soloist, by modelling the expectation and its violation in the context of jazz improvisation performance[6].
- Released a refined symbolic music dataset according to Jazz standards, enhancing the variability of the accompaniments of the pieces available.[7]

2.3 Thesis Outline

Research on musical interaction requires employing interdisciplinary means of understanding based on dynamic perceptual principles. In that respect, the main concepts, empirical studies, subjective evaluations and modeling approaches that are developed in this dissertation stem from those principles. Our systems receive and analyze various forms of music related information, including raw audio, sensorial and skeletal data, as well as different types of symbolic representations. Hence, the remainder of this dissertation is organized into three main parts.

Part II focuses on the utilization of skeletal features to train various architectures, both unimodal and multimodal, in the tasks of musical gesture recognition, virtual music instrument performance and audio-driven dance motion synthesis. Specifically, in Chapter 3 we explore computational models for musical gesture recognition using the Leap Motion sensor. We compare different architectures, including recurrent and convolutional models, and evaluate the performance of handcrafted and "raw" features. Leveraging the effectiveness of our gesture recognition system, we then develop a web-based system for interacting with 3D virtual music instruments, as described in Chapter 4. The system utilizes the Leap Motion sensor and

[5] https://github.com/manosplitsis/MusicRep

[6] https://github.com/kosmasK/JazzICat

[7] https://zenodo.org/record/3523222

employs state-of-the-art web technologies, enhancing the virtual musical instruments with realistic visual and audio feedback. Our final system that addresses skeletal-based features is presented in Chapter 5, where we examine the application of deep CNN architectures for modeling and synthesizing novel dance motion sequences conditioned on audio features. The proposed deep multimodal architecture benefits from the stacked dilated convolutional operations for capturing the long-term spatio-temporal context of dance sequences, thus allowing synthesis of arbitrarily length.

Next, in Part III we consider deep learning models trained with symbolic music data in the tasks of automatic music generation and real-time music accompaniment synthesis, towards investigating the effects of different data encodings on the latent representations captured by the computational models. In Chapter 6 we propose a system that simulates the interplay between a human soloist and an artificial accompanist, in the context of jazz improvisation. Our proposed methodology includes the development of "a model within a model", that allows the artificial agent to have its own model of anticipation for the human improviser. The system demonstrates harmonic compliance with chart chords and adaptability based on contextual information. Further experimentation with various symbolic music data encodings for automatic music generation, are investigated in Chapter 7, emphasizing their impact on the resulting musical structure. Evaluation results demonstrate that the choice of encoding method significantly influences aspects such as melodic phrasing and metric organization.

Finally, in Part IV we conclude this dissertation with a summary of the key findings of our research and offer a discussion on possible future research directions.

Part II

Skeletal Features

CHAPTER 3

Gesture recognition for musical interaction

3.1 Introduction

With the recent advancements in Motion Capture (MoCap) technology, there has been a growing demand for touchless interfaces. This has, in turn, sparked further research endeavors in the fields of human motion analysis and gesture recognition. HCI applications that directly utilize gesture recognition algorithms include sign language recognition [69], robotics [70], Virtual Reality (VR) [71], and musical interaction [72] among others. This Chapter centers around hand gesture recognition, with two primary types of gestures identified in the literature: (a) *static* and (b) *dynamic* hand gestures [73]. *Static* hand gestures involve capturing postural information by extracting hand contours and regions of interest at a specific moment. In contrast, *dynamic* hand gestures focus on the temporal evolution of finger joint positions and overall hand movement throughout a performed motion pattern.

However, dynamic hand gestures are still challenging to be robustly recognized by computational models, due to the complex anatomy of the hand skeleton, which provides 27 Degrees Of Freedom (DOFs) in total [74]. The motion trajectories of fingers present great diversity of possible shapes, which could be subjected to serious occlusions [75], as well as diverse intra-class and inter-class similarities that further complicate the recognition pro-

cess [76]. Moreover, the musical gestures performed during an instrumental performance are influenced by various perceptual mechanisms, including visual, auditory, and tactile sensory cues. These factors need to be considered when designing gesture-driven virtual musical instruments [77].

With the emergence of depth sensors such as the Creative Senz3D[1], Microsoft Kinect[2] and Leap Motion[3], various studies have explored the potential of employing such controller interfaces for interacting with virtual musical instruments [78–80]. However, the interaction frameworks that are usually employed in these gestural applications do not consider the active role of prediction, rather than focusing on reactive mappings [81]. Recent advancements in the fields of Machine Learning (ML) and Artificial Intelligence (AI) outperformed traditional systems based on Gaussian Mixture Models (GMMs) and Hidden Markov Models (HMMs) approaches [82], while bringing tremendous improvements in temporal pattern recognition tasks. Nonetheless, significant work remains to be done for achieving satisfactory recognition performance in expressive and real-time musical interaction scenarios [83].

Therefore, this Chapter focuses on recognizing dynamic hand gestures by utilizing the Leap Motion sensor in the context of musical interaction with virtual instruments, inspired by real-world instrumental gestures. Recognizing such gestures is challenging since:

- they convey a great amount of temporal information;
- the recognition window is very short;
- there is a lack of physical constraints (e.g. specific tangible instrument body or string positions); and
- there is great diversity in the way a single gesture is performed by different or even individual users.

The proposed computational models in this Chapter receive hand motion tracking data captured with the Leap Motion sensor. The sensorial features are processed by a sliding temporal window that constructs subsequent overlapping sequences, to be used as input to the gesture recognition systems. In all our experiments, we consider a classification task between 7

[1] `https://us.creative.com/p/web-cameras/creative-senz3d`
[2] `https://developer.microsoft.com/en-us/windows/kinect`
[3] `https://www.leapmotion.com/`

predefined musical gestures, along with a dedicated category to handle arbitrary (i.e. irrelevant) gestures (tagged as "unknown"); thus modelling two families of musical gestures, namely, tapping and plucking, towards simulating percussive and stringed based instrumental interactions respectively. In our initial experiments, we evaluate the performance of a recurrent computational architecture that utilizes a dense layer to compute feature embeddings, followed by an LSTM network for modelling the temporal feature patterns, achieving average recognition rate of 91.77% in a 10-fold cross-validation setup. In a cross-participant setup, where the data of a participant are used as the test set, we report a recognition rate of 85.5%. Also, we examine the incorporation of a few gesture samples from test participants, simulating the scenario of fine-tuning the system with the users' own gestures to enhance accuracy.

Next, we explore the filtering effectiveness of convolutional filters to model multidimensional data sequences and propose two computational models that incorporate CNNs for the classification task at hand. First, we test the effectiveness of 1D CNNs to produce feature embeddings by replacing the dense layer in our initial architecture, and keeping the LSTM network for sequence modelling. Our second method relies completely on a deep CNN architecture that utilizes consecutive convolutional and max pooling operations. The experimental results demonstrate statistically significant improvements compared to our initial dense-LSTM method, reporting average accuracies of 94.32% and 94.44% respectively. Our results pertain to window-based accuracy, which measures the recognition accuracy within a window of Leap Motion frames. This window-based accuracy enables real-time application in scenarios involving consecutive windows of gesture recognition computations.

The remainder of this Chapter is organized as follows. In Section 3.2 we introduce the research problem that we aim to address with the study presented herein, along with its application context. Next, in Section 3.3 we discuss some related studies in the context of dynamic hand gesture recognition and gesture-based musical interaction. In Section 3.4 we describe the collected dataset and propose various computational architectures, based on convolutional and recurrent operations. In Section 3.5 we present the experimental setup and report the results of an initial ablation study, towards evaluating the contribution of different data features, embeddings and architectural configurations of the RNN model. Likewise, in Section 3.6

we present the experimental setup and evaluation results regarding the different computational models that utilize CNNs. Finally, Section 3.7 concludes this Chapter by summarizing the main contributions and observations drawn from our study.

3.2 Problem Definition and Application Context

In the scope of the iMuSciCA[4] project, we developed a Science, Technology, Engineering, Arts and Mathematics (STEAM) education platform that allows students, among other activities, to employ scientific and engineering principles in constructing their custom virtual musical instruments with which they can interact. Accurate music performance is important for achieving the goal of STEAM education, which is the integration of all scientific elements of iMuSciCA in an enjoying and meaningful experience for the students. In order to provide a realistic performance experience associated with physical interactions (such as tapping a drum or plucking a string), it is important to utilize appropriate gestures inspired by real-world instrumental performance. However, developing virtuosity in playing a virtual musical instrument is not necessary, as these instruments can be customized to produce a diverse range of timbres and specific note sets that are meaningful and interesting. In this regard, interacting with virtual instruments can serve as a medium for musical expression, even for individuals without traditional musical training (not only within the context of STEAM education).

In this Chapter we focus on exploiting LSTM and CNN computational models in the dynamic hand gesture recognition task using Leap Motion data. The proposed algorithms specifically target quick gestures used to rapidly trigger events during musical performances with 3D virtual instruments, drawing inspiration from gestures employed during performance with physical instruments. The main objective is to enhance the user experience by enabling control of virtual musical instruments through intuitive hand gestures. While our goal is to create a versatile gesture recognition module suitable for various 3D virtual instruments, we emphasize on interaction with string and percussive instruments. Hence, the primary focus of the hand gesture classes in our study revolves around actions such as plucking a string with one of the fingers or performing a tapping gesture on

[4] http://www.imuscica.eu/

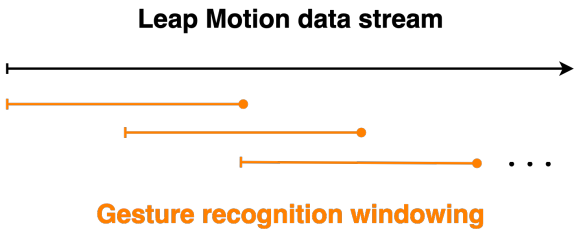


Figure 3.1: Overview of the temporal windowing, where successive overlapping input frame sequences of raw hand tracking data are fed to the recognition system.

a percussion instrument. Our objective is to achieve real-time recognition of these quick gestures by employing the proposed gesture recognition algorithms on continuously sliding overlapping windows, as illustrated in Figure 3.1. However, the quantitative evaluation that we present herein, focuses on the performance of the different gesture recognition models within a single window of data. As a result, the proposed methodology, collected data, and presented results primarily address strategies for training and evaluating our systems on a window level, encompassing a single gesture and not sequences of gestures. Also, it is important to highlight that the proposed methods have been proven effective in generic-purpose gesture recognition tasks as well [84].

3.3 Related Work

Early contributions to the task of 3D dynamic gesture recognition relied mostly on the extraction of hand features from consecutive RGB image sequences by applying computer vision and image processing techniques [85]. However, it is quite difficult to capture the rich spatio-temporal movement relations of dynamic hand gestures with a monocular camera sensor [75]. With the advent of innovative depth sensors such as the Microsoft Kinect and Leap Motion sensor, there was an increasing interest by the research community to incorporate depth tracking data streams for estimating real-time hand gestures.

One of the first studies that targets sign language recognition based on tracking data from a Leap Motion sensor was developed by Marion *et al.* [86], where ad-hoc hand features computed from the raw positions and orientations of the fingertips are fed to a Support Vector Machine (SVM) clas-

sifier for estimating the performed letter signs. Furthermore, they examined additional features extracted from a combination of depth data captured from both Kinect and Leap Motion sensors, which could increase the overall recognition performance. A similar approach was followed in [87], where Leap Motion and Kinect sensors were jointly calibrated by means of combining two classifiers for the different feature streams; a SVM classifier for the Leap Motion-based features and a Random Forest for the depth-based features of Kinect.

A music related project that utilizes both Leap Motion and Kinect sensors for interacting with a virtual musical instrument, called "Intangible Musical Instrument" [88], targets piano-like gestures. More specifically, two Leap Motion sensors (i.e. one for each hand) are dedicated to recognize the piano-like gestures which are directly mapped to the inherent dynamics, articulation and duration metaphors of natural music interaction, while a Kinect sensor targets the motion trajectories of the head, arms and vertebral axis, which are indirectly mapped to a granular sound synthesis engine. An attempt to use simple artificial neural networks for real-time recognition of piano-like gestures was introduced in the "Virtual Piano" instrument [89], where finger-based features were firstly computed from raw Leap Motion tracking data and then fed to a simple Multilayer Perceptron (MLP) classifier. An interesting study that focuses on out-of-range (not captured entirely by the sensors) and overlapping (with significant occlusion) hand gestures is called "Embodied Sonic Meditation" [90], where the authors experiment with K-Nearest Neighbors (KNNs), Binary Decision Trees and SVMs classifiers for recognizing 7 ancient Buddhist hand gestures named mudras.

With the breakthrough in deep learning architectures there is a growing trend towards the development of complex models that are able to extract high-level features and predict the spatio-temporal evolution of body and hand gestures. Among the most successful approaches incorporate the development of methods that rely on CNNs, which have proven effectiveness in filtering raw sensorial data. For instance, McCartney *et al.* [91] proposed a 3D gesture recognition model based on CNNs, which can identify 2D projections of the 3D space by combining the output of three different 2-layered CNNs for each individual projection plane (i.e. XY, YZ and XZ). A different approach was introduced by Molchanov *et al.* [92], who utilized depth and intensity channels, along with 3D CNNs, where each channel was fed to a dedicated network. Devineau *et al.* [93] introduced a hand gesture

recognition system based on 1D CNNs, that receive input sequences of raw hand-skeletal joint positions, achieving state-of-the-art performance on the Dynamic Hand Gesture dataset[5] from the 2017 3D Shape Retrieval Contest (SHREC) [94]. Deep CNNs have also been applied for Human Activity Recognition (HAR) from incoming skeletal tracking data provided by depth sensors [95]. However, most methods either consider pre-segmented gesture data sequences or treat segmentation and classification as separate problems. For instance, Dynamic Time Warping (DTW) [96] and HMMs [75, 97] have been used as segmentation methods to determine presumable start and stop points of gestures.

On the other hand, RNNs are able to address this issue, since their computational architecture allows to capture the temporal patterns within a gestural sequence. For instance, Molchanov *et al.* [76] proposed a gesture recognition system that utilizes a deep 3D CNN model for spatio-temporal feature extraction, a recurrent layer for global temporal modeling and a softmax layer for predicting class-conditional gesture probabilities. It is only recently that some approaches have proposed RNN architectures with LSTM gates for recognizing dynamic human gestures. For instance, Sarkar *et al.* [98] introduced a methodology that involves the initial computation of 3D hand descriptors from real-time depth data acquired from a mobile Time-of-Flight sensor. These computed descriptors are subsequently utilized as input for a single-layered deep LSTM network, which tackles a classification task with four distinct categories. Núñez *et al.* [99] addressed HAR and hand gesture recognition tasks, by employing a combination of CNN and LSTM networks on 3D skeletal data sequences obtained from full-body and hand tracking sensors. The CNN training was conducted separately, and in a second stage, the full CNN-LSTM architecture was adjusted accordingly. Similarly, the research described in [100] demonstrates the effectiveness of an LSTM architecture in segmenting hand gestures from raw Leap Motion hand tracking sequences. The segmented gestures are then provided as input either directly to an LSTM network, or indirectly to a CNN architecture, through filters learned by a denoising auto-encoder. Another example of combining convolutional and recurrent architectures was presented in [101], where an end-to-end 3D CNN-LSTM model was trained to recognize gestures in video sequences, achieving close to state-of-the-art accuracy on

[5] http://www-rech.telecom-lille.fr/shrec2017-hand/

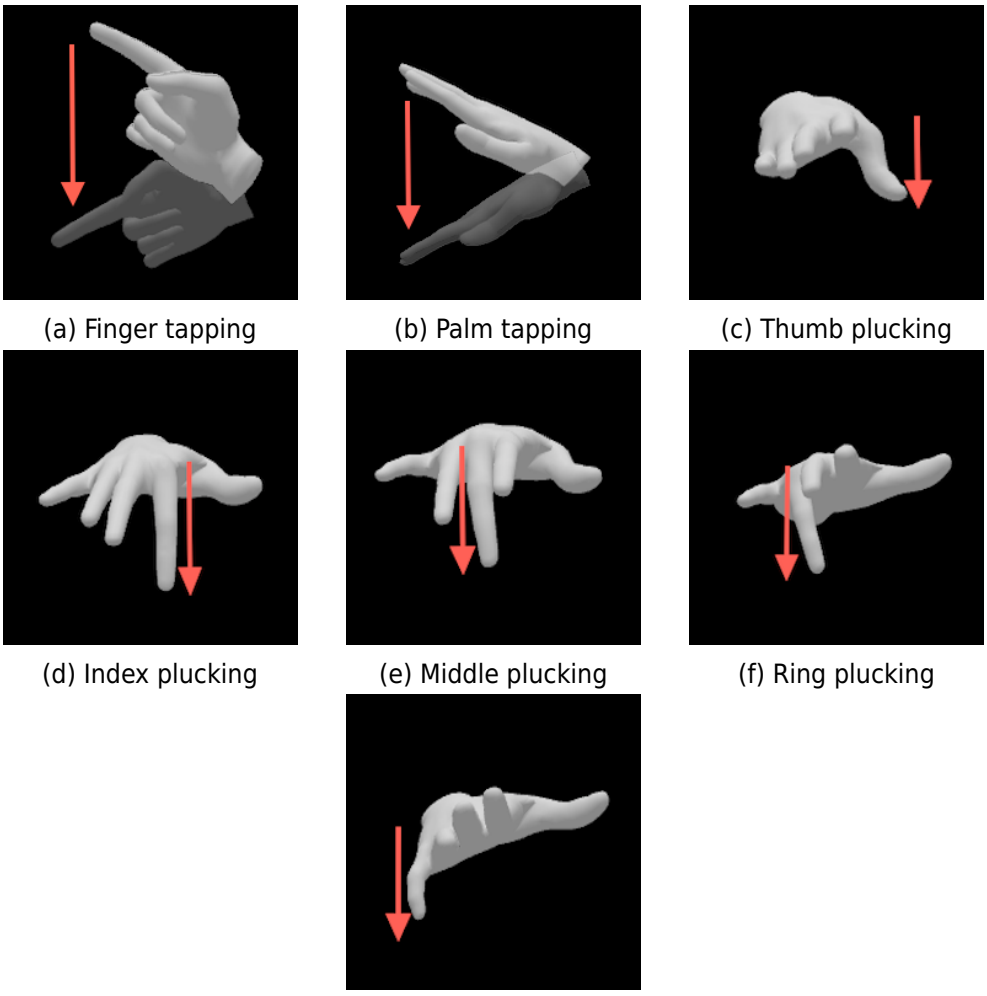
the ChaLearn dataset [102]. To the best of our knowledge, up to the current date there is a lack of similar deep learning approaches in the context of HCI and gesture-enabled virtual musical instruments.

3.4 Methodology

To the best of our knowledge, the gestures required for the intended application context have not been utilized in any previous research study. Furthermore, these gestures are performed without any physical constraints or feedback. In other words, there is no physical mass or tactile information available to convey the momentary evolution of the gesture to the user. Consequently, it is anticipated that different users will perform the same gesture in diverse ways. This highlights the necessity of employing gesture recognition methods based on more sophisticated and complex computational architectures and utilizing multiple examples from various users. To address these requirements, multiple instances of gesture data were collected from multiple users. The dataset was then used to train and test recurrent and convolutional deep neural network for gesture recognition.

3.4.1 Dataset description

In Figure 3.2 we illustrate the targeted hand gestures, specifically focusing on plucking and tapping actions. A description of these gestures can be found in Table 3.1, which also corresponds to the label annotations in the collected dataset. In total, there are 8 gesture classes: 5 related to plucking, 2 related to tapping, and one class labeled as “unknown”. The inclusion of the “unknown” class allows the system to identify instances that do not match any of the targeted gestures. This prevents the computational models from incorrectly categorizing every user action, including cases where no gesture is performed. Regarding the plucking gesture, we consider variations performed by each of the five fingers individually (refer to Figures 3.2c – 3.2g). As for the tapping gesture, we examine both tapping with an open palm (Figure 3.2b) and tapping with the extended index finger (Figure 3.2a). For the “unknown” gesture class, participants were instructed to perform random hand movements that were as dissimilar as possible to the other 7 gestures.



(a) Finger tapping (b) Palm tapping (c) Thumb plucking

(d) Index plucking (e) Middle plucking (f) Ring plucking

(g) Pinky plucking

Figure 3.2: Illustration of the considered instrumental gesture classes corresponding to the right hand. For each gesture, the temporal evolution of the fingers' motion trajectories follows the direction of the arrow.

Table 3.1: Considered gesture classes and their labels.

Gesture Description	Label
Palm tapping	RHPT
Index finger tapping	RHIT
Thumb finger plucking	RHTP
Index finger plucking	RHIP
Middle finger plucking	RHMP
Ring finger plucking	RHRP
Pinky finger plucking	RHPP
Unknown	RHUK

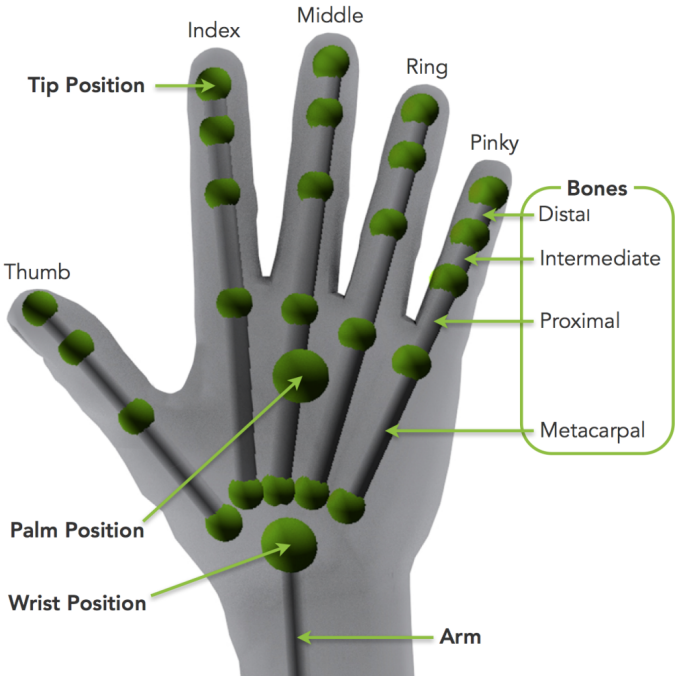


Figure 3.3: Skeletal representation of the hand as it is recognized by the Leap Motion sensor [103].

Data were collected from 10 participants (5 female and 5 male) using the Leap Motion sensor for the 8 gesture classes (of the right hand) described above. Each gesture recording was lasting 2 seconds with an intermediate break of 3 seconds countdown before recording the next sample. The considered dataset includes in total 1019 gesture samples, with 10-15 samples for each gesture class per participant. The recording sampling rate was set to 50 frames per second (FPS). However, there are always expected fluctuations in the actual frame-rate, which can unpredictably vary depending on the CPU power and current CPU processes of the operating system during gesture recording. To compensate for the minor differences in the frame length of each recorded sample, all samples were zero-padded up to a specific number of total frames. The Leap Motion sensor identifies the hands that fall within its capturing field of view (FOV) range and generates a stream of frames, where each frame includes information about the positions and velocities among many other properties of the tracked hands. An example of the hand hierarchy provided by the Leap Motion software development kit (SDK) is presented in Figure 3.3.

All these measurements constitute a set of 186 features, called the

"raw" set of features, which are used to train the considered gesture recognition models, detailed as follows:

- **Positions:** 3D positions of all finger joints (including a zero-length array for describing the missing metacarpal thumb bone), palm center, wrist and elbow (cardinality: 84).
- **Velocities:** 3D velocity vectors for palm center and fingertips (cardinality: 18).
- **Directions:** 3D vectors of directions of hand and fingers, along with the 3D vector of the palm normal (cardinality 21).
- **Miscellaneous:** This feature set includes the arm and palm widths, length of each finger, estimated pinch and grab strengths, finger touch distances and touch zones, 3D position of the palm's sphere radius, stabilized positions for palm and fingers, along with the hand type constant (i.e. left/right) (cardinality: 61).

In addition to the "raw" features, in our initial experiments we also consider a set of 30 handcrafted features for the purpose of comparison, as they are described in [87]. These features consist of:

- 5 values representing the cosine of the angle between each fingertip and the plane defined by the palm.
- 5 values indicating the distances between each fingertip and the center of the palm.
- 5 values representing the distances of each finger from the plane defined by the palm.
- 15 values denoting the 3D positions of each fingertip within the orthonormal axis defined by the hand's direction and normal vectors.

All the aforementioned features are normalized in the range of $[0, 1]$ for each gesture recording. According to Marin *et al.* [87] the length of each finger is normalized based on the length of the middle finger, which is considered the largest one. The collected dataset, named the *Leap Motion Hand Gestures for Interaction with 3D Virtual Music Instruments* (LMHGIf3DVMI), has been released with open-access copyrights [104]. Furthermore, the interface that we developed to record the gesture samples is accessible online[6]

[6] `https://github.com/kosmasK/air-writing-recognition`

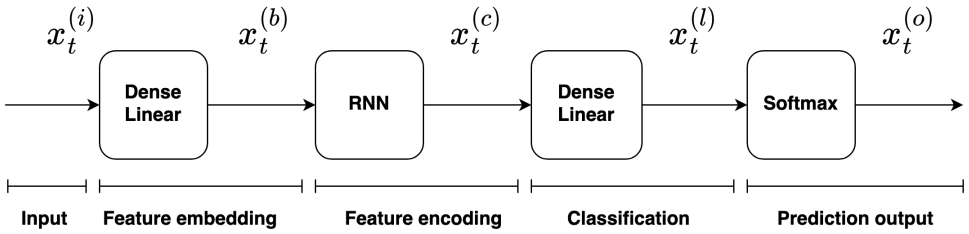


Figure 3.4: Architecture of the employed LSTM-based neural network.

3.4.2 Proposed Computational Models

As mentioned above the proposed systems focus on the classification task of a single window of input data that includes one of the 8 classes. In our initial experiments, we assess the performance of a recurrent computational architecture that employs a dense layer to compute feature embeddings, followed by a LSTM network for capturing temporal feature patterns. Next, we explore the effectiveness of incorporating convolutional filters to model multidimensional data sequences. We propose two computational models that utilize CNNs for the classification task at hand. The first method involves testing the efficacy of 1D CNNs for producing feature embeddings. In this approach, we replace the dense layer in our initial architecture while retaining the LSTM network for sequence modeling. The second method relies entirely on a deep CNN architecture that employs consecutive convolutional and max pooling operations.

Initial LSTM architecture

Our initial methodology at its core employs a LSTM neural network and its architecture is illustrated in Figure 3.4. The input to this network, denoted as $x_t^{(i)}$, consists of a window comprising a sequence of 75 frames. The data within the window can either be in its "raw" form, generated by the Leap Motion sensor (186 dimensions), or it can be represented by the handcrafted features (30 dimensions). All the employed operations and functions (i.e. multiplication, sum and division) are considered to be performed element-wise. The indices (i), (b), (c), (l) and (o) represent the (i)nput, (b)ottom dense linear layer, LSTM (c)lass output, (l)inearly transformed output and (o)utput of the entire network respectively. To process the input vectors, we employ a "Dense linear" layer (i.e. fully connected), which performs a linear

transformation to embed the vectors into a space of fixed dimensionality, as it is given by the following equation:

$$x_t^{(b)} = W_i x_t^{(i)} + b_i \tag{3.1}$$

Where W_i and b_i are the arrays of weights and biases respectively in the bottom "Dense linear" layer. The embedding representation is further encoded using a RNN with LSTM gates as:

$$x_t^{(c)} = \mathcal{F}_{\text{LSTM}}\left(x_t^{(b)}, \vartheta^{(c)}\right) \tag{3.2}$$

Where $\vartheta^{(c)}$ represent the parameters of the LSTM network, defined as a function $\mathcal{F}_{\text{LSTM}}$ of the architecture. Finally, the LSTM output is linearly transformed in a space with dimensionality equal to the number of gesture classes (top "Dense linear" layer) as:

$$x_t^{(l)} = W_o x_t^{(c)} + b_o \tag{3.3}$$

Where W_o and b_o are the weights and biases of the top "Dense linear" layer, the output of which is passed through a softmax function that results to the probabilities of each target class as:

$$x_t^{(o)} = \frac{e^{x_t^{(l)}}}{\sum e^{x_t^{(l)}}} \tag{3.4}$$

The denominator in equation (3.4) represents the sum of all elements in the $e^{x_t^{(l)}}$ (element-wise exponential of $x_t^{(l)}$) array. The recognized gesture class is the one with the highest probability:

$$c_t^{(o)} = \text{argmax}\{x_t^{(o)}\} \tag{3.5}$$

Equation 3.5 returns the index of the $x_t^{(o)}$ vector with the maximum value, corresponding to the number of the one-hot encoded assigned class.

CNN-based Approaches

The proposed computational architectures that incorporate CNNs, leverage on their ability to capture relations in sequences of events and extract meaningful features that increase classification accuracy. In particular, we adopt CNNs in two different architectures. The first deploys a convolution

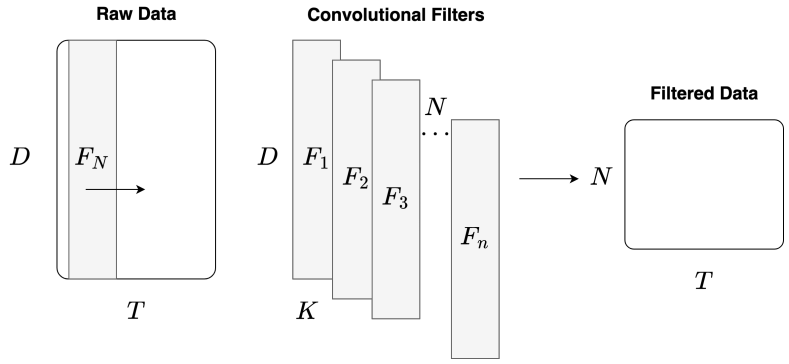


Figure 3.5: Filtering process with 1D-CNN used for automated feature learning from the "raw" input data sequence.

layer for feature extraction that subsequently feed a LSTM network for sequence learning, similar to our initial approach. The second is a deep convolutional architecture with several 1D CNN layers in the feature dimension, followed by max pooling in the time dimension.

In Figure 3.5 we present the 1D CNN filtering process used in both methodologies. The Leap Motion sensor produces sequences of D measurements, corresponding to the "raw" features as described above in Section 3.4.1. Let us denote with T the time window of the input sequence of "raw" data, expressed in number of frames. On each D-dimensional sequence of T frames, we employ N 1D convolutional filters of kernel size K that slides across frames, extracting one feature for each frame using the Rectified Linear Unit (ReLU) activation function. We also apply zero padding at the end of the sequences in order to preserve equal sequence lengths for all gesture samples in our dataset. Thereby, each filter produces a single 1D time series, which are stacked in an $N \times T$ matrix. It should be noted that the ReLU output for each time frame is locally normalized across a neighborhood of l time frames by following the 1D version of the normalization process proposed in the ImageNet model [105], as:

$$b_x^i = a_x^i / \left(k + \alpha \sum_{j=\max(0,i-l/2)}^{\min(T-1,i+l/2)} (a_x^j)^2 \right)^{\beta} \tag{3.6}$$

Where b_x^i and a_x^i correspond to the normalized and non-normalized activation values respectively.

Our first method that incorporates CNNs, referred to as CNN-LSTM, is presented in Figure 3.6. It shares similarities with our initial method, as it

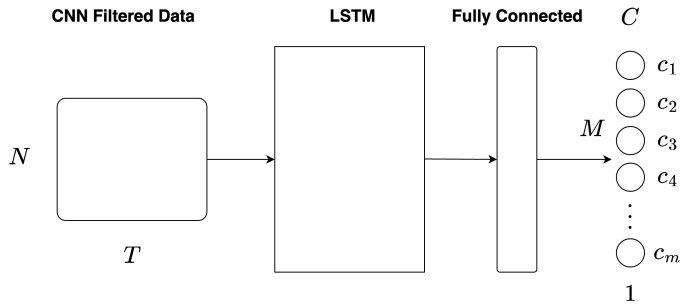


Figure 3.6: CNN-LSTM Method: Feed the CNN-learned features (see Figure 3.5) to a LSTM neural network for sequence learning and with a fully connected layer for classification.

still utilizes a LSTM network to model sequential information. However, the main difference lies in the introduction of a CNN network to compute feature embeddings (i.e. the CNN filtering process depicted in Figure 3.5), replacing the "Dense linear" layer used in the previous approach. The output of the LSTM network is used as input to a fully connected layer with linear activations, enabling classification among the targeted classes. During training, the system learns the parameters of the CNN filters, the parameters of the LSTM model, as well as the weights of the fully connected output layer.

The second method is illustrated in Figure 3.7, and it is referred to as dCNN. This method is based on a deep CNN architecture, with 4-layers applying consecutive 1D convolutional and max pooling operations. The process begins with the CNN-generated feature embedding matrix followed by 4 consecutive "bisects". The output of the fourth layer is stacked (on the time dimension) to form a vector of size $N \times T/16$ which is then processed by a fully connected layer with linear activations for classification. During training, the system learns the parameters of the CNN filters and the weights of the fully connected output layer.

3.5 Initial Ablation Study

In this set of experiments we follow an ablation study, aiming to provide a deeper understanding on the effects of the different data features, embeddings and architectural configurations of the RNN model, on the musical gesture recognition task.

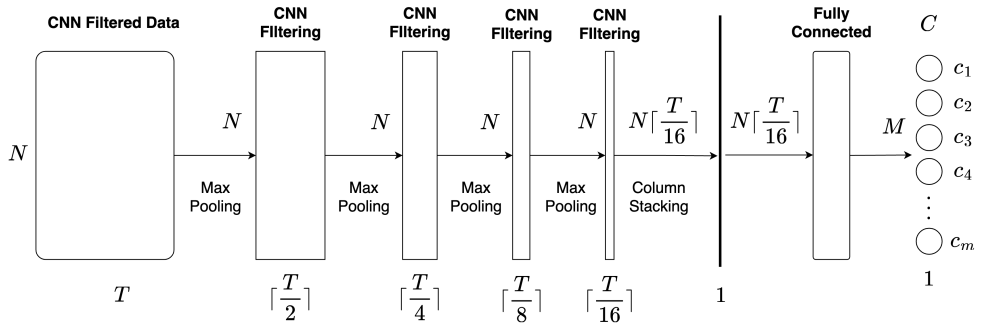


Figure 3.7: dCNN Method: Consecutive application of convolution and max pooling operations followed by a fully connected layer for classification.

3.5.1 Experimental setup

In our initial experiments we consider two types of input data, including (i) "raw" sensorial measurements as provided by the Leap Motion SDK, and (ii) handcrafted features, computed according to [87]. We ensure that all input sequences have a fixed length of 75 time frames (including zero padding). The specific training parameters used in our experiments include a learning rate of 0.001 using the Adaptive Moment estimation (Adam) optimization algorithm [106] to minimize the cross entropy cost function, with $L2$ regularization of weights set to 0.015, a dropout rate of 0.5 applied on the LSTM gates and gradient clipping during back propagation in the range of $[-1, 1]$. These parameter values were determined after conducting several experiments on a small subset of our dataset. We run experiments with 32, 64 and 128 embedding dimensions for all types of input features (see Equation (3.1)), tested various numbers of LSTM cells (32, 64 and 128) and layers (1, 2 and 4), as well as Bidirectional Long Short-Term Memory (BLSTM) gates in our recurrent network.

Overall we apply two different evaluation procedures: (i) the classical 10-fold cross validation in which we randomly select 90% of the samples per participant and gesture class for training, while keeping the rest 10% for testing, and (ii) cross participant validation, where we leave out one participant for testing and include the remaining participants in the training set. We implement all the considered models and experiments with the TensorFlow framework[7] on a computer with Intel Core i7 Central Processing Unit (CPU) at 2.80 GHz and 16 GB random-access memory (RAM) and one

[7] `https://www.tensorflow.org/`

NVIDIA Tesla K40c Graphics Processing Unit (GPU).

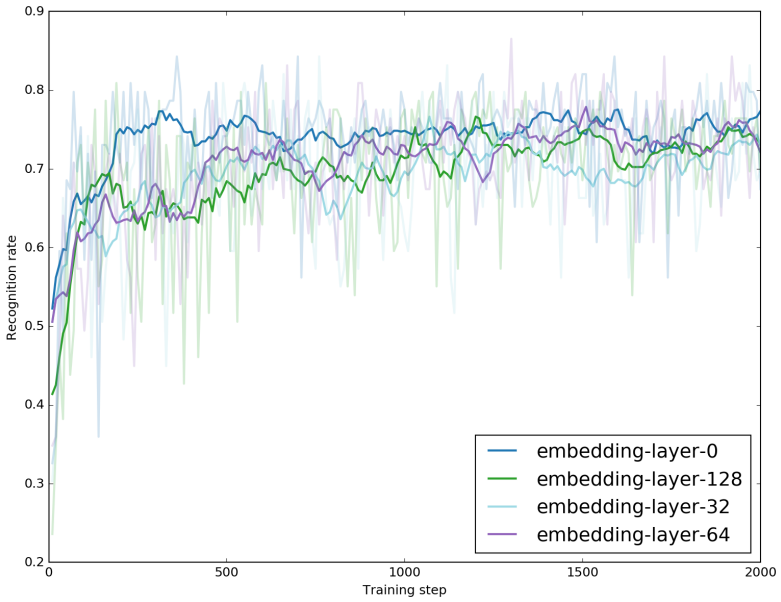
3.5.2 Results

We start our evaluation by using a single LSTM layer with 128 cells and test different sizes of embeddings, compared against the absence of such transformation. Figures 3.8a and 3.8b present the moving averages of the recognition rates on the test samples of participant #7 for 2000 training steps. In the case of "raw" data as input, the introduction of an embedding layer improves the performance, with the embeddings of 32 dimensions reporting the best results. On the other hand, in the case of handcrafted features, it can be seen that adding an embedding layer has a negative impact on the performance of the model.

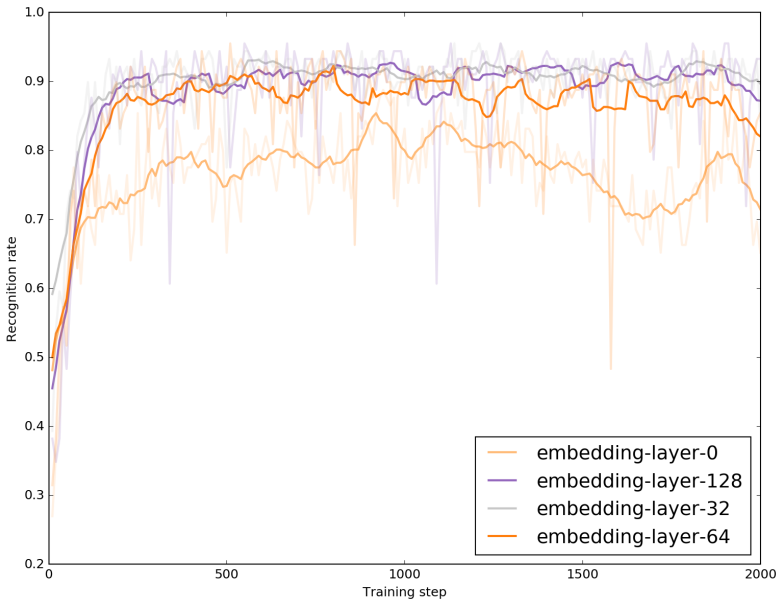
Next we evaluate the effect of different numbers of LSTM cells. In this set of experiments we use an embedding layer of 32 dimensions for the case of "raw" data, while for the handcrafted features we use no embeddings. The results are presented in Figures 3.8c and 3.8d, where we plot the moving averages of the recognition rates on the test set of participant #7 for 2000 training steps. It can be seen that the scenarios that involves 128 LSTM cells report the best results for both types of input data. It is important to be mentioned, that even though we tested greater number of cells, we observed a negative impact on the performance of the system.

Furthermore, we run experiments considering deeper architectures of multiple LSTM layers, as well as models based on BLSTM gates. In Figures 3.8e and 3.8f we plot the moving averages of the recognition rates for 2000 training steps using the two types of input data for fold #2. We can see that the best performance, regarding "raw" features, is reported by the single layered LSTM model, whereas in the case of handcrafted features, the most performant architecture is the one that employs a stack of 4 LSTM layers. Moreover, we observe that the recognition rates regarding "raw" data is greater than the rates reported by the models trained with handcrafted features. At the same time the reported performance variability of the handcrafted features is greater than the corresponding variability of "raw" data, indicating that the proposed architecture follows more stable training process in the case of "raw" features.

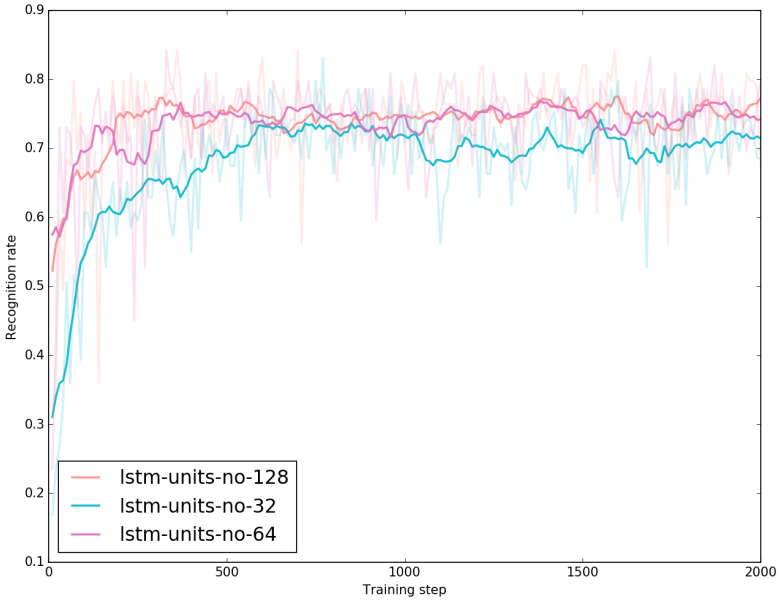
In Tables 3.2 and 3.3 we present the average test recognition rates and



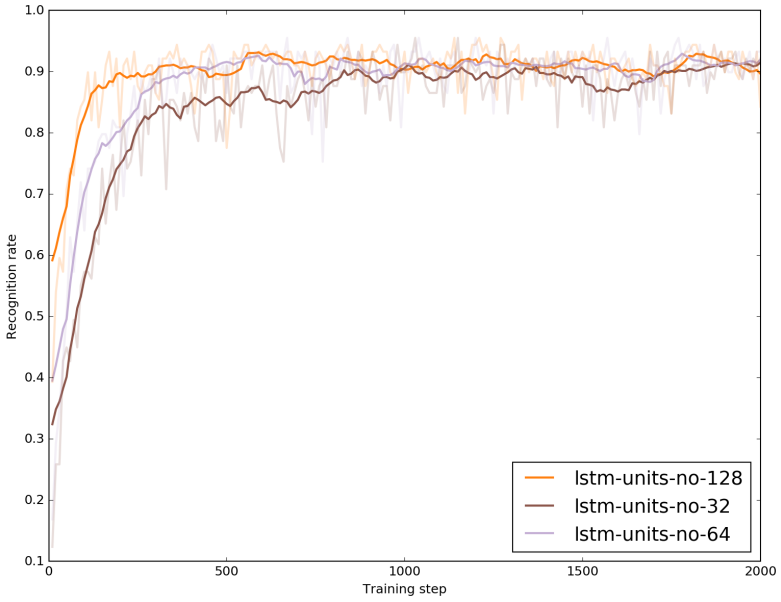
(a) Test accuracies on participant #7, considering various sizes of embedding dimensions with handcrafted features.



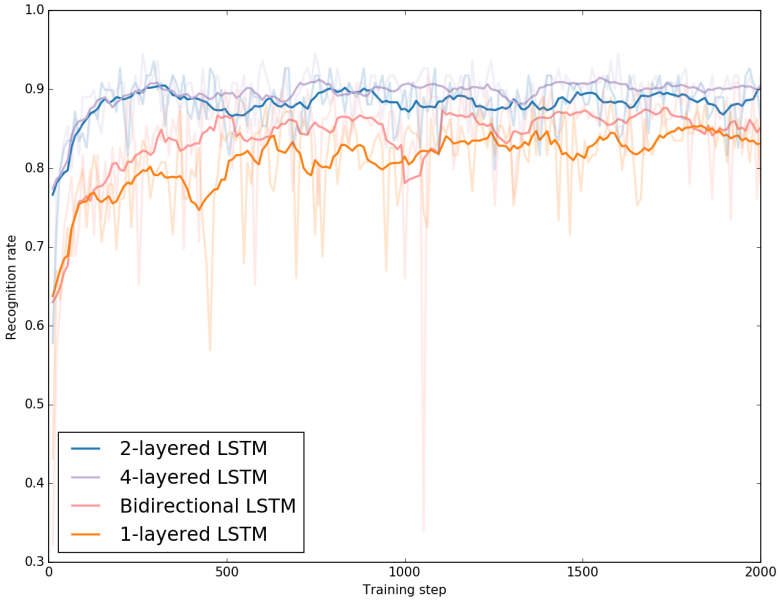
(b) Test accuracies on participant #7, considering various sizes of embedding dimensions with “raw” features.



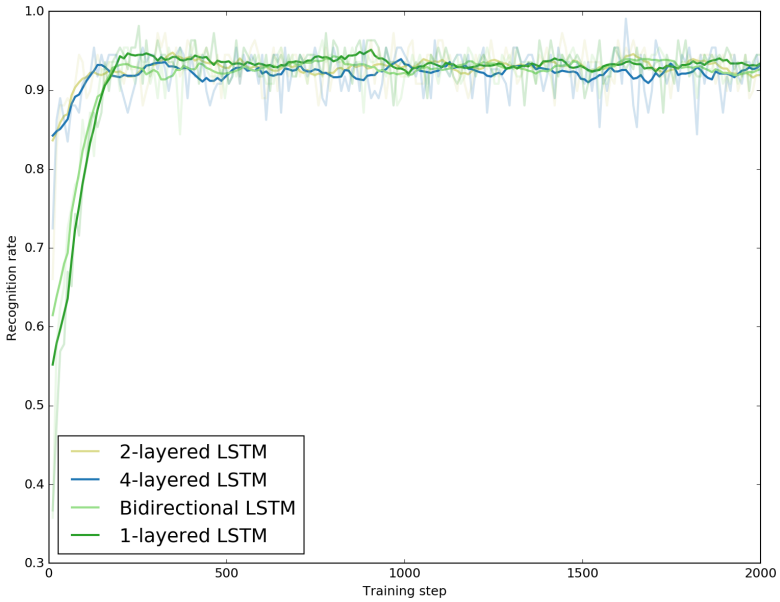
(c) Test accuracies on participant #7, considering various numbers of LSTM cells with handcrafted features.



(d) Test accuracies on participant #7, considering various numbers of LSTM cells with “raw” data.



(e) Test accuracies on fold #2, considering BLSTM gates and various numbers of LSTM layers with handcrafted features.



(f) Test accuracies on fold #2, considering BLSTM gates and various numbers of LSTM layers with “raw” data.

Figure 3.8: Plots of the reported test accuracies of the different experimental scenarios for 2000 training steps.

Table 3.2: Recognition rates for various configurations for 10-fold cross validation.

Configuration	**Recognition Rate** (Average $\pm$ std)
Handcrafted, no embedding, 4-layered 128 LSTM encoding	$90.56\% \pm 2.04\%$
Raw Data, 32 dense embedding, 128 LSTM encoding	$92.62\% \pm 1.82\%$

Table 3.3: Recognition rates for various configurations for participant cross validation.

Configuration	**Recognition Rate** (Average $\pm$ std)
Handcrafted, no embedding, 4-layered 128 LSTM encoding	$63.31\% \pm 4.21\%$
Raw Data, 32 dense embedding, 128 LSTM encoding	$85.50\% \pm 2.36\%$

their standard deviation (STD), for the two types of evaluation protocols (i.e. 10-fold cross validation and participant cross validation), using the two architectures corresponding to the cases of having as input handcrafted features and "raw" data respectively. The averaging has been performed with respect to the various trials on the test sets as well as with respect to the training steps between 800 and 900 steps for the "raw" data and 200 to 300 steps for the handcrafted features. We use different ranges since the two systems converge after different number of training steps. Overall, the architecture that uses an embedding layer and a single LSTM layer on "raw" data reports better recognition rates, than the 4-layered counterpart on handcrafted features. Furthermore, we notice a significant decrease in the average recognition rate during cross participant validation, particularly when using handcrafted features. This observation suggests that the number of users in the training set is not sufficient to produce models that can generalize well and achieve consistent performance regardless of testing user.

In order to address this issue under real-world circumstances, usually a

Table 3.4: Recognition rates of subjects with ids #5 and #7 for different number of gesture samples in the training set.

Number of samples in training set	Participant #5	Participant #7
0	89.55%	71.03%
2	89.81%	85.38%
5	94.29%	85.81%

few gesture samples could be given by the users to "fine-tune" or adapt the system to their own gestural temperament. To this end we consider a set of experiments where we keep one test participant out from the training process, and on a later stage we use few samples of each gesture class of this participant to evaluate the contribution to the recognition rates. To have a better understanding of the impact of the adaptation approach, we select participants #5 and #7 from our dataset to test this scenario, corresponding to the two opposite extremes based on their performance reported in the cross participant evaluation. The results presented in Table 3.4 demonstrate the average accuracies between 800 and 900 training epochs on the test sets of participants #5 and #7. The results indicate that model adaptation with only few samples of user data would lead to better performance. Especially regarding participant #7, the average accuracy increases from 71.03% up to 85.38% when only 2 adaptation gestures are given. Similarly, for participant #5, the average accuracy increases from 89.55% to 94.29% with 5 adaptation gestures. Moreover, during all experiments we measured the average time of our system to produce results on input sequences of 75 time frames, being around $2.8\,\mathrm{ms}$.

3.6 CNN-based Experiments

In this set of experiments we leverage the results from the ablation study, and extend our initial methods towards testing the effectiveness of convolutional operations to (i) provide better embeddings, compared to simple linear projection utilized in our initial models, and (ii) capture the spatio-temporal relations of multidimensional data sequences compared to their recurrent counterpart.

3.6.1 Experimental Setup

In all experimental scenarios evaluated in this campaign, we consider 1D CNNs with $N = 64$ convolutional filters of kernel size $K = 2$ for each layer employed in the various models. As it regards the training parameters that produce the reported results, we apply a learning rate of 0.001 using the Adam optimization algorithm [106] for minimizing the cross entropy cost function, with $L2$ regularization of weights set to 0.015, and gradient clipping in the range of $[-1, 1]$ during back propagation. Furthermore, we employ a dropout rate of 0.5 on the LSTM cells and on the stacked features of the dCNN. Moreover, regarding the local normalization applied after the ReLU activations, we have set $k = 1$, $\alpha = 0.0002$, $\beta = 0.75$ and the neighborhood "radius" equal to $l = 5$. The LSTM network in the CNN-LSTM methodology involves 128 LSTM cells.

In the experiments we employ a 10-fold cross validation scheme, considering only the dataset of "raw" features. We evaluate all the experimental scenarios on a computer with Intel Core i7 CPU at 2.80 GHz, 16 GB RAM and 1 NVIDIA Tesla K40c GPU. For the implementation of the methods we use version 1.3 of the TensorFlow framework. Finally, we use batches of 50 gesture samples and train all models for 1000 epochs.

3.6.2 Results

In Table 3.5 we present the recognition rates and the run times on the testing sets of the 10-folds, using the proposed architectures of CNN-LSTM and dCNN, and compare them against the initially evaluated method based solely on a LSTM architecture. Regarding the CNN-LSTM method, we can see that the replacement of the "Dense linear" layer with a CNN layer provides better feature embeddings that improves by 2.5% the overall performance of the LSTM architecture on the musical gesture classification task. Similarly, the dCNN architecture also presents an improvement of 2.5% in the testing recognition rates compared to the initial LSTM method.

This performance improvement is statistically significant at 5% significance level across all folds, as indicated by a two-sided Wilcoxon rank sum test [107] on the given accuracy distributions, rejecting the null hypothesis that the accuracy distributions between the LSTM, CNN-LSTM (p-value$=0.0232$) and dCNN (p-value$= 0.0373$) come from distributions from the same median. The differences between the CNN-LSTM and dCNN methodologies

Table 3.5: Recognition accuracy and run time of the CNN-LSTM and dCNN methods and comparison with the LSTM method.

	Accuracy			Run time (ms)		
Fold No	**LSTM**	**CNN-LSTM**	**dCNN**	**LSTM**	**CNN-LSTM**	**dCNN**
0	90.91%	92.93%	95.96%	3.253	2.047	0.695
1	89.29%	95.54%	97.32%	2.547	2.475	0.614
2	94.50%	94.50%	94.50%	2.846	2.525	0.741
3	91.07%	93.75%	92.86%	3.179	2.761	0.427
4	94.59%	94.59%	98.20%	2.731	1.797	0.623
5	86.36%	99.09%	91.82%	2.611	1.886	0.735
6	93.69%	93.69%	93.69%	2.803	3.455	0.440
7	92.73%	90.00%	91.82%	2.602	1.852	0.916
8	92.59%	94.44%	95.37%	2.743	4.558	0.453
9	91.96%	94.64%	92.86%	2.547	3.155	0.695
Average	91.77%	94.32%	94.44%	2.786	2.651	0.633
Unbiased STD	2.40%	2.14%	2.12%	0.237	0.831	0.149

are not statistically significant (p-value$= 0.8499$). Furthermore, we report the average run times of the different methods (see Table 3.5), for recognizing one gesture sample of 75 frames. The LSTM and CNN-LSTM methods require approximately the same computational time to provide a single prediction, around $2.7\,\mathrm{ms}$. However, the dCNN method is quite faster, since it requires only $0.6\,\mathrm{ms}$ to respond with a prediction, which is more than 4 times faster than the LSTM and CNN-LSTM methods.

Table 3.6 showcases the confusion matrix obtained from the experiments conducted with the dCNN method on the testing sets of the 10-folds. Upon examination, it can be observed that there is minimal confusion between most gesture classes, except for the “unknown” (RHUK) and “middle finger plucking” (RHMP) gestures, which exhibit a higher number of misclassifications. Specifically, the middle finger plucking (RHMP) is often mistaken for the index finger plucking (RHIP), ring finger plucking (RHRP), and “unknown” (RHUK) gestures. This outcome is somewhat expected since many individuals tend to move their index and ring fingers downward when performing the middle finger plucking gesture. Furthermore, the “unknown” (RHUK) gesture is naturally prone to confusion with other gesture classes as it encompasses various types of gestures, including the finger plucking gestures.

Table 3.6: Confusion matrix results from the proposed dCNN method.

		Predicted							
		RHIP	RHIT	RHMP	RHPP	RHPT	RHRP	RHTP	RHUK
Ground Truth	RHIP	95.77%	0.85%	0.83%	0.00%	0.00%	0.85%	0.00%	1.70%
	RHIT	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	RHMP	2.34%	0.00%	86.80%	0.00%	0.00%	4.66%	0.80%	5.41%
	RHPP	0.00%	0.00%	0.00%	96.19%	0.63%	1.27%	0.00%	1.91%
	RHPT	0.00%	0.83%	0.81%	2.46%	94.15%	0.00%	0.00%	1.75%
	RHRP	0.78%	0.00%	0.00%	1.57%	0.79%	95.26%	0.00%	1.61%
	RHTP	0.00%	0.00%	0.00%	0.00%	0.00%	0.69%	97.93%	1.38%
	RHUK	1.19%	1.88%	1.80%	0.59%	0.59%	3.08%	1.27%	89.60%

3.7 Conclusions

In this Chapter we tested various computational models, based on recurrent and convolutional operations with various settings, on the task of musical gesture recognition. We used the Leap Motion sensor to construct a dataset of 8 hand gestures (7 musical gestures and 1 corresponding to arbitrary movements). According to the initial ablation study, we demonstrated that the computational architecture involving a linear embedding layer and a single LSTM layer, is able to form the basis of a real-time gesture recognition engine, receiving as input directly "raw" sensorial data. As it regards the contribution of handcrafted features [87], we presented that the models that used these features as input, consistently reported the lowest performance in all experimental scenarios. We have also demonstrated that although the system is not user independent, its performance can be improved significantly, by providing only a few gesture samples of the end-user to fine-tune the parameters of the computational model.

Next we evaluated two methods that employ CNNs for the task at hand. The first method leverages the outcome of our ablation study combining a CNN to compute feature embeddings and the LSTM for sequence learning. The second method involves a deep CNN architecture that utilizes consecutive 1D convolutions and max pooling operations. In our experiments, both methods outperformed the initial LSTM-based model. Regarding the CNN-LSTM approach, we showed that by replacing the fully connected layer with a 1D convolutional layer for feature embedding, leads to a substantial improvement in the reported recognition accuracy. Likewise, the dCNN architecture also presented notable improvements in the reported gesture

recognition rates compared to the initial LSTM-based method. However, there is no statistically significant difference in performance between the dCNN and CNN-LSTM methods. It is worth noting that the dCNN method demonstrates a significant reduction in computation time compared to both the LSTM and CNN-LSTM approaches.

CHAPTER 4

Interacting with virtual musical instruments

4.1 Introduction

Research related to motion and gestural analysis have been conducted in various scientific areas including cognitive science, communication theory, linguistics and music. However, in recent years, gestural interaction gained an increasing interest in the HCI research community, due to the reduction of cost and widespread availability of various sensors that allow the acquisition of natural gestures in great detail and in a non-intrusive manner [108]. As a consequence, the trends of 3D User Interfaces and VR re-emerged [109, 110]. Additionally, their importance in the immersion of virtual interaction stems from their ability to enhance the visual feedback by introducing an additional layer of realism [71]. Even thought there are multiple studies that experiment with augmented and virtual musical instruments, most of them are custom-made and platform dependent, while their setup is usually difficult to reproduce [111]. To this end, web technology standards, such as HyperText Markup Language (HTML)5[1], Web Audio[2] and WebGL[3], render modern web browsers as flexible and powerful platforms for designing and utilizing such interactive multimedia applications.

[1] https://www.w3.org/TR/2010/WD-html5-20100624/

[2] https://www.w3.org/TR/webaudio/

[3] https://www.khronos.org/registry/webgl/specs/latest/

Additionally, their goal is to hide the underlying hardware complexity and provide a common framework for developing real-time and platform independent applications.

In this regard, our approach evolves around integrating state-of-the-art web technologies to develop an open-access cross-platform virtual 3D environment, towards enabling users to interact conveniently in real-time with 3D virtual musical instruments. Moreover, the interactive setting discussed in this Chapter forms an integral part of an educational platform focused on STEAM education[4]. Its objective is to assist students in grasping fundamental scientific principles through engaging and interactive music-based activities. Additionally, our proposal promotes the aspect of musical expressiveness and exploration by providing realistic aural feedback based on a physical model-based sound synthesis engine. As a MoCap system we employ the Leap Motion sensor and employ its JavaScript SDK, enabling the deployment of our gesture recognition models presented in Chapter 3. The remainder of this Chapter is organized as follows. In Section 4.2 we present some related studies. Next, in Section 4.3 we describe our methodology and system architecture along with the different tools that we employ in the development of our application. In Section 4.4 we evaluate the interactive system based on usability testing campaigns and analyze our results. We conclude this Chapter in Section 4.5, discussing the limitations and contributions of our approach.

4.2 Related Work

In order to play a musical instrument, the musician interacts continuously with the instrument by performing a set of complex subtle control gestures, which from their side are affected by various proprioceptive sensorial feedback, including the sensorimotor, haptic, visual and auditory stimuli cues [77]. These unique characteristics of musical gestures, usually require multidisciplinary approaches for conducting meaningful research. However, building computational models for the analysis of higher level gestural features still remains a challenging task. Therefore, natural gestures are usually employed for controlling virtual musical instruments in a way of simulating the corresponding sensory information, by approximating the actual

[4] https://workbench.imuscica.eu/

interaction status of real instrumental performances. In this sense, we can specify three main research directions in the context of gesture-enabled HCI as follows [112]:

- *Sound synthesis control*, where the user modifies in real-time fundamental sound synthesis properties of the virtual instrument, such as note pitch, timber and velocity.
- *Score-level control*, where the user alters semantic features of a predefined musical score.
- *Sound processing control*, by means of post-production events, where the user may manipulate the amount of digital audio effects or control the spatialization of sound during a live performance.

However, there are many research studies that experiment with more than one of the aforementioned directions. For instance, NexusUI [113] is a Javascript library aiming to address both sound synthesis and processing functionalities, since it offers various touch-compatible interfaces that can be integrated in web audio applications. Roma *et al.* in their project called "Handwaving" [114] present a participatory musical system that was developed based on web standards. Their approach takes advantage of the built-in accelerometer of smartphones in order to recognize specific gestural patterns and produce sounds in a given musical context. Other proposal employ electromyography sensors, like the Myo Armband[5], for analyzing musical performance gestures [115] or to control sound and light spatialization [116].

Common RGB and depth sensor cameras, such as the Leap Motion[6] and Microsoft Kinect[7], are frequently used in gesture-driven musical projects. The hand tracking capabilities of Leap Motion sensor have been tested in projects regarding expressive real-time sound synthesis [89, 117, 118], as well as modulating digital effects and spatialization of sound in multi-array speakers installations [119]. On the other hand, Microsoft Kinect targets whole body interactions. For instance, Mandanic *et al.* developed the system called "Disembodied Voices" [79] that receives as input motion data from a Kinect sensor, to control articulated events which are performed

[5] https://www.myo.com/

[6] https://www.leapmotion.com/

[7] https://developer.microsoft.com/en-us/windows/kinect

by a virtual choir. Moreover, multiple studies employ whole-body skeletal information to evaluate music conduction gestures, using various ML techniques, such as GMMs [120], multimodal HMMs [121] or DTW [96].

However, there are limited solutions that specifically address the crucial aspects of visual and haptic feedback. For instance, Leonardo *et al.* [122] have conducted experiments involving the development of custom-built haptic controllers. These controllers enable the simulation of realistic touch feedback for instruments such as keyboards and bowed instruments. Berthaut *et al.* in their study presented in [123], aim to bridge the gap between the transparent nature of virtual instruments and the tactile stimuli of physical objects. Their approach involves the use of depth camera sensors as gestural acquisition systems, and projectors to visually enhance physical objects by displaying virtual graphics. When it comes to immersive virtual reality (VR) applications, examples often employ Head-Mounted Displays (HMDs) and marker-based MoCap systems like OptiTrack[8]. These systems enable control of virtual avatars in VR environments with high precision [124].

Upon reviewing the bibliography, it becomes apparent that user-friendly setups for musical virtual instruments are limited. The majority of the existing applications either rely on costly MoCap systems [115, 124] and specialized custom-built controllers [122], or they utilize software tools specific to particular platforms [111]. Nevertheless, there are a few applications that facilitate audio interaction through web-based platforms [113, 114], but they lack of realistic visual feedback.

4.3 Methodology

As mentioned above, the system described herein is deployed as part of a STEAM education platform that promotes the learning of sciences though music in secondary education. In this regard, the novelties of our work are twofold: (i) we enhance the virtual musical instruments with realistic visual and audio feedback; and (ii), we try to provide easy access to virtual musical interaction activities by strictly utilizing web technologies that allow us to run the application in modern web browsers. An overview of our system is depicted in Figure 4.1.

[8] http://optitrack.com/

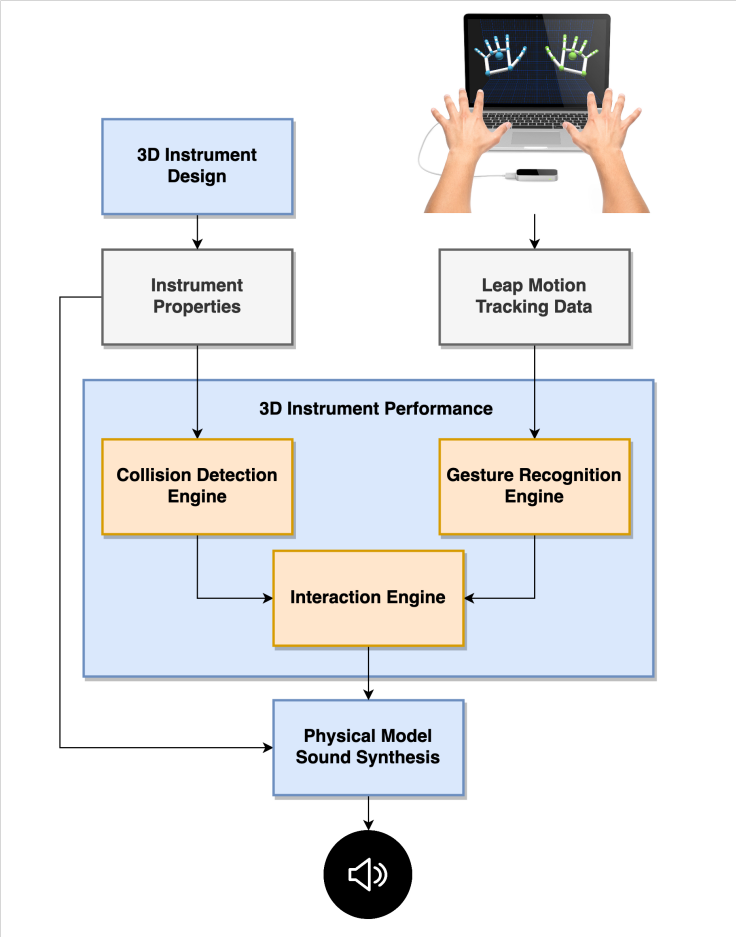


Figure 4.1: General overview of the proposed system architecture.

The architecture comprises three primary components (corresponding to the blue colored boxes in Figure 4.1). The first component aims to provide an environment that enables students to design a 3D virtual musical instrument. Users can modify specific instrument parameters that directly impact its sound characteristics. The second component encompasses a physical model-based sound synthesis engine for the virtual instrument. The physical parameters of the virtual instrument are simulated (although sometimes with simplification) similarly to those found in physical instruments. The third component is a 3D environment where users interact with the virtual instruments using the Leap Motion. This component considers the geometric properties of the designed instrument to compute the corresponding interaction feedback between the captured hand motions and the virtual instrument. Gesture data is processed in real-time and transmitted

to the physical model-based sound synthesis engine, bringing the musical instrument to “life”.

In our study, we focus on two categories of instruments, including string and percussion instruments. Specifically, for string instruments, we examine a monochord instrument with two strings divided by a bridge, resulting in a total of 4 interactive string components. As for percussion instruments, we consider simple drum membranes with circular or square shapes, along with a xylophone. We provide a detailed description of the methodology applied to develop the proposed system and its submodules as follows.

4.3.1 3D Instrument Design

The *3D Instrument Design* environment allows users to create a 3D graphical representation of predefined virtual instruments, even without advanced skills in 3D graphics design. An example of designing a monochord instrument is provided in Figure 4.2. While the underlying modeling software offers various tools for editing 3D objects, it was important to restrict the available functionalities. The 3D modeling environment consists of multiple 3D editing algorithms, providing features like painting, sculpting and parametric design. This limitation prevents users from creating distorted 3D models while maintaining an engaging application experience. Alongside the 3D editing tools, users are encouraged to modify several physically-based modeling parameters of the instrument, such as string material and tension. This additional feature provides opportunities for customization and exploration within the virtual instrument design process.

The modular core engine is written in C++ and it utilizes OpenGL[9], thus enabling a cross-platform deployment. The modeling engine has been already ported to several platforms, including, desktop, mobile and web-based versions. Our system described herein benefits from the web-port of the 3D modeling engine. The transpiling was made with emscripten[10], converting the C++ algorithms to JavaScript code, which can be run by any HTML5-compatible web browser. Beside 3D editing tools, the modeling engine is also enabled to work modern VR HMDs as an embedded application.

[9] `https://www.opengl.org/`
[10] `http://emscripten.org`

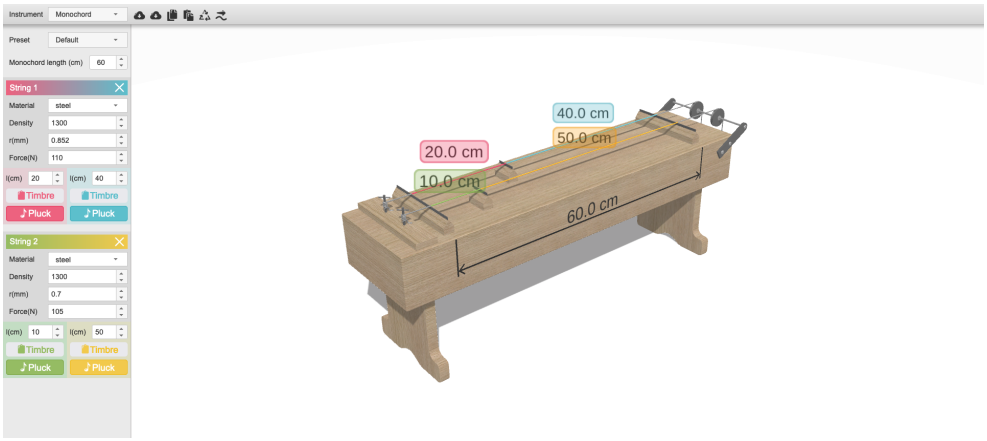


Figure 4.2: Example of the 3D Instrument Design environment, customizing a monochord instrument.

4.3.2 Physical Model-Based Sound Synthesis

In contrast to sound sampling or additive synthesis, physical model-based sound synthesis aims to closely emulate natural aural phenomena. This approach involves representing sounding objects, such as strings, plates, bars and membranes, with physical characteristics like geometry and material properties. These objects are connected and interact with each other through specific actions like striking, tapping, or plucking. The resulting physical system evolves over time according to complex mathematical equations that model the behavior of the objects and their interactions. By leveraging principles from physics, this synthesis technique can produce nuanced and vibrant sounds. While our sound engine primarily relies on a modal computational approach, which describes object resonances as combinations of modes, performing real-time sound synthesis for a virtual musical instrument still requires substantial computational resources. Initially written in C++, the Modalys [125] engine was ported to JavaScript using the emscripten framework, to enable its deployment on a web-based environment. We implemented various optimization strategies to achieve satisfactory instrument performance without noticeable latency or audio artifacts.

4.3.3 3D Instrument Performance

The *3D Instrument Performance* environment encompasses 3 modes of interaction, namely the *Physical Interaction*, *Gesture Interaction*, and *Mixed*

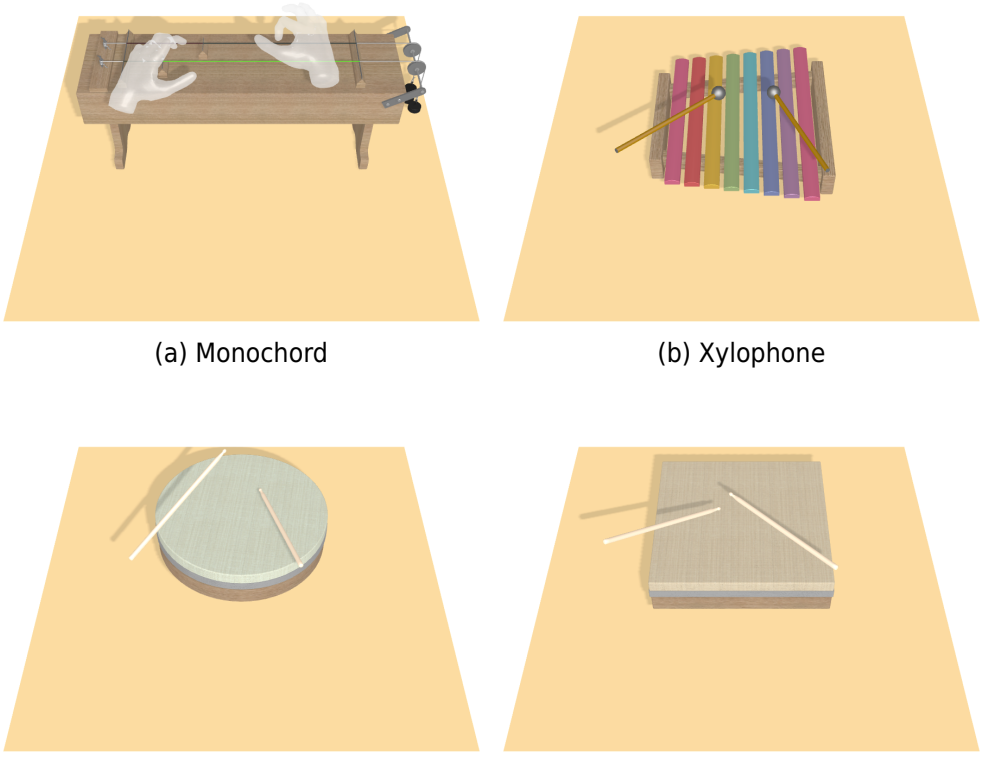
Interaction. These modes are encoded within the architecture depicted in Figure 4.1. The input to this interactive system includes the geometrical parameters of the 3D virtual instrument and the "raw" hand tracking data obtained from the Leap Motion sensor. This information is then processed by 2 key components, including the Collision Detection Engine (CDE) and the Gesture Recognition Engine (GRE). The CDE operates separately from the GRE, and detects whether the hands come into contact with the virtual instrument. Also, it provides additional information such as collision points, speed and direction vectors. On the other hand, the GRE focuses solely on "raw" Leap Motion data and identifies specific gestures performed by the user's hands, as described in Chapter 3.

The Interaction Engine (IE) receives the outputs from the CDE and GRE as input and triggers corresponding messages to the physical model-based sound synthesis engine, according to the underlying virtual instrument type. Especially, in the case of the monochord instrument, we map the movements of the user's physical hands to a set of virtual 3D hands. As it regards the percussion instruments, we map the position and orientation of the user's palm, either to a set of virtual mallets (in the case of xylophone) or drumsticks (in the case of drum membranes). Moreover, we provide a recording feature that allows users to record their hand movements and subsequently review their performances. During playback, the recorded hands appear in the virtual world, resembling the Leap Motion data, and interact with the CDE and GRE accordingly. We developed the *3D Instrument Performance* environment using the Three.js[11] framework, which is a mature and lightweight JavaScript library that offers straightforward WebGL rendering. The various 3D virtual instruments and their respective performance examples are illustrated in Figure 4.3. Detailed descriptions of the 3 modes of interaction are provided as follows.

Physical-based Interaction

When selecting the *Physical Interaction* mode, users engage with the virtual musical instruments as if they were interacting in the physical world with real instruments. In this mode, the IE considers only the output from the CDE. Additionally, the CDE processes the Leap Motion tracking data by representing the hand as a collection of line segments. Each segment

[11] `https://threejs.org/`



(a) Monochord (b) Xylophone

(c) Circular drum membrane (d) Square drum membrane

Figure 4.3: Examples of musical interaction with the considered 3D virtual instruments.

corresponds to a different finger, following the anatomical structure of the human hand and adhering to the specifications of the Leap Motion device. Specifically, each finger is represented by four continuous line segments, except for the thumbs, which consist of three segments. This representation is depicted in Figure 3.3.

Similarly, musical instruments are represented in a simplified manner, focusing solely on the interactive components rather than the entire 3D mesh of the virtual instrument. Specifically, in the case of a stringed instrument, each string is represented as a line segment, while drum membranes and xylophone bars are simulated as circular or square surfaces based on their shape. Within the CDE, computational algorithms are employed to detect and report collision events between these geometric shapes. For instance, when dealing with the monochord instrument featuring four strings, the CDE identifies collisions between the line segments representing the

fingers and the four line segments representing the instrument's strings. Based on the collision results, the IE sends appropriate messages to the physical model-based sound synthesis engine, triggering the corresponding sound synthesis processes.

Gesture-based Interaction

In contrast to the *Physical Interaction*, the *Gesture Interaction* mode disregards the 3D model of the virtual instrument and omits the CDE, while relying solely on hand gestures to perform actions on the virtual instrument. In this mode, the position of the hands in the 3D scene does not actively influence the interaction with the virtual instrument. Instead, their visibility within the Leap Motion FOV is only required, in order to use the "raw" tracking data to feed the gesture recognition model, by following the sliding window process described in Chapter 3. We employed the best dCNN computational model, according to the experimental results presented in Section 3.6. Our implementation is based on the TensorFlow.js[12], version 1.4.0, to deploy the trained model on the user's browser (i.e. client-side) and infer recognition results faster, avoiding possible network latency. In the case of the monochord, we consider only the finger plucking gestures (see Figure 3.2) as valid user actions to trigger musical events, while for the percussion instruments we rely on the finger and palm tapping gestures. When the corresponding gesture is recognized, the IE sends the appropriate message to the physical model-based sound synthesis engine, using predefined values for the collision points and velocity vectors.

Mixed Interaction

In the *Mixed Interaction*, the IE considers both the outputs of the CDE and the GRE. It can be considered similar to the *Physical Interaction*, but with the restriction that a gesture must be performed in order to trigger the interactive component of the underlying instrument. For instance, if a finger plucking gesture is detected by the GRE, then the results of the CDE that correspond solely to that finger are considered, and only if that finger collides with a string, then the IE sends the appropriate message to the sound synthesis engine. It is important to highlight that this mode of interaction was developed in a later stage to compensate the limitations presented by

[12] `https://www.tensorflow.org/js`

the physical and gesture-based interactions, according to the results of our usability evaluation campaign, described as follows.

4.4 Usability Testing

The evaluation of the 3D performance environment was conducted through usability tests with students aged between 13 and 16. These tests took place in schools located in Belgium (11 students), France (10 students), and Greece (29 students). Participants were given a usability scenario and a questionnaire, while the researchers conducting the tests engaged in informal conversations with the students and took note of any remarks that were not captured by the questionnaires. The usability scenario presented to the participants involved performing tasks and playing music using the various tools provided in our STEAM platform. The goal was to encourage participants to explore different aspects of the GUI and the functionality of the *3D Instrument Performance* environment, with the aim of uncovering any weaknesses or misinterpretations related to the overall functionality of the system. The questionnaires included questions around whether the user had an enjoying experience, if the GUI elements of the environment were easily accessible and intuitive, and whether the setup was audio-visually pleasing.

In summary, the usability scenario provided a step-by-step guide for users to select instruments and explore different modes of interaction at various stages. The main focus of the evaluation was to assess the physical and gesture-based interaction modes. Each participant was handed a translated version of the scenario in their native language, and the coordinating researcher provided instructions for each subsequent step. Once all the tasks in the usability scenario were completed, the participant filled out a questionnaire, including the following questions:

- **Q1:** How familiar are you with performing a musical instrument? (*Likert scale*: **0:** I don't play any instrument, **1:** I consider myself a musician)
- **Q2:** How would you rate the overall usability of the interface? (*Likert scale*: **0:** Not usable at all, **1:** Very easy to use)
- **Q3:** How would you rate the physical interaction mode with the instrument?

Table 4.1: Compressed representation of user answers in the questionnaires (total of 50 participants) in a normalized rage of $[0, 1]$.

Question	Mean	Std	Skewness	Kurtosis
Q1	0.42	0.32	0.52	-0.75
Q2	0.73	0.22	-0.42	-0.58
Q3	0.68	0.24	-0.35	-0.64
Q4	0.71	0.24	-0.98	1.19
Q5	0.55	0.28	-0.38	-0.58
Q6	0.62	0.29	-0.21	-1.03
Q7	0.73	0.36	-0.94	-0.35

(*Likert scale*: **0:** Very difficult, **1:** Very easy)

- **Q4:** How would you rate the realism of the produced sound when performing the virtual instrument?
  (*Likert scale*: **0:** Very poor, **1:** Excellent)
- **Q5:** How would you rate the responsiveness of the instrument?
  (*Likert scale*: **0:** Very poor, **1:** Excellent)
- **Q6:** How would you rate the gesture based interaction mode with the instrument?
  (*Likert scale*: **0:** Not good at all, **1:** Very good)
- **Q7:** Was it easy to record the gestures of the performance?
  (*Multiple choice*: **0:** No, **0.5:** Not sure, **1:** Yes)

Question 7 of the questionnaire utilize multiple-choice options, while the remaining questions employ a Likert scale ranging from 1 to 5. The descriptions provided in the aforementioned list correspond to the numeric values used in the "compressed" representation of the results found in Table 4.1. In this representation, all answers are encoded as normalized values ranging from 0 to 1. For the multiple-choice question, numeric values were assigned to each possible answer, so that negative responses were given a value of 0, positive responses were assigned a value of 1, and intermediate responses were assigned a value of 0.5. Regarding the Likert scale responses, they were normalized by linearly transforming the original range of $[1, 5]$ to the range of $[0to1]$.

In Table 4.1 we present the mean values (0 and 1 stand for the negative and positive ends), the standard deviation (which indicates the agreement among participants), the skewness (positive values show a skew towards

the negative end and vice versa) and the kurtosis (greater values indicate sharper edges towards the mean). By analyzing the collected answers, we observe that the participants provided positive views on the general usability and appearance of the environment (Q2). In terms of the physical-based interaction mode with the virtual instrument, the participants held moderate to positive opinions (Q3). Although, they encountered difficulties in understanding how to perform the gesture-based interactions, yet their answers were comparable for both interaction modes (Q6). Despite the challenges, users found gesture interaction more interesting based on written feedback, provided as optional comments to the researchers conducting the tests. Most students rated positively the sound realism (Q4) of the virtual instruments, but there were differing opinions regarding the "responsiveness" of the environment (Q5). This discrepancy could be attributed to variations in hardware setups used in the different locations where the usability testings took place, such as differences in CPU, GPU and memory resources which can significantly impact the performance of the environment. Regarding the task of recording the musical performance and reviewing the interactive session (Q7), students generally found it relatively easy to accomplish.

4.5 Conclusions

In this Chapter we presented a web-based system that utilizes the Leap Motion sensor for interacting in real-time with virtual musical instruments within a 3D environment. Compared to the existing methods, our approach (i) enhances the virtual musical instruments with realistic visual and audio feedback, and (ii) facilitates accessibility to virtual musical interaction activities by employing state-of-the-art web technologies that enable cross-platform deployments. The architecture of our proposal consist of three core modules, including (i) an instrument design environment where the user is able to alter physical features of various instrument models, (ii) a physical model-based sound synthesis engine that produces realistic sound, and (iii) a musical performance environment where the virtual instruments can be performed according to various interaction modes. We validate our system based on subjective evaluations involving students from three different EU countries. The collected answers provided the basis to develop the mixed interaction mode, aiming to address the technical drawbacks of the phys-

ical and gesture-based interaction modes, as reported by the participants. Also, their answers regarding the overall usability, audio realism and design of our system were positive. However, we observed a discord between the provided answers regarding the responsiveness of our system, which can be ascribed to the variability of the underlying computer hardware employed during the usability tests. Since the architectural design of our system follows a client-side approach, its performance is heavily affected by the underlying hardware.

CHAPTER 5

Automatic Dance Motion Generation

5.1 Introduction

Humans perform a large set of different movements in every-day activities such as jumping, running, walking, dancing etc. Furthermore, human motion is affected by various subjective parameters and complex cognitive mechanisms, including the auditory perception, physical conditions as well as the cultural background [126]. Especially, the ability to dance by performing motion patterns that follow a musical composition, is an inherent and fundamental human characteristic. Dancing is a universal, symbolic body language, mostly working as a medium of artistic expression to convey important emotional flows [127], considering that people often dance to music as a form of cultural and religious ritual, or as a recreational activity in social occasions [128].

From a scientific perspective, human motion generation, although less advanced than other generative tasks such as text-to-speech (TTS) synthesis [129, 130] and automatic music generation [131, 132], is an equally important applicative field of study. For instance, generating artificial skeletal motion sequences enables data augmentation, which in turn improves the performance of computational models in most scientific areas related to human motion analysis [133]. Particularly automatic dance motion synthesis methods have great potential in computer graphics and multimedia

industry as an artistic tool for virtual avatar animations [134, 135] or for computer-aided dance teaching [136, 137].

From a technical perspective, modelling dance motion sequences is even more challenging because it requires the ability to creatively synthesize original and continuous motion patterns with high long-term spatio-temporal complexity that reflects the non-linear relationship with the accompanying musical content [138]. In other words, dancing inherently constitutes a multimodal process where a dancing pose at any moment can be followed by a plethora of possible movements, while it requires to be properly aligned with the given musical style and context [139]. In order to construct a representative multimodal dance motion dataset, one option is to obtain 3D skeletal information by utilizing MoCap systems and recruiting professional artists [139–141]. However, this method is usually costly and time-consuming. An alternative approach is to employ automatic pose estimation frameworks, such as OpenPose [142], for extracting skeletal keypoint sequences from online videos. Although this method is cheaper, it depends heavily on the accuracy of the underlying pose estimation framework, which inevitably requires an additional post-processing stage in order to correct the estimated poses on a frame-by-frame basis and construct training data of acceptable quality [138, 143–145].

Most of the proposed methods in the literature tackle the challenges of automatic dance motion generation by implementing RNN for modelling the temporal correlation of skeletal poses with musical information and generating novel motion sequences [138, 140, 144–151]. Nevertheless, RNNs are computationally cumbersome and inefficient for modelling very long sequences, since the error accumulation in the predicted pose sequences allows synthesis over a limited range of future time-steps [152]. Furthermore, it was demonstrated that RNN-based approaches disregard some specific motion characteristics when they are trained with simple reconstruction losses [153]. To overcome both issues, some studies treated dance synthesis as a retrieval task from predefined choreographic units, which however limits the creativity of the generated dance sequences [149]. Other studies implemented GANs and proposed various types of musical and motion feature correlation discriminators [138, 150, 151] for supervising the training process. Yet, adversarial methods suffer from the so-called “Mode Collapse” problem, complicating the generalization of the model by introducing further instabilities during training [154]. In terms of computational

efficiency, CNNs present various advantages over RNNs, considering that convolutional operations do not depend on computations performed in previous time-steps, thus providing faster inference while allowing the parallelization of the training process [155]. To the best of our knowledge, there is only a single proposal exploiting the utilization of CNNs for the task at hand [143], but its architectural efficacy remains questionable due to defective implementation details which do not allow reproducible results.

In this Chapter we try to address the limitations of RNN-based frameworks and propose an autoregressive dilated causal CNN with highway gating functions [156], for automatically generating 2D skeletal dance motion sequences conditioned on audio and skeletal information. The proposed deep multimodal architecture benefits from the stacked dilated convolutional operations for capturing the long-term spatio-temporal context of dance sequences, thus allowing synthesis for any given length. Furthermore, by employing an attention mechanism we fuse the latent representations of past skeletal poses and audio features, in order to stochastically generate novel, variable and complex motion patters, enhancing the overall creativity of our system. The main contributions of our study presented herein are summarized as follows:

- We propose a multimodal deep CNN architecture capable of generating novel dance motion sequences of arbitrary length.
- We propose an autoregressive curriculum learning scheme for self-supervised multimodal feature learning, to deal with prediction error accumulation during the synthesis process.
- We provide explanatory visualizations of the proposed attention-based feature fusion that contribute to the understanding of our method.
- We conduct 2 different user-studies for the subjective evaluation of the effectiveness of both unimodal and multimodal setups for the realistic synthesis of expressive dance motion sequences.
- We validate the data collection methodology proposed in the literature and release a novel dataset containing paired music and 2D skeletal pose sequences along with the source code of the considered experiments, promoting the reproducibility of our proposal[1].

[1] https://github.com/kosmasK/multimodal-dance-generation

The rest of this Chapter is organized as follows. In Section 5.2 we provide a detailed overview of closely related studies. The employed methodology for collecting our dataset, as well as the proposed computational architectures are described in Section 5.3. Next, in Section 5.4 we introduce the various unimodal experimental scenarios and discuss their results. Similarly, in Section 5.5 we thoroughly evaluate the proposed multimodal architecture over quantitative, qualitative and subjective experiments and present our results, in addition to explainable visualizations for interpreting the effect of the attention mechanism to the outcome of the generated sequences. Section 5.6 summarizes the contributions of our study described in this Chapter.

5.2 Related Work

Research in human motion related problems, such as motion segmentation [157] and classification [84, 158], has greatly benefited from modern MoCap technology. Furthermore, the advent of publicly available image datasets with annotated 2D human body joints [142, 159, 160] and their correspondence to 3D human shapes [161–163], has paved the way for extensive improvements in the accuracy of automatic human pose estimation methods.

During the last decade, the advances in leveraging the computational effectiveness of modern GPUs, have allowed deep neural networks to be employed in modelling sequential information; a trend that has also been exploited in several frameworks for human motion prediction. For instance, the authors in [164] proposed an autoregressive generative model based on a RNN encoder-decoder architecture that predicts human body poses frame-by-frame. More recently, deep neural networks have been successfully deployed in expressive motion generation tasks that correlate multiple modalities. For instance, the system presented in [165] generates skeletal motion sequences of a pianist playing the keyboard instrument by using Musical Instrument Digital Interface (MIDI) notes as an input stream. In a similar fashion, the authors in [166] proposed a LSTM RNN framework for generating sequences of arm and hand keypoints of either a pianist or a violinist, according to the given audio input. Consequently, feature fusion methods for correlating various modalities are usually employed when working with multimodal data.

According to [167], there are two main fusion approaches, including:

- Multimodal approaches that merge the initial modalities without necessarily providing a bidirectional mapping of the initial representation spaces to the new representation space.
- Crossmodal approaches that focus on bidirectional correlation of the initial modalities, in other words such methods are trained to directly map one modality to another.

In this regard, dance motion is also considered to be a highly expressive type of human motion and a complex cognitive process that involves multiple perceptual stimuli. In the literature, various research attempts have been proposed to tackle the multimodal challenges of dance motion generation. For instance, an audio beat tracking algorithm based on CNNs has been embedded as a Nao robot application, where the dance moves of predefined choreographies are synchronized with the output of a proposed automatic beat tracking computational model [168]. Another rhythm-based proposal was presented in [169], where the authors developed an audio-driven LSTM architecture for generating new step charts for the popular "Dance Dance Revolution" video game. An interesting multimodal approach for conditioning the generation of dance movements was presented in [170], where the authors used Laban Movement Analysis (LMA) in order to extract and annotate basic movement emotions, which were used to in-paint predefined dance movements.

Nevertheless, audio-driven dance motion-generation has been widely exploited as either a multimodal or crossmodal sequence-to-sequence (seq2seq) problem, correlating audio and musical features with skeletal poses. In Table 5.1 we summarize all the audio-driven dance generation proposals related to the study at hand.

Earlier contributions in this field employed MoCap technologies to record 3D dance motion sequences and construct training data. Inspired by the unimodal Chor-rnn architecture [146], the authors in [147, 148] considered dance synthesis as a multimodal 3D skeletal-based motion synthesis problem conditioned on audio spectral features and past information, by employing RNNs in conjunction with Mixture Density Networks (MDNs) for controlling the variability of the generated dance sequences. A crossmodal approach for learning the mapping from music information to mo-

Table 5.1: Summary of automatic audio-driven dance generation studies.

Framework	Pose	Model Architecture	Synthesis	Genres	Duration (seconds)	Dataset Availability	Source code Availability
Wallace *et al.* [148]	3D	MDN-LSTM	multimodal	Contemporary	3240	✗	✗
GrooveNet [171]	3D	FCRBM	multimodal	EDM	1380	✓	✗
Tang *et al.* [140]	3D	LSTM autoencoder	crossmodal	Cha-cha, Tango, Rumba, Waltz	5640	✓	✗
ChoreoNet [149]	3D	CNN-GRU autoencoder	crossmodal	Cha-cha, Tango, Rumba, Waltz	5640	✗	✗
Ahn *et al.* [141]	3D	1D CNN VGG-like autoencoder, dilated causal CNN generator	multimodal	Cha-cha, Tango, Rumba, Waltz	5640	✓	✗
Lee *et al.* [143]	2D	dilated causal CNN autoencoder	multimodal	K-pop	22536	✗	✗
Dancing2Music [138]	2D	GRU-VAE GAN	crossmodal	Ballet, Zumba, Hip-Hop	255600	✓	✓
Qi *et al.* [144]	2D	LSTM-SA	multimodal	Jazz	4000	✓	✗
Ren *et al.* [150]	2D	GRU GAN	crossmodal	K-pop, Ballet, Popping	18740	✓	✓
Dance Revolution[145]	2D	Transformer-LSTM	crossmodal	J-pop, Ballet, Hip-Hop	43200	✓	✓
Li *et al.* [172]	3D from 2D	Transformer	multimodal	Street dance	180120	✗	✓
AIST++ [173]	3D from 2D	Transformer	multimodal	3 Old-school, 7 New-school	18694	✓	✓
DeepDance [151]	3D from 2D	CNN-LSTM GAN	multimodal	Various	18000	✓	✓

tion sequences in an unsupervised manner was proposed in [171] with the GrooveNet model, which uses a combination of Factored Conditional Restricted Boltzmann Machines (FCRBM) and RNNs to synthesize dance movements for a given audio input. However, due to the small amount of training data employed in their study (only 4 pairs of music and MoCap recordings), the model could not generalize well to music tracks beyond the training data. To this end, the authors in [140] released a dataset of pairs of music and MoCap recordings, representing 4 different types of dance genres with total duration of 94 minutes, and proposed a music-oriented dance choreography synthesis system based on a LSTM autoencoder architecture for mapping acoustic features to motion sequences. The same dataset was also employed in [141], where the authors proposed a multimodal system that first determines the genre of the input music based on a Visual Geometry Group (VGG)-like [174] autoencoder and then chooses a pose generator for the corresponding genre in order to synthesize new pose sequences, based on a dilated causal CNN architecture. The main drawback of this approach is that the pose generator must be trained separately for each genre. The same dance genres were studied in the ChoreoNet [149] framework, with the difference that dance motion sequences were divided in smaller groups of poses that formed the so-called Choreographic Action Units (CAUs). On the fist stage these pre-recorded CAUs along with their corresponding audio features were used to train a CNN-Gated Recurrent Unit (GRU) autoencoder that learns how to map a given song to sequences of CAUs, while on the second stage they trained a U-net [175] architecture for blending the generated CAUs of the first stage, so as to have smoother transitions between the individual units.

Due to the great amount of freely accessible dance videos, mostly hosted on platforms such as YouTube[2], multiple studies utilized automatic pose estimation methods to extract 2D skeletal poses from such videos and construct training data. The system presented in [144] proposed an autoregressive multimodal autoencoder based on two LSTM encoders for the two unimodal inputs (skeletal features and music features), which are fused in the decoder with a self-attention mechanism. Similarly, the authors in [143] proposed a multimodal convolutional autoencoder for synthesizing original dance sequences, conditioned on the mel-spectrogram of an input song. A

[2] https://www.youtube.com/

crossmodal GAN based on variational autoencoders (VAEs) was presented in Dancing2Music framework [138] that proposed the so called "synthesis-by-analysis" learning approach. The main idea of this proposal is to capture the beat and style of the given music and generate a 2D dance unit sequence, which is then passed to a beat warping post-processing module to render the output choreography. This study is considered an important milestone and other studies in the literature usually compare their results against it. Another interesting GAN-based crossmodal architecture was presented in [150], implementing an audio encoder with a GRU network for constructing latent features, which are passed to a feed-forward network in order to generate the corresponding skeletal poses. A similar approach was proposed in DeepDance [151], however its main difference lies in the employed dataset, since the authors applied a pre-trained framework for retrieving 3D poses from 2D skeletal keypoints.

A line of studies have applied the Transformer architecture [176] to the audio-driven dance motion generation task, inspired by its effectiveness in cross-modal problems such as language translation [177–180]. For instance, the authors in [145] employed a Transformer-based audio encoder for extracting latent features which are then passed to an LSTM-based pose generator for synthesizing new dance sequences. The Two-Stream Motion Transformer model was presented in [172], implementing two individual Transformer encoders for calculating the pose and audio latent features followed by a late fusion of the two streams for predicting the pose of the next time-step. However, the late fusion strategy actually decouples the joint feature learning, resulting to unrealistic motions. In order to improve the quality of the generated dance motion sequences the authors in [173] proposed the Full-Attention crossmodal Transformer model that implements a full attention decoder in order to form a joint latent representation from the two Transformer-based encoded modalities. Also, during training they employed a form of curriculum learning that further enhanced the ability of the model to generate more realist 3D dance sequences compared to [172]. However, this model was trained with clips of audio-dance pairs with a duration of around 10 seconds each, thus limiting its ability to generate longer sequences.

To the best of our knowledge, most studies in the literature utilize RNNs in order to model the spatio-temporal context of dance motion. Although most of these proposals achieve promising results in motion sequence gen-

eration, there are still various limitations. RNN-based approaches are not computational efficient for very long sequences [152]; thus focusing mainly on generating motion sequences of short duration (less than a second), which is not optimal for capturing the expressive context of dance motion. Longer desired sequence duration tends to accumulate errors that result to either stagnant or unrealistic generated motion sequences [153, 172, 181]. The only proposal that uses solely CNNs for modelling the spatio-temporal information of dance motion is [143], which theoretically is capable to generate dance motion sequences of arbitrary length, but it lacks of reproducible results. On the contrary our proposal tries to bridge this gap and work as a baseline for generative CNN-based sequential architectures.

5.3 Materials and Methods

In this section we describe the employed methodology for collecting the appropriate videos from crowd-sourced online video hosting platforms, along with the pre-processing and post-processing procedures, which are necessary for extracting paired skeletal and audio features. Next we describe an initially performed ablation study, that helped us to conclude to the considered computational models that were designed with the subjective of predicting the skeletal pose of the sequential future time-step, based on a temporal sliding window approach of past context. The reader can find the source code of our implementation along with the collected dataset online[3].

5.3.1 Dataset

The fastest and most cost-effective method for constructing skeletal training samples is to apply automatic pose estimation frameworks on videos collected from online platforms. Our study requires paired audio and skeletal features which are already available from previous contributions, however most of them were designed based on proprietary protocols that limit their adaptation to our study. For instance the duration of the video clips in the datasets released in [138] and [150] are only 6 and 5 seconds respectively, thus rendering them inefficient for long-term synthesis.

In this regard, we decided to validate the methodology proposed in the literature and construct a new dataset for the study at hand that would

[3] https://github.com/kosmasK/multimodal-dance-generation

meet our requirements. We extracted human-based skeletal poses by utilizing the OpenPose [142] framework to a set of videos downloaded from YouTube. We collected 100 solo K-pop choreographies that contained only a single dancer in the camera frame, with duration of around 3 minutes each and a frame rate of 30 FPS. Due to body occlusions and variable light intensity in the videos, multiple skeletal poses contained unrecognized keypoints, which were rectified by applying linear interpolation between neighboring frames. For organizing the computed skeletal keypoints we used the *BODY25* representation format[4] and kept the 2D coordinates of the first 15 keypoints. Next we normalized the skeletal coordinates by applying Min-Max normalization, defined by the bounding box of the overall skeleton motion in each video separately. In order to maintain the spatial context of the skeletal poses, we took into account the aspect ratio of each bounding box in the normalization formula. In addition to the 15 keypoints, we computed 14 limb lengths by calculating the euclidean distance between the corresponding keypoint coordinates that constitute each limb, similar to [143]. Thus, each skeletal pose in our dataset is represented by a 44-dimensional vector of skeletal features in the continuous space.

Since, in this work, we want to capture the multimodal correlation between the audio stimuli and dancing motions, we separate the audio channel from the video stream. We use the Librosa [182] audio and music analysis library to load the audio files with a sampling rate of 22050 Hz and compute the mel-spectrogram $S \in \mathbb{R}^{T \times M}$ with a Fast Fourier Transform (FFT) window of 1024 samples and a mel-filter bank of 80 bins. To put all resulting magnitude values into a positive range we add 1 before computing the logarithm [183]. Also, in order to obtain time aligned audio features that match the data rate of the skeletal poses we chose a proper FFT window overlap length. We finally normalize the audio features with a power-law function as:

$$S = (\frac{S}{max(S)})^{\gamma} \in \mathbb{R}^{T \times M}, \tag{5.1}$$

with T and M corresponding to the total number of time-frames and audio features respectively, and $\gamma = 0.6$, which is the gamma compression factor that works as a type of high-pass filter, emphasizing higher frequencies over the lower ones [129].

[4] https://cmu-perceptual-computing-lab.github.io/openpose/web/html/doc/md_doc_02_output.html

5.3.2 Dilated Causal Highway Convolutional Layer

The proposed architecture captures the spatio-temporal context of a sequential data stream by stacking Dilated Causal Highway Convolution (DCHC) layers. The key advantage of multilayered CNNs is their ability to create hierarchical structures that provide a shorter path for capturing long-term dependencies, compared to the recurrent representations modeled by RNNs [155]. Especially, the causal padding that is applied to the input stream ensures temporal integrity during feature learning, while the exponential increase in the dilation factor allows for a greater temporal receptive field [184]. Furthermore, deep architectures can leverage trainable highway gating functions [156, 185] for regularizing the flow of information through the network nodes and optimize their parameters faster. Although previous studies have already introduced the concept of the DCHC architecture [129, 143], it has to be noted that the provided descriptions lack sufficient implementation details. Specifically, the formula of the DCHC block proposed in [143], uses the $\tanh$ as an activation function for the transform gate with output range between -1 and 1, affecting further the carry gate to output values between 0 and 2. This does not agree with the original definition of the highway gating functionality that requires the carry gate to output values between 0 and 1 [156].

In our study, we consider the architecture of the DCHC layer shown in Figure 5.1. The output of the DCHC is calculated by the following function:

$$DCHC(X) = \sigma(H1) \odot tanh(H2) + (1 - \sigma(H1)) \odot X \tag{5.2}$$

The input of the layer is a sequence $X \in \mathbb{R}^{W \times F}$, where W and F correspond to the temporal context and feature dimensions respectively. Then, the input passes through the filter $H(x)$ and transform $T(x)$ gates, where their parameters $H1$ and $H2$ are learned by two separate dilated causal convolutional operations with the same dilation factor d. A $\tanh$ activation function is applied to the output of the filter gate, while the transform gate utilizes a sigmoid function in order to implement effectively the information highway algorithm. It should be highlighted that in our implementation we initialize the bias vector of the filter gate with a negative value (i.e. -1) such that the network is initially biased towards the carry gate. With a zero initialization scheme the model failed to converge completely, generating random numbers. The carry gate works as a residual connection that com-

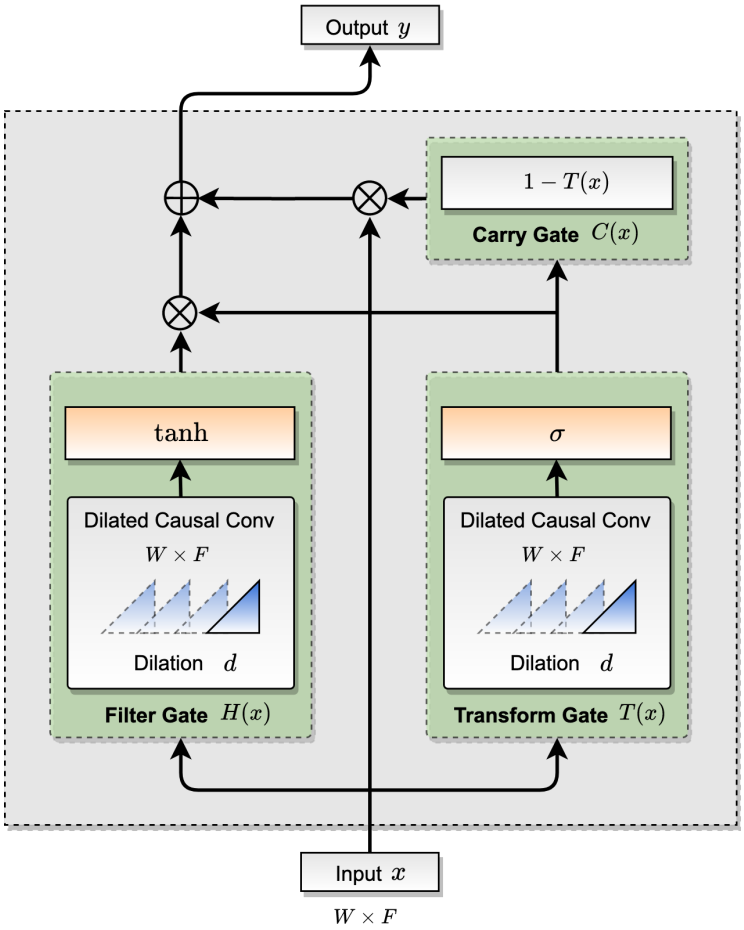


Figure 5.1: Dilated Causal Highway Convolutional layer.

bines the layer input with the output of the transform gate and computes the amount of the input information that will be carried to the output of the DCHC layer. In other words, the combination of the transform and the carry gates determine how much of the output Y is produced by transforming the input and carrying it [156].

5.3.3 Preliminary Ablation Study

The majority of related systems have used RNNs to model the temporal context of sequential data. The only method that relied entirely on convolutional netoworks [143], proposed a multimodal autoencoder architecture for automatic dance motion synthesis, where the two encoded input streams (skeletal poses and audio features) were fused with a conditional decoder, thus resembling the methodology presented in the gated CNN layer of the

PixelCNN model [186]. Initially, our attempts were focused towards replicating the multimodal architecture of [143], but we were not able to achieve comparable results. It is important to highlight that the descriptions of the DCHC block and the conditional autoencoder lacked important details. Moreover, no pretrained model or supplementary source code was available for reference. We therefore believe that there is not enough evidence to support the effectiveness and reproducibility of their system.

In this regard, having as a starting point our implementation of the system proposed in [143], we performed an architectural ablation study in order to evaluate the entanglement of each layer to the given objective. Initially, the multimodal encoder-decoder architecture was not able to generalize and the synthesized skeletal structures were either collapsing or converging to the global mean resulting to stagnant poses with no motion. Consequently, we discarded the encoder corresponding to the audio stream and focused only on the skeletal input. Then we trained multiple unimodal architectures, by removing one convolutional layer from the model each time, to assess its contribution to the feature learning process. After evaluating empirically the generated skeletal motions from each model we concluded to the unimodal autoencoder architecture, as follows.

5.3.4 Unimodal DCHC Autoencoder

The objective of the unimodal computational model is to predict the successive future skeletal poses conditioned on past poses from a sequence of poses X, hence our problem can be formulated as a product of conditional probabilities [184]:

$$p(X) = \prod_{t=1}^{t=T} p(x_t|x_1, \ldots, x_{t-1}) \tag{5.3}$$

An overview of the proposed architecture of the unimodal DCHC-based autoencoder is illustrated in Figure 5.2. The sequence of input poses X is defined by a temporal sliding window with dimensionality $W \times S \in \mathbb{R}$, where W and S correspond to the temporal dimension of the window and the number of the skeletal features, respectively. In each iteration the sliding window progresses forward for one time-step/frame. For instance in time-step n, the model receives as input a window of frames from $T_{n-w} - T_n$ and outputs a window of the same temporal dimensionality corresponding to the

$T_{n-w+1} - T_{n+1}$ time frames. During synthesis, the model follows the autoregressive objective mentioned above, by using its past predictions to generate the current pose.

The bottom layer of the encoder is a 1×1 convolution $\in \mathbb{R}^{W \times F}$, which upsamples the feature space of the input poses by computing F number of filters, followed by a ReLU activation function and 2 consecutive 1×1 convolutional layers with one more ReLU activation in-between. Next a stack of 10 DCHC layers encodes the output of the previous convolutional layer, each having a dilation factor $d \in [1, 3, 9, 27, 1, 3, 9, 27, 1, 1]$, respectively. The selected dilation stages ensure a wide receptive field, capable of capturing enough past context sufficiently. The output of the last DCHC layer is downsampled by a 1×1 convolution $\in \mathbb{R}^{W \times F/2}$, followed by a $\tanh$ activation.

The decoder receives the resulted latent features and applies a stack of 6 DCHC layers with dilation factors $d \in [1, 3, 9, 27, 1, 1]$. The output is then processed by 3 consecutive 1×1 convolutional layers with $\tanh$ activation functions. Finally, the top layer of the decoder architecture employs a 1×1 convolution $\in \mathbb{R}^{W \times S}$ with a sigmoid (σ) activation, in order to output the predicted future skeletal keypoint coordinates. All 1×1 convolutional operations have kernel size of 1, while the convolutions of the DCHC layers are computed with kernel size of 3.

5.3.5 Attention-based Multimodal Feature Fusion

The multimodal architecture that we propose for the audio-informed dance motion generation task was mainly inspired by [143], and it was implemented according to the results of the performed ablation study. Specifically, we carried out the appropriate architectural modifications to the DCHC layer as it was described in Section 5.3.2, allowing the model to properly leverage the functionality of the highway gates. Furthermore, we replaced the PixelCNN-like conditional layer in the decoder with a scaled dot-product attention mechanism [176], in order to fuse the encoded representations of the multimodal input.

In this regard, the objective of the model is to predict the successive future skeletal poses conditioned on past poses X and audio information Y of the same time resolution, formulated as a conditional probability:

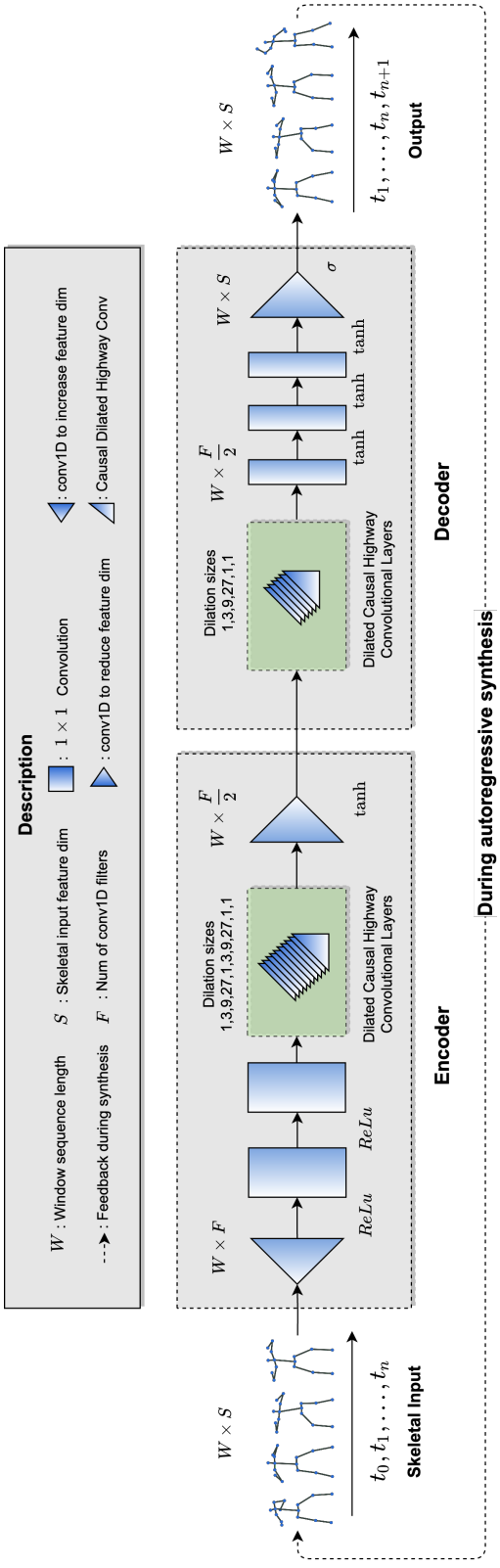


Figure 5.2: Overview of the considered unimodal DCHC-based architecture.

$$p(X|Y) = \prod_{t=1}^{t=T} p(x_t|x_1, \ldots, x_{t-1}, y_1, \ldots, y_{t-1}) \tag{5.4}$$

The multimodal input data streams in (5.4) are parsed by 2 temporal sliding windows with dimensionality $W \times M$ and $W \times S$, where W, M and S correspond to the temporal dimension of the input window and the number of the audio and skeletal features respectively. During training, in each iteration the sliding window progresses one time-step. For instance, in time-step n, the model receives a sequence of frames from $T_{n-w} - T_n$ and outputs a sequence of same temporal length, corresponding to the $T_{n-w+1} - T_{n+1}$ time frames.

An overview of the proposed multimodal network is illustrated in Figure 5.3. The two input data streams are encoded by two unimodal encoder networks of the same architecture, as described in Section 5.3.4. Then, the decoder receives the resulted latent representations and utilizes an attention mechanism, which works as stochastic conditioning for generating novel pose sequences. According to the original Transformer decoder [176], the target sequence (query) is used to condition the weighted sum of the input values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. This allows every position in the decoder output to attend over all positions in the input sequence. Similarly, the two encoded streams are merged into a joint representation set at the top layer of the proposed decoder network by applying a scaled dot-product attention mechanism, that packs the encoded skeletal features into a query matrix (Q) and the encoded audio features into pairs of key (K) and value (V) matrices, in order to compute the compatibility function. An optional mask can be added for preventing past information to flow in the decoder network and preserve the autoregressive property. The encoded skeletal features are also added to the output of the attention mechanism with a residual connection, followed by weight normalization. Then, similar to the unimodal decoder, a stack of 6 DCHC layers with dilation factors $d \in [1, 3, 9, 27, 1, 1]$ processes the salient features, followed by 3 consecutive 1×1 convolutional operations with $\tanh$ activation functions. Finally, a 1×1 convolution $\in \mathbb{R}^{W \times S}$ with a sigmoid activation outputs the predicted future skeletal keypoint coordinates. During the synthesis phase, the model generates poses based on an autoregressive procedure.

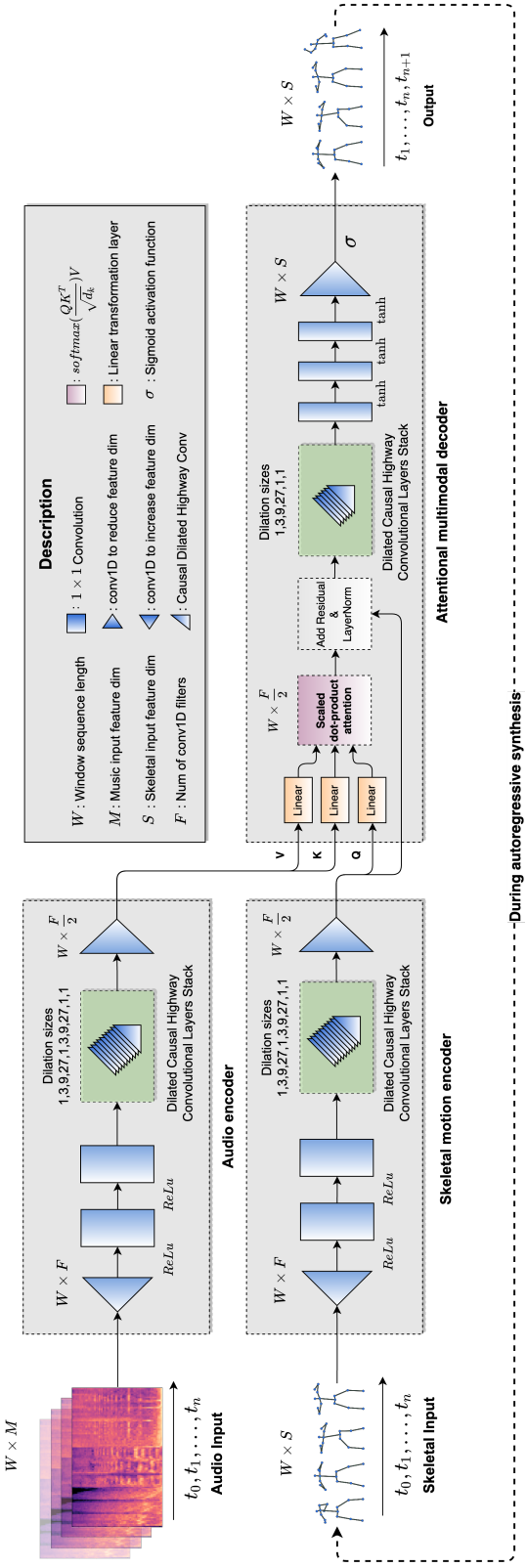


Figure 5.3: Overview of the considered multimodal architecture.

5.4 Unimodal Dance Synthesis Evaluation

In this section we describe the employed experimental setup along with the different evaluation scenarios that we designed, oriented towards investigating to what extent smaller or larger sizes of past context, affect the ability of the proposed unimodal autoencoder to synthesize skeletal-based dance motions, both in terms of pose consistency and motion variability. In this regard, we trained a set of models with different hyperparameter settings and generated a large amount of sequences for evaluating the effect of these parameters both empirically and subjectively based on a user study. Furthermore, according to the collected answers, the model with the best performance is compared with a RNN baseline architecture by computing various quantitative metrics.

5.4.1 Experimental Setup

All the experiments were implemented with the Pytorch [187] deep learning framework[5]. We designed various scenarios for investigating the effect of utilizing different sizes of past context on the ability of the model to synthesize human-like skeletal poses. The past context of the model is primarily specified by the temporal size W of the sliding window that is used as input to the model during training. However, this is not the only hyperparameter that affects the model's ability to capture long-term sequential correlations. The way that the loss is computed is also regulating the generalization of the model. In our experiments, we trained our models with a full teacher-force supervised learning procedure and monitor their performance based on the $L1$ loss function similar to [143], by computing the distance between the predicted and ground truth pose skeletal features as:

$$\mathcal{L}_{keyp}(pred^k, real^k) = \frac{1}{w}\sum_{i=1}^{w}|pred_i^k - real_i^k| \tag{5.5}$$

$$\mathcal{L}_{limb}(pred^l, real^l) = \frac{1}{w}\sum_{i=1}^{w}|pred_i^l - real_i^l| \tag{5.6}$$

$$\mathcal{L}_{comb} = \lambda_1\mathcal{L}_{keyp} + \lambda_2\mathcal{L}_{limb} \tag{5.7}$$

where two different $L1$ losses for the predicted keypoints (5.5) and the predicted limbs (5.6) are summed in (5.7) by utilizing the corresponding

[5] `https://pytorch.org/`

weighting factors λ_1 and λ_2. In our study we set $\lambda_1 = \lambda_2 = 1$, since we observed that our model was not able to converge with lower values. Also, considering that the loss function is calculated as the mean of sums, the model generalization is also affected by the employed batch size, in addition to $w \leq W$, which is the amount of predicted context that we take under account when computing the loss value that is used by the optimization function.

To this end, for our study we considered three groups of models with different input window sizes (30, 120, 500 time-steps per training sample), and during training we calculated the loss function based on five different sizes of past context (1, 15, 30, 120, 500), accordingly. We randomly selected 90 videos for training the models, while the remaining 10 formed our validation set. Totally, in our study we trained 12 unimodal DCHC models for 100 epochs, with batch size of 16 examples (i.e. input sequences), using the Adam optimization algorithm [106] with 0.0001 learning rate. Then we selected randomly 10 priming sequences of 10 seconds each from the validation files and generated 1 minute long sequences of skeletal dancing motions based on an autoregressive process, for each one of the 12 trained models, for the same set of 5 epochs (20, 40, 60, 80, 100), resulting to total 600 synthesized sequences.

5.4.2 Qualitative evaluation

By comparing the generated videos empirically, the differences of the models are evident on their ability to synthesize human-like skeletal poses, with consistent variability of motions. Some representative examples are illustrated in Figure 5.4. Regarding the models that were trained with a window size of 30 frames (see Figure 5.4b), they were unable to generalize when using either 1 or 15 past steps for computing the loss function, even after they were trained for 100 epochs. However, the model that was trained for 100 epochs and used 30 time-steps both for the input window size and for computing the loss function, was capable to generate motion sequences, but their long-term performance was poor, since the limb proportions started to collapse resulting to totally random movements. Similar behavior was observed in the motion sequences that were generated by the models that were trained with 120 and 500 frames of input, but with short past context (1,15,30) when computing the loss function (see Figure 5.4c).

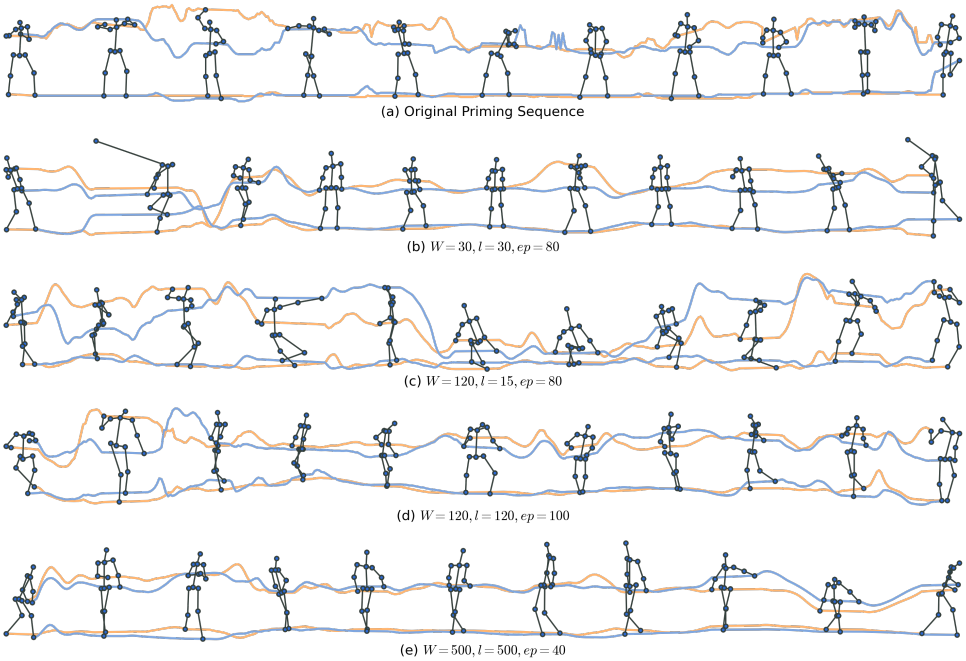


Figure 5.4: Examples of priming and generated motion sequences, illustrated as motion trajectories through time (left to right) of the skeletal keypoints corresponding to the hands and toes.

The models that were trained with samples of 120 frames (see Figure 5.4d), presented better performance in terms of pose consistency, but the motion patterns were not realistic enough. The best results were performed by the model that used 500 steps for the window size and for computing the loss function. Furthermore, we observed that this model generalizes sooner than the rest and was capable to synthesize consistent poses with smoother motion trajectories, even when it was trained for only 40 epochs (see Figure 5.4e).

5.4.3 Subjective Evaluation

We further conducted a user-study to subjectively evaluate the realism of the generated motion sequences and validate our empirical observations. To this end, we randomly selected 5 priming motion sequences from the validation set and then chose 5 representative synthesized sequences for each window length (30, 120, 500). Hence, each participant was presented with 20 tests in a random order so as to avoid any possible biases. Each test included a video clip of 10 seconds with only skeletal motion and no

sound, that could be either an original or a generated sequence. Then, the participants had to rate the presented video by answering the following 3 questions in a Likert scale from 1 (Artificial) to 5 (Real):

- **Q1:** Rate the overall realism of the skeletal motion.
- **Q2:** Rate the pose consistency of the skeletal structure.
- **Q3:** Rate the motion variability of the skeletal structure.

We designed our user-study with SurveyJS[6] by leveraging its online native services to host the random tests and store the collected answers. We invited 24 evaluators to participate in our study, from which 16 were male and 8 female, with the majority being 20 to 40 years old and all of them being familiar with the concepts of machine learning and artificial intelligence.

Totally we collected 1440 answers and their distributions are presented in Figure 5.5. We also apply a kernel density estimator for presenting the probability density of the answers at different values. By inspecting the median values (white dot points) of the answers, we observe that there is a tendency among the participants to rate higher the pose consistency and the motion variability of sequences that were generated from the trained models with an input window of 500 frames. Furthermore, to our surprise, these synthesized sequences are also rated as being more realistic than the actual ground truth, although the pose consistency and motion variability of the original data were always rated higher.

In order to determine the statistical importance of this preference, we formulated the appropriate AB pairs and performed the corresponding non-parametric Wilcoxon rank sum test, having as null hypothesis that there is no difference in the perceived realism between the actual and the generated sequences. The calculated p-values (see Table 5.2) demonstrated that the differences between the medians of the ground truth and the synthesized sequences are statistically significant, indicating that there are enough evidence to reject the null hypothesis.

5.4.4 Quantitative Diversity evaluation

RNN-based networks have been proven effective for modelling the temporal correlations of skeletal poses in dance motion sequences. In this regard,

[6] https://surveyjs.io/

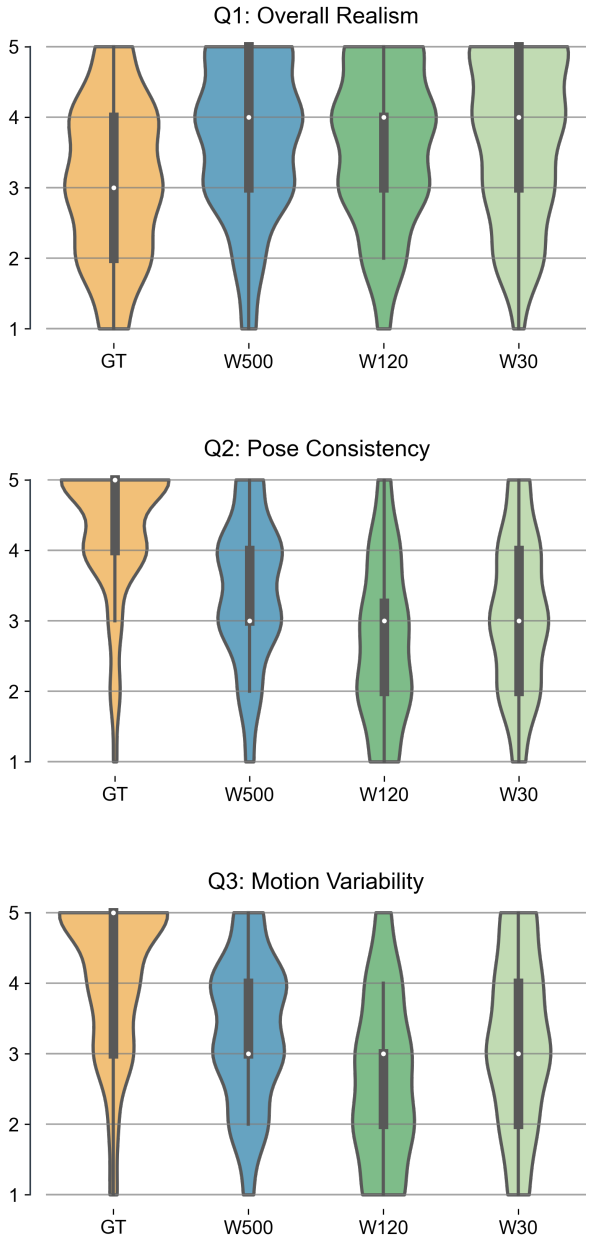


Figure 5.5: Boxplots of the collected ratings from the unimodal user study. The colored shapes around the box plots represent the distributions of the answers by applying a kernel density estimator function.

Table 5.2: Wilcoxon rank sum test p-values between the original and the synthesized motion sequences.

AB pairs	Q1	Q2	Q3
GT–W500	9.53×10^{-5}	7.85×10^{-12}	5.61×10^{-9}
GT–W120	9.18×10^{-4}	1.46×10^{-21}	1.46×10^{-21}
GT–W30	8.26×10^{-5}	7.68×10^{-16}	2.60×10^{-9}

according to the results provided by the user-study, we selected the unimodal DCHC architecture that received the best ratings and compared its performance with a baseline CNN-LSTM network, following the same settings proposed in [188]. The baseline model comprises one 1×1 convolutional layer as a feature embedding operation, followed by a stack of 3 LSTM layers with 1024 hidden cells for modelling the temporal correlation of the skeletal features. Finally, a sigmoid activation function is applied on the output of the top LSTM layer for providing the future pose prediction. We also implemented the baseline architecture in Pytorch and followed the same hyperparameter settings and training strategy as described in our experimental setup.

In order to automatically evaluate the two unimodal architectures, we generated 180 sequences of 2000 poses for each model by choosing a random priming skeletal sequence of 2 seconds from the validation set. Then we used the coordinates of the predicted skeletal poses and computed various diversity metrics which are usually employed in related studies [138, 145, 150, 172, 173], including:

1. The *Fréchet Inception Distance (FID)* [189] score that measures the distance between the distributions of the generated poses and the original. We calculated the FID score of the original sequences by randomly dividing the validation set in two equal parts of 1 min clips.
2. *Inter-sequence diversity* score that computes the variability between pairs of sequences. We compute this metric by randomly selecting sequences and measure the $L2$ distance between the feature vectors of each pair. The final score is the average over the sequence length.
3. *Intra-sequence diversity* metric that computes the variability of poses belonging to the same sequence. We divide each pose sequence into 4 segments, and compute the $L2$ feature distance among all possible pairs. Next we average over all pairs from all sequences.

Table 5.3: Comparison of diversity scores between the baseline model and our proposed unimodal architecture.

Model	**FID** $\downarrow$	**Inter-seq L2** $\uparrow$	**Intra-seq L2** $\uparrow$
Original	0.11 ± 0.01	0.76 ± 0.03	0.35 ± 0.08
CNN-LSTM [188]	0.55 ± 0.02	0.36 ± 0.02	0.52 ± 0.10
DCHC-UNI	0.46 ± 0.02	0.75 ± 0.02	0.52 ± 0.08

For preventing randomness in our results we performed 200 trials for the diversity metrics and report their average and standard deviation in Table 5.3. From the statistics we can observe that the CNN-LSTM network is quite deterministic since it does not present enough variability among the different generated sequences, as it is reported by the inter-sequence diversity score. On the contrary, the proposed unimodal DCHC network, even though it is a deterministic model, reports greater inter-sequence diversity that is quite close to the original. Also, the reported FID score indicates that the distribution of the generated poses are closer to the original, compared to the CNN-LSTM model. However, both unimodal architectures report the same intra-sequence score that is higher than the original, indicating that the generated motion patterns may not be smooth enough, generating faster frame-by-frame transitions between the skeletal coordinates of the poses.

5.5 Audio-informed Dance Synthesis evaluation

In this section we describe the applied experimental setup along with the different experimental scenarios that we employed for evaluating the performance of our proposed multimodal DCHC autoencoder in terms of realism and style consistency. Herein we extend our previous experiments by training the considered multimodal architecture based on a curriculum learning strategy. Furthermore we provide a thorough assessment of our proposal by employing 2 related state-of-the-art frameworks, in order to generate numerous motion sequences that were used to conduct qualitative, quantitative and subjective evaluations. Also, we try to determine the effect of curriculum learning on the proposed feature fusion, compared to the fully guided teacher-forcing scheme, by providing comprehensive visualizations of the attention mechanism.

5.5.1 Experimental Setup

Autoregressive generative models are known to suffer from the "Exposure Bias" problem, referring to the train-test discrepancy that arises when only ground-truth context is used at training time, but generated one at test time [190]. To this end, we exploited our proposed architecture on an autoregressive curriculum-based self-supervision strategy, in contrast to a typical fully teacher-force supervision scheme for modelling the audio-driven conditional objective. Specifically, the proposed autoregressive curriculum learning strategy was implemented by providing an initial ground-truth input sample of W time-steps and then autoregressively generating N future poses before computing and minimizing the loss function as described in (5.5), (5.6) and (5.7). Furthermore, we evaluated 2 different setups regarding the proposed attention-based feature fusion, where we calculated the attention scores (compatibility function) with or without the application of an upper triangular look-ahead mask, which further restricted the attention mechanism to attend only past inputs.

Also, in our experiments we compared the performance of the 4 models deriving from the 2 supervision strategies, against 2 adversarial cross-modal state-of-the-art frameworks. As a primary baseline we chose the system described in [138], since it is considered to be the first study that formulated the audio-driven dance motion generation task. As a second baseline we selected the current state-of-the-art proposed in [150], where the authors also compared their system with the primary baseline [138] in their presented evaluation campaign.

We implemented our proposed multimodal architecture with Pytorch and trained our 4 models using the same hyperparameter settings and dataset split as it was described in Section 5.4.1. The only difference is that the 2 models that were trained with the self-supervised curriculum learning scheme, generated $N = 5$ future poses for each input sequence in the training batch. As it concerns the 2 baselines, we used their corresponding publicly available pre-trained models and transformed our validation dataset accordingly, so as to comply with the different input requirements of each framework. We chose not to train the baselines with our dataset, since both studies employed either K-pop or urban dancing style in their experiments. However, it should be highlighted that our models were trained with pairs of audio and skeletal features that were both aligned in 30 FPS, while the

models in [138] and [150] considered raw audio as input and generated skeletal feature sequences of 15 FPS and 10 FPS respectively. Another limitation of [150] is the fact that it receives audio clips of only 5 seconds in order to generate the corresponding skeletal dance sequence. In this regard, we did not apply any kind of post-processing on any of the generated sequences, avoiding possible biases in our experiments. In our results we refer to [138] and [150] as "NIPS" and "MM" respectively, while our models that were trained with teacher-forcing correspond to "FAF" (full attention fusion) and "MAF" (masked attention fusion), in addition to "SS-FAF" and "SS-MAF" that denote our self-supervised models.

5.5.2 Diversity and Multimodality Evaluation

Choosing proper quantitative metrics for assessing automatically the performance of generative systems is a quite challenging task. Especially, the complexity of dancing motions is reflected by highly expressive and diverse body movements. To this end, various diversity metrics have been proposed in the literature as it was described in Section 5.4.4. Furthermore, audio-driven dance generation systems usually require additional metrics for validating their ability to synthesize original sequences in cases where the same conditioning is applied [138]. Therefore, in addition to the 3 diversity metrics, herein we also consider the following 2 multimodality metrics:

1. *Same-music conditional multimodality*, where given the same audio features, we generate multiple skeletal pose sequences and compute the $L2$ feature distances between all possible pairs. Similarly to the intra-sequence metric, we average over all pairs from all sequences.
2. *Same-skeletal conditional multimodality*, where given the same priming skeletal sequence but different audio features, we generate numerous motion sequences and compute the feature distance similar to the aforementioned same-music multimodality metric.

For computing the diversity metrics we randomly selected 20 audio clips with duration of 10 seconds for each song in the validation set, along with 20 random priming poses. Then we generated 180 skeletal dance motion sequences of 10 seconds each, for all the considered models in this study, totaling 1080 synthesized dance motion sequences. However, the models that were trained with the curriculum learning approach were not able to

generate sequences with only a single priming pose. Thus requiring longer sequences as initial context, which based on empirical observations was increased to 2 seconds (i.e. 60 poses). Furthermore, to overcome the limitation of MM to generate motion sequences of 5 seconds, we simply divided each 10 second audio sample into 2 parts and concatenated the generated motion sequences respectively.

Regarding the same-music multimodality metric, we randomly selected from each of the 9 songs in our validation set an audio clip with duration of 10 seconds, and by using a random priming pose (or sequence of poses for the self-supervised models), we generated 10 extra dance motion sequences that were conditioned on the same audio features, resulting to 90 synthesized dance motion sequences per model (total 540 sequences). The same-skeletal multimodality metric applies only to our self-supervised models that require more than a single pose as initial context. In this regard, we randomly selected 10 priming skeletal sequences of 2 seconds and for each priming sequence, we selected 1 random audio clip of 10 seconds from every song in our validation set, for generating 9 dance motion sequence that were conditioned on the same priming skeletal features, summing to 90 synthesized motion sequences for each self-supervised model (total 180 sequences).

It should be highlighted that both NIPS and MM frameworks output skeletal features in different ranges, thus requiring to normalize all generated poses to $[0-1]$ range for retrieving comparable results. Also, in order to prevent randomness in our experiments, we run 200 trials for all the diversity and multimodality metrics and present their average and standard deviation in Table 5.4. According to the computed diversity scores, our proposed architecture outperformed both baseline frameworks, while both our models employing full-attention feature fusion reported the closest FID scores to the original. Additionally, we observe that the autoregressive mask promoted both the inter-sequence and intra-sequence diversity. However, the self-supervised models presented slightly inferior performance compared to our models that were trained with teacher-forcing. The worst FID score was reported by NIPS indicating that multiple poses were quite far from the original distributions. Although, its reported inter-sequence metric indicates high variability between the different synthesized sequences, which can be explained by the fact that NIPS is a VAE network that samples a priming pose from the learned skeletal distributions for initializing the hidden states

of the RNN dance sequence generator.

Furthermore the same-music multimodality score reported by NIPS is also justified by its VAE-based cross-modal architecture. On the contrary, MM learned a direct cross-modal mapping by using a GRU-based audio encoder and a multi-layer perceptron as pose generator that presented deterministic performance, according to the reported multimodality score. As it concerns our models, the reported scores in both multimodality metrics validate the stochastic ability of our proposed multimodal feature fusion architecture to generate diverse dance motion sequences, whether conditioned on the same audio features or on the same priming skeletal context. Similar to the diversity metrics, we observe that our models that employed a look-ahead mask to compute the compatibility function presented superior performance compared to the full-attention feature fusion.

5.5.3 Qualitative Evaluation

By visualizing the generated sequences we try to assess the considered audio-informed models from a qualitative perspective. Some representative examples of synthesized pose sequences are presented in Figure 5.6, illustrating the motion trajectories of 10 seconds, that correspond to the skeletal hands and toes, while we visualize a complete pose every second.

As it concerns our multimodal architectures that were trained with teacher-forcing, we observe that the absence of the look-ahead mask is crucial, since it affects the ability of the model to synthesize expressive and diverse dance motions. Furthermore, the motion sequences that were generated by the full-attention feature fusion model (i.e. FAF), tend to freeze within the first second. Even though its FID metric indicated that the generated poses may be realistic (best FID score), the reported intra-sequence diversity failed to capture the almost stagnate motion patterns. On the other hand, by applying a triangular autoregressive mask in our proposed attention-based feature fusion (i.e. MAF), the model is forced to maintain its causal functionality, generating highly expressive motion patterns. This observation is also reflected on the computed qualitative assessment, since the MAF model steadily reported the highest scores in all diversity and multimodality metrics.

The self-supervised models that were trained with the proposed curriculum learning technique, presented similar performance to the teacher-

Table 5.4: Comparison of diversity scores between the baseline models and different setups of our proposed multimodal architecture.

Model	**FID** ↓	**Inter-seq L2** ↑	**Intra-seq L2** ↑	**Multimodality**$_{sm}$ **L2** ↑	**Multimodality**$_{ss}$ **L2** ↑
Original	0.11±0.01	0.76±0.03	0.35±0.08	-	-
NIPS [138]	1.27±0.04	0.73±0.10	0.25±0.06	0.59±0.08	-
MM [150]	0.67±0.01	0.22±0.01	0.19±0.03	0.00±0.00	-
FAF	0.28±0.01	0.44±0.01	0.37±0.06	0.44±0.03	-
MAF	0.43±0.01	0.53±0.01	0.49±0.03	0.53±0.01	-
SS-FAF	0.34±0.01	0.42±0.02	0.31±0.05	0.38±0.03	0.41±0.03
SS-MAF	0.53±0.02	0.54±0.01	0.44±0.15	0.49±0.11	0.49±0.01

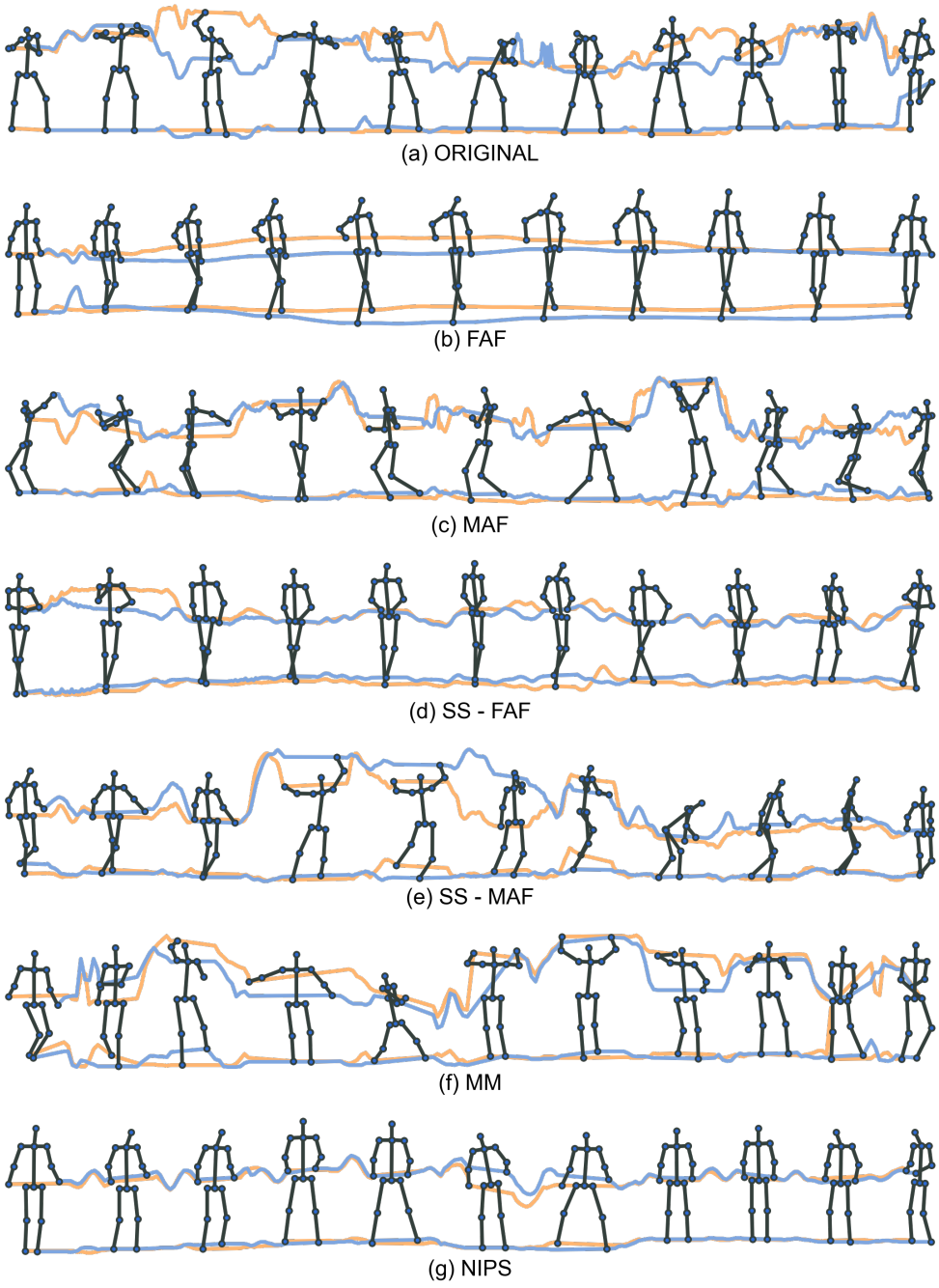


Figure 5.6: Generated examples from the considered models in our audio-driven dance motion synthesis study.

forced models. By inspecting the shape of the motion trajectories, the superiority of the masked-attention feature fusion (i.e. SS-MAF) to synthesize diverse motion patterns, compared to the full-attention scheme (i.e. SS-FAF), is more than evident. Furthermore, we observe that the self-supervision training approach alleviated the problem of the frozen poses in the case of the full-attention, although the diversity of the generated sequences remains low, with lots of micro-fluctuations in the coordinates of the generated skeletal keypoints between consecutive frames. The same floating micro movements are also present in the sequences generated by SS-MAF, affecting further the overall smoothness and realism of the generated motion patterns.

Regarding the dance motion sequences generated by the 2 baseline methods, we can see that the sequences generated by NIPS resemble the sequences that were synthesized by SS-FAF, while MM presented many similarities with our models that employed an autoregressive mask for computing the attention scores. Additionally, we observed that some sequences synthesized by NIPS were prone to error accumulation, resulting to poses that were moving outside the frame borders, while this fact is also indicated by the reported FID score. The limitation of MM to synthesize clips of only 5 seconds with a frequency of 10 FPS have an impact on the overall smoothness of the generated motion patterns. Furthermore, the simple concatenation of consecutive generated clips is more than apparent, which likely influences the perceived realism.

5.5.4 Visual Explanations of Feature Fusion

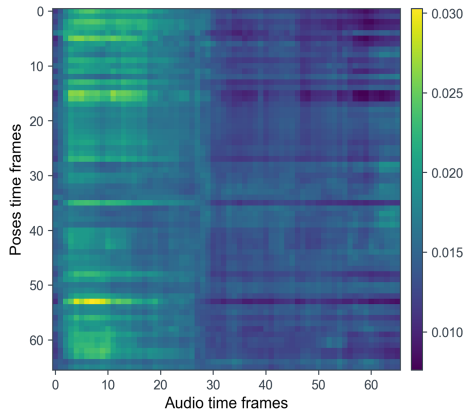
The proposed feature fusion method employs both audio and skeletal motion features in the compatibility function, computing the possible attention matrix that lies in the small subspace of $\mathbb{R}^{W \times W}$. Overall, there are 2 different design directions that incorporate different levels of information for multimodal feature alignment. One option is to apply a full-attention scheme where the compatibility function have access to all time-steps, while the alternative is to use an upper triangular mask, restricting the attention mechanism to consider information only from past context.

In Figure 5.7 we present some examples of the attention matrices as they were computed by our models trained either with full-attention or by utilizing the corresponding look-ahead mask. From a computational per-

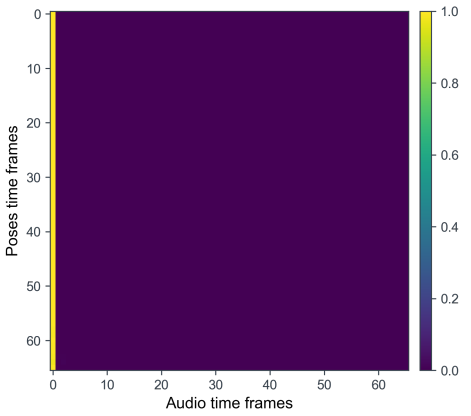
spective, the full-attention approach is quite costly to train with typical teacher-force learning. As it can be seen in Figure 5.7a, the computed attention matrix resembles a dense matrix with quite small values, affecting the model to generate sequences with frozen poses. Since there is a natural linear correlation of the temporal order of the skeletal poses with the corresponding audio features, we would expect to retrieve a sparse scoring matrix with non-zero values, following the main diagonal of the matrix. Especially the utilization of causal masking, should be enforcing the model to learn this diagonal correlation. However, we observe that the masked attention model even after it was trained for 100 epochs, it fails to pay attention to the audio features, thus scoring as most important only the skeletal features (see Figure 5.7b).

Therefore, in order to address these issues we applied a self-supervision scheme to train our proposed architecture, leveraging prior knowledge that could be embodied into the model, towards alleviating the training burden of the attention module. In this regard, we followed the proposed autoregressive curriculum learning approach to train our multimodal architecture with both full-attention and masked-attention feature fusion methods. By visualizing the computed alignment scores, we show that this simple self-supervision scheme actually facilitated the training process and improved the attention matrix in both methods. Specifically, we observe that the sparsity of the full-attention score matrix was increased (see Figure 5.7c), while the masked-attention model learned to compute scores that are nearly diagonal (see Figure 5.7c). However, the full-attention alignment matrix even though it presented high sparsity, it still failed to capture the causal spatio-temporal relation of the two modalities. As it was observed in our qualitative evaluation, this defect is also reflected in the generated dance motion sequences, where the synthesized poses seem to stagnate within the first second.

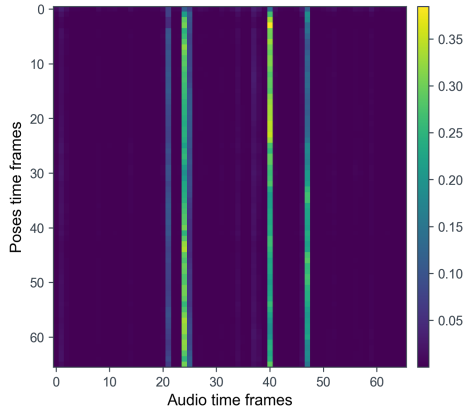
To this end, by visualizing side-by-side the attention weights along with the multimodal input streams, we try to provide interpretable insights regarding the effect of the different attention methods to the scoring decisions and outcome of the models, as it is illustrated in Figure 5.8. Specifically, we can see that the full-attention model (see Figure 5.8a), due to the fact of having access to all time-steps, is prone to error accumulation that is also reflected in the corresponding alignment matrix, by giving most of its attention to a single audio frame and thus synthesizing poses that are very



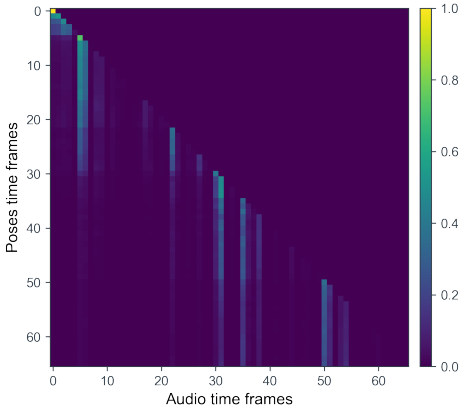
(a) Teacher-forced full attention.



(b) Teacher-forced maksed attention.



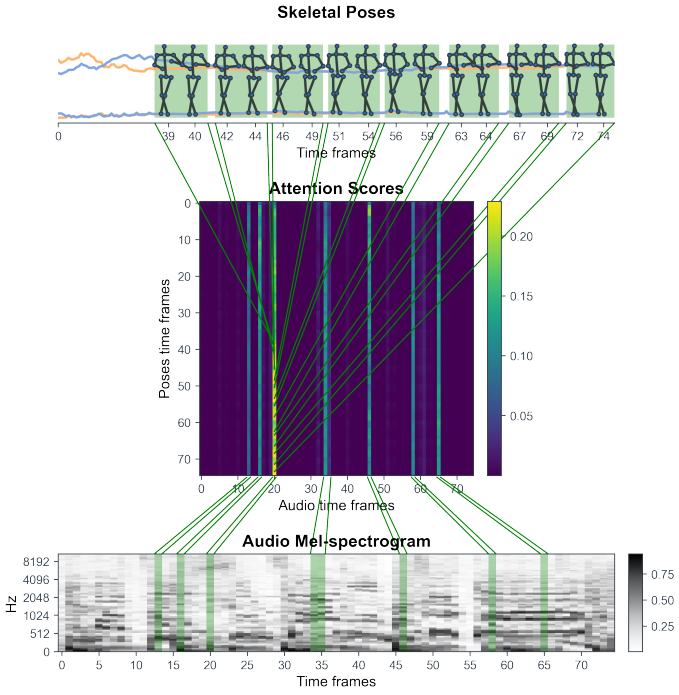
(c) Self-supervised full attention.



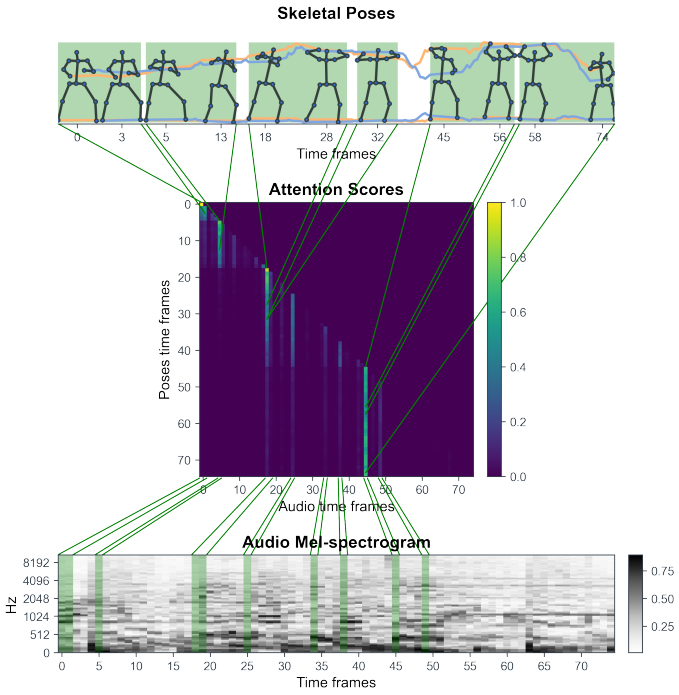
(d) Self-supervised masked attention.

Figure 5.7: Examples of the different training approaches and their effect on the attention scores.

similar to each other. On the other hand, the masked-attention example presented in Figure 5.8b, is more expressive than the full-attention model because internal tokens learned to follow the autoregressive momentum, which is further prompted by the applied look-ahead mask. By providing access only to previous time-steps, encourages the network to focus on the temporal correlation of the two modalities, which along the proposed curriculum learning approach are the key factors for training a model that does not suffer from motion freezing and maintains its expressiveness after few predictions during synthesis in test time.



(a) Self-supervised full attention.



(b) Self-supervised maksed attention.

Figure 5.8: Examples of the two self supervised attention-based feature fusion alignment matrices, along with the corresponding input streams.

5.5.5 Subjective evaluation

In order to subjectively evaluate the realism and style consistency of the generated dance sequences we conducted a user-study, by including pairwise comparisons of our multimodal method with the 2 baseline frameworks and real dances. Based on our qualitative observations, we discarded our full-attention feature fusion models from the user-study due to their poor performance, thus keeping only those sequences that were generated from the 2 masked-attention models.

Specifically, from the 180 sequence generated by each individual method in our diversity evaluation, we randomly selected pairs of dances with the same music, formulating the following 10 AB groups: (Real-NIPS), (Real-MM), (Real-MAF), (Real-SSMAF), (SSMAF-NIPS), (SSMAF-MM), (SSMAF-MAF), (MM-NIPS), (MM-MAF), (NIPS-MAF). We repeated this process 2 times for each participant in order to create 20 random tests, while the order of the tests as well as the methods in each AB group were randomly shuffled, avoiding possible biases. All generated sequences were rendered in the native output frequency of each model, while the original dance sequences were rendered in all possible frame rates (10, 15, 30 FPS) according to their pair in each comparison. Every group was presented to the user as a single test containing 2 videos side-by-side, followed by 2 questions of preference where the user was required to select either A or B, answering:

- **Q1:** Which dance is more realistic regardless of music?
- **Q2:** Which dance matches the music better?

In order to automate this process we designed our user-study by adopting the SurveyJS[7] library to our requirements. Furthermore, we developed a web-server for creating and presenting on-the-fly the random tests to the user, while all the answers along with their corresponding random test templates were stored in a database, so we could later process them accordingly. We invited 45 participants in our study, from which 30 were males and 15 females, with the majority of their ages spanning from 20 to 50 years old, while 9 users being older than 50 years. We also asked the participants to rate their familiarity with the concepts of machine learning and artificial intelligence, and most evaluated themselves at least to the moderate level.

[7] `https://github.com/surveyjs/survey-library`

At the end of the survey, we also included an area for comments as optional feedback. Totally we collected 1800 preference answers and the results are presented in Figure 5.9.

From the statistics, it becomes clear that real dances were in most cases perceived as being more realistic and style consistent to the music in every comparison. However, we found that the reported preferences in our user-study, regarding the groups of real choreographies with both MM and NIPS, are significantly lower from those presented in their manuscript. For instance in Q1, 30% of the users in our study preferred MM over real dances, in contrast to 51% that was reported in [150]; similarly for NIPS, herein we accounted 12% versus 32.6% that was presented in [138]. Moreover, we observed that MM outperformed both our methods in direct comparisons, while our model that was trained with teacher-forcing performed better than both NIPS and our self-supervised approach.

According to the collected comments, the users observed that the videos from NIPS presented slow-paced motion patters that were almost stagnant, while the self-supervised sequences (SSMAF) included numerous pose fluctuations that affected negatively the users' preference. Regarding the sequences synthesized by our teacher-forced model, the users consistently commended that the sizes of the generated poses were bigger than other sequences, which might be induced by the bounding-box normalization applied in our training data. Furthermore, they noted that in songs with faster tempo, the generated motion patters were failing to follow the musical context. Additionally, we were expecting the low frame rate of MM to have a negative impact on user preference, however the participants commented that these sequences were quite expressive and consistent with the given music. Although, many users were able to identify the concatenation gap between the 2 consecutive clips that formed each 10-second video, suggesting the inefficiency of MM to synthesize long-term choreographies. To this end, by comparing the realism preference statistics with the FID scores reported in our quantitative evaluation, the discrepancy is apparent, which further indicates the inability of FID as an automatic metric to provide human-like assessment regarding the realism and expressiveness of synthesized dance motion sequences.

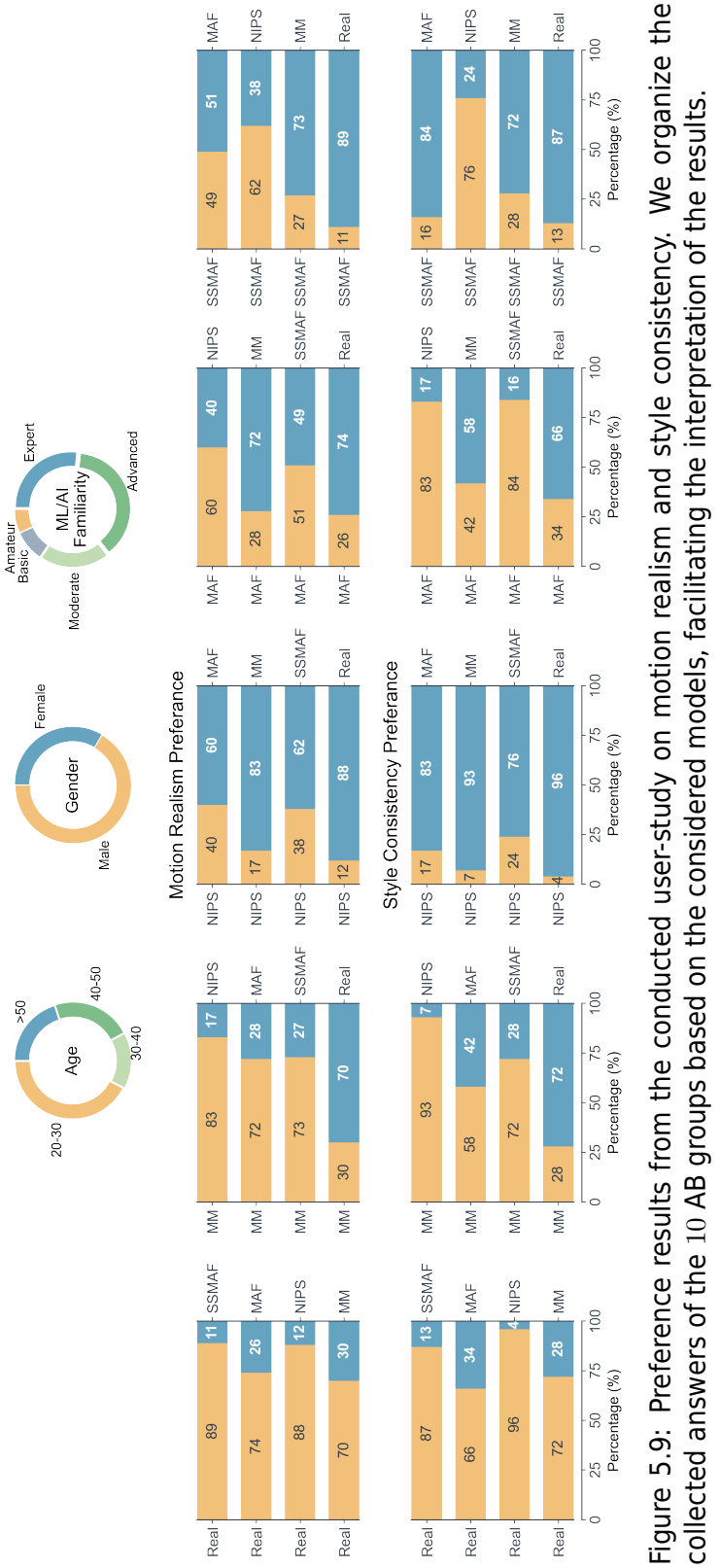


Figure 5.9: Preference results from the conducted user-study on motion realism and style consistency. We organize the collected answers of the 10 AB groups based on the considered models, facilitating the interpretation of the results.

5.6 Conclusions

In this Chapter we presented a thorough study on the ability of deep CNNs to model and synthesize novel dance motion sequences from both skeletal and audio features. The motivation of this work lies in the fact that most of the available proposals in the literature, utilize RNNs for modelling the spatio-temporal correlation of audio and motion sequences. However, the recurrent representations created by RNNs are susceptible to prediction error accumulation, restricting the models to generate short motion sequences, usually of less than 100 poses. The proposed DCHC layer bridges this gap by leveraging the effectiveness of stacked dilated causal convolutions to provide a wider temporal receptive field of 500 poses with only few layers, while the information-flow of the network is controlled by highway gating transforms, allowing the model to generalize faster to the given objective. Furthermore, the proposed conditional decoder employs an attention mechanism to fuse the latent multimodal representations of past audio and motion information, in order to stochastically control the generation process, enhancing the creativity of our system.

In our experimental campaign we trained the proposed unimodal architecture solely with skeletal information, by considering 12 different setups, based on 3 different temporal window lengths as past context. Then, we autoregressively synthesized 600 motion sequences by randomly choosing priming information from the validation set and provide a qualitative evaluation based on empirical observations. In order to subjectively evaluate the perceptual realism of the generated motion sequences, we further conducted a user study, where 25 participants had to compare original and synthesized sequences, by rating their consistency in terms of pose integrity and motion variably. The collected answers show that the best performance was presented by our models that were trained with temporal context of 500 time-steps. Additionally, we compared our best model proposed by the users with a CNN-LSTM [188] baseline by computing various diversity metrics, validating the superiority of our convolutional network to effectively generate diverse and realistic skeletal motion sequences.

Regarding the proposed multimodal architecture, we considered 2 approaches for computing the attention scores by either applying a full-attention scheme or a look-ahead mask. In our experiments we exploited both typical teacher-forcing and self-supervised autoregressive curriculum

learning, for training a total of 4 models, in order to compare their performance with 2 state-of-the-art audio-driven dance motion synthesis frameworks. Overall, we generated 1800 dance motion sequences for evaluating the perceived realism, motion diversity and multimodality based on numerous quantitative, qualitative and subjective experiments. Specifically, the quantitative metrics indicated that our models outperformed both baselines, while the best trade-off performance was yielded by the models that employed an autoregressive mask in the attention mechanism. Our empirical observations on the generated sequences supported the qualitative superiority of masked attention over full-attention feature fusion. We further interpreted this difference by visualizing representative examples of computed score matrices, while providing adequate explanations of the effect of self-supervision training on the model's ability to learn the cross-modal spatio-temporal alignment of the two encoded input streams.

For subjectively evaluating the performance of the masked attention feature fusion method, we conducted a user-study based on pairwise comparisons of our masked models with the 2 baseline systems and real dances, totaling 1800 preference answers from 45 participants. The results from the collected answers proposed the superiority of the sequences generated by [150] in terms of motion realism and style consistency, while our teacher-forced model outperformed our self-supervised method and [138]. However, the pre-trained model provided by [150] is limited to generate sequences of only 5 seconds and according to the participants' feedback, the concatenation gap of the underlying clips that formed each 10-second video in our user-study was always noticeable. Overall, we observed that the sequences generated by [138] were consistently found to present the worst performance in all the considered experimental scenarios. Although our composite reconstruction loss may be sufficient to encourage our self-supervised masked-attention model to capture the causal spatio-temporal relation of the two modalities, it still presented minimal motion fluctuations in the generated sequences, with a negative impact in the perceived realism.

Part III

Symbolic Music Features

CHAPTER 6

Jazz Improvisation Accompaniment

6.1 Introduction

The use of automatic systems for generating music is a captivating vision and a multidisciplinary research problem studied for decades. The diversity of automatic music generative systems relies on their different objectives and the musical content that they produce, such as chord progressions, melody generation, accompaniment arrangements and counterpoints [191]. Already from the late 1950s and early 1960s, composers such as Lejaren A. Hiller [192] and Iannis Xenakis [193] explored stochastic models for algorithmic music generation.

With the recent advances in the computational capabilities of modern computers, there is an exploding tendency of generative system proposals, incorporating complex artificial neural network architectures as a technical foundation. Conditional generative models based on GANs have been used to combine unpaired lead sheet and MIDI datasets for generating lead sheet arrangements. The lead sheet arrangement can be defined as the process that receives a lead sheet as input and outputs piano-rolls of a number of instruments to accompany the melody of a given lead sheet. For instance, Liu and Yang [194] proposed an architecture that comprises 3 stages (lead sheet generation, feature extraction and arrangement generation) in order to generate eight-bar phrases of lead sheets and their arrangement.

The feature extraction stage is responsible to compute symbolic-domain harmonic features from the given lead sheet in order to condition the generation of the arrangement. Wang and Xia [195] developed a framework for generating both lead melody and piano accompaniment arrangements of pop music. Specifically, they consider a chord progression as input and propose 3 phases for generating a structured melody with layered piano accompaniments. First, the harmony alternation model receives a given chord progression in order to transform it to a different one that fits better with a specified music style based on HMMs. Then, the melody generation model generates the lead melody and the layered accompaniment voices through seasonal Autoregressive Moving Average (ARMA) processes. The final phase implements the melody integration model that is responsible for integrating the melody voices together to form the final piano accompaniment.

On the other hand, RNNs are often used to generate sequences of musical content in a stepwise manner, where the network input is the previous note and output is considered the predicted note to occur on the following time interval [196]. Similarly, RNN architectures with LSTM gates have been utilized for generating blues style melodies conditioned on a given chord progression [197]. By definition, LSTM-based models have the ability to correlate and capture the temporal context of a sequence, thus simulating the human cognitive abilities for predicting sequential information. Also, RNNs have proven efficacy on modelling complex musical structures such as polyphonic chorales. For instance, the "DeepBach" system was trained to generate four-part chorales in the style of J.S. Bach [198]. As it is demonstrated by Jaques *et al.* in [199], generative systems can be also constrained by music theory rules via a reinforcement learning mechanism. In addition to the music theory rules, Lewandowski *et al.* [200] considered probabilistic harmonic and rhythmic rules based on distribution estimators, conditioned on the output of a RNN model that was trained to discover temporal dependencies from polyphonic music scores of varying complexity.

Other approaches take into account the chord progressions for providing longer musical structures. For instance, the work presented in [201] utilized a text-based LSTM network to capture the relationships within text documents that contained symbols of chord progressions. Another example based on chord progressions is the "JamBot" system [202] that generates music in two steps. The bottom network is a LSTM architecture that predicts

a chord progression based on a chord embedding, while a second LSTM network generates polyphonic music based on the predicted chord progression received from the bottom network. Nevertheless, this approach lacks the ability of modeling interactions within a polyphonic musical ensemble. In order to overcome this limitation, Chu *et al.* [203] proposed a hierarchical architecture, where each level is an individual RNN that generates different accompaniments for the synthesized song. A monophonic melody is generated first, followed by the accompanying chords and drums.

In the scope of the "Impro-Visor" project[1], Johnson *et al.* [204] proposed a neural network architecture consisting of two LSTM-based sub-networks that jointly learn to predict a probability distribution over future notes conditioned on past notes of the melody. Furthermore, researchers from the same laboratory developed the "JazzGAN" system [205] that utilizes RNN-based GANs to improvise monophonic jazz melodies over given chord progressions. Their results indicated that the proposed system was capable to address frequent and diverse key changes, as well as unconventional and off-beat rhythms, while providing flexibility with off-chord notes. Other proposals incorporate music theory grammar in combination with LSTM neural networks to generate jazz music. For instance, [206] extracted the interval, duration and note category information from jazz MIDI files and trained a LSTM model to learn the transition probabilities between notes. Then they take advantage of the music grammar in order to arrange and output the generated sequence of notes.

LSTM networks have been also tested for generating jazz music compositions constrained by a given performer's style. In particular, De Prisco *et al.* [207] developed a 3 staged generative system, consisting of a One-Class SVM for learning the performing style of a specific jazz musician, an LSTM network to generate patterns relevant to the learned style and a splicing system to compose melodic lines in the given style. Splicing systems are formal models for generating "languages" (i.e. sets of words), inspired by a recombinant behavior of DNA [208]. A music splicing composer requires to define an alphabet, an initial set and a set of rules. Another example of a complex system that utilizes LSTM-based networks for learning statistical correlations between instruments within a jazz piano trio ensemble (piano, bass, drums) was proposed by Hori *et al.* in [209]. They trained a LSTM

[1] `https://www.cs.hmc.edu/~keller/jazz/improvisor/`

architecture to learn the relationship between the musical features of the piano performance that is applied on top of a HMM, which is responsible to segment the bass and drums performance feature spaces. Overall the system was reported capable of generating coherent rhythmic patters and bass melodies as accompaniments to a piano solo input. However, the authors specify that their model can be further improved due to the lack of available jazz datasets. To this regard, Hung *et al.* [210] employed transfer learning techniques aiming to solve the problem of jazz data insufficiency. They proposed a Bidirectional GRU VAE generative model, trained on a dataset of unspecified genres as source, and a Jazz-only dataset as target.

It is worth noting that only a few projects experiment with real-time creative scenarios where a human improviser is accompanied by an automatic agent without any musical constraints. To this end, Kaliakatsos *et al.* [211] proposed an accompaniment system that employs Differential Evolution and Genetic Algorithms for producing the accompanying music. Another approach to the task of real-time music generation for jazz improvisation, was presented by Hutchings and McCormack in [212], where they implemented a composite system with a LSTM-based melody agent that was trained on chord progressions of jazz "standard" compositions, and a rule-based harmony agent that manipulates precomposed melodies for improvising new themes and variations. The composition flow between the agents is controlled by a rating system that rewards harmonic consistency and melodic continuity.

The objective of our work in this Chapter is to explore the attributes of real-time musical accompaniment that an artificial agent can offer to a human improviser within a context similar to traditional forms of jazz improvisation. This involves adhering to predetermined harmonic sequences and metrical structures as constraints. Software tools and methods that are able to generate "static" accompaniment to human soloists, exist for a long time [213]. On the other hand, the scenario that we examine in our study includes "spontaneous" alterations in accompaniment responses of an artificial agent both in terms of rhythm and harmony, based on the improvisation of a human soloist. The algorithmic cornerstone of our approach relies on LSTM-based architectures. The motivation for pursuing and studying such an approach in modeling human-machine improvisation and the reasons for choosing to examine basic deep learning neural networks as an algorithmic backbone is analyzed in the following section.

6.2 Research Scope and Contributions

In music, "masterful" violation of anticipation has been identified as key component for the emergence of emotion, meaning, concepts and overall interest [214]. Furthermore, anticipation is shaped by the exposure to stimuli with common characteristics, a fact that induces relations between fundamental mechanisms of music understanding and statistical learning [214]. The basic principles of jazz improvisation evolve around the violation of expectation, with improvising musicians constantly attempting to introduce meaningful novelty in the way they express themselves, and communicate with other musicians in real-time. Therefore, jazz improvisation could be described as an exemplar for studying the core-mechanism of music cognition, which is the interplay between anticipation and its violation.

Communication between improvising musicians is a key element for achieving interesting and meaningful improvisations. Especially in jazz improvisation, each musician assumes multiple roles that are of particular relevance to our study. The key characteristics of these roles can be summarized as follows:

1. *Preserve harmonic and rhythmic characteristics of a piece*. Typical jazz improvisation incorporates a standard jazz melody with a fixed harmonic description in a fixed metric structure. However, these components are expected to be creatively altered by improvising musicians (usually not the metric structure though), towards creating meaningful violations of anticipation on the overall harmonic and rhythmic domain. For instance, chord substitutions are usual, either by introducing chords that include alternate voicings, extensions or even by including new chords altogether (e.g. tritone substitution).
2. *Express original ideas*. Violation of harmonic/rhythmic expectations is considered to come "with a reason". A common approach for soloists to attempt to build new musical phrases when improvising, is by creatively modifying and combining "standard" jazz licks, a fact that helps towards building and violating anticipation. Jazz licks in the muscle memory of the soloist are products of statistical learning, built through practicing and listening multiple jazz pieces, excerpts and phrases.
3. *Communicate musically with the improvisation/accompaniment of*

other musicians. In a broad sense, the role of the accompanist is to highlight musical choices of the soloist, or even further, understand the intentions of the soloist and improvise accompaniments accordingly. Therefore, communication, on the side of the accompanist, includes predicting the intentions of the soloist and preparing the response in a timely manner, given that proper accompaniment needs to be provided concurrently with the solo. Jazz musicians, as musicians in any other music genre, develop a common perception that can be described as the integration of a "similar" statistical model, both in the soloist and the accompanist. This model allows the accompanist to roughly predict the imminent soloist choices during improvisation.

To this end, an artificial agent that is able to perform *basic* musical accompaniment in real-time under the aforementioned setting needs to have: (i) the ability to comply with harmonic and metrical constraints set by an input chart; (ii) a model of anticipating for imminent actions of the human soloist; (iii) a dictionary of accompanying voicings for given chords that is rich enough for producing diverse/interesting accompaniment; and (iv) the ability to "adapt" its playing style to the anticipated choices of the human soloist, both in terms of voicings and rhythm. Since the problem description incorporates statistical learning and given the fact that deep neural networks have exhibited impressive capabilities in capturing the prominent statistical behavior in large amounts of training data, our study examines the incorporation of such ML approaches for the task at hand. Therefore, the research questions revolve around the suitability of deep learning methods for the described improvisation setting, under the methodological framework that is presented in Section 6.3. These questions are formulated as follows:

- Is the considered ML system able to capture "static" harmonic information of a given chart in a setting of "dynamic" constraints (i.e. changing playing style of the human soloist)?
- To what extent is the proposed system responding to "dynamic" components introduced by the human agent?
- Is the examined setup suitable for real-time performance, both in terms of robustness and computational feasibility?

Recent advances in deep learning include the development of systems that are able to generate music that adapts to pre-configured constraints. In general terms, such systems either compose music *sequentially* or *non-sequentially*. Sequential systems (e.g. as the one presented by [215]), provide decisions for each note depending only on previous notes, with additional potential constraints in place. Regarding non-sequential systems, such as "DeepBach" [198], new notes are inserted by sampling, forming "dynamic" constraints for notes that are inserted later on, regardless of time precedence. In other words, notes at the end of the piece could be inserted at an earlier stage, than notes that appear earlier in the piece, depending on randomly sampled priorities. In one sense, a system that is able to perform real-time accompaniment, as described in our study presented herein, needs to be able to both compose sequentially (since time moves forward while performing) and comply with constraints that change as the composition is constructed (since the human soloist is expected to violate the expectations reflected by the solo predictive model).

The main contribution of this paper is that it studies the characteristics of a complex, multi-layered neural network where both static and dynamic components are combined for preforming predictions. The real-time improvisation setup discussed herein offers a well-defined platform of experimentation with potential interest for real-world applicability.

6.3 Materials and Methods

The proposed system provides real-time accompaniment to a human musician, based on a given harmonic description of lead sheet chord symbols. The role of the system is to reflect harmonic information as given in the lead sheet and also interpret this information with variability, responding to the predicted implied harmonic variability of a human solo. Hence, training data for our system need to include information about:

1. *Metric structure*, for letting the system become aware of measure changes.
2. *Lead sheet information*, for learning to comply with given lead sheet chords.
3. *Human solo channel*, for learning to respond to what the human soloist is expected to play.

4. *Accompaniment channel*, for learning to play proper accompaniment chords/voicings over the given lead sheet chords.

Up to our knowledge, such a dataset containing all the aforementioned properties is missing from the research community. To this end, Section 6.3.1 describes the processes for constructing our dataset by starting off with an initial dataset collected from online resources that covers most of the requirements. Next, we present the proposed system that incorporates two layers of information processing, starting with a dedicated computational model for predicting the imminent steps of the human performance, followed by a second model for integrating this prediction along with other static constraints (i.e. metric and lead sheet information), towards providing the final chord accompaniment prediction.

6.3.1 Data Preparation

The initial dataset[2] [194] contains all necessary information about the pieces, including tempo, beat, melody and the chords on a lead sheet. It should be noted that only lead sheet information is included in this dataset without actual notation of the accompaniment chords. In order to address this issue we performed a harmonic enrichment procedure that is described in detail later in this section. Furthermore, we use the beat information to indicate the start of a measure. A single time-step corresponds to the $1/24$ of a quarter note, a time resolution which is fine enough to even represent rhythm values of $64th$ triplets. The melody and the accompanying chords are represented as 128-key piano rolls with the aforementioned time resolution, where each active note at each time instance is annotated with the respective velocity value. With this representation however, the information about a note repetition is potentially obscured. For instance, there is no differentiation between a single note/chord of a quarter duration (24 time-steps) and two successive notes/chords with a duration of an eighth per note (12 time-steps). A time resolution reduction from 24 steps per beat (quarter) to 2 steps per beat was performed, such that each time-step was represented by $1/2$ of a quarter note, which is an eighth note. In other words, from each beat (24 time-steps) we only kept the melodic information of the $1st$ and $13th$ time-step, by splitting each quarter (24 time-steps) in

[2] https://github.com/wayne391/lead-sheet-dataset/

half. Thus keeping only the first of each of the two subsets of time instances (12 time-steps).

In order to construct a suitable and compact representation of chord information in the form of a jazz standard lead sheet, we use the information extracted from the accompanying chords channel of the initial dataset. Specifically, instead of keeping the velocity values of the chord notes and their MIDI value, we only kept the pitch class of their root, as well as the type of those chords, by using ready-made functions from the Music21 Python library[3] [216], which contains a set of tools for computer-aided musicology. Moreover, we chose to represent the jazz standard chord information as a binary vector of size 15, where the first 12 bits represent the root pitch class information, while the remaining 3 bits represent major/minor $3rd$, perfect/augmented/diminished $5th$ and major/minor $7th$ respectively. The reason for performing such an abstraction for representing chord information on the lead sheet is motivated by the fact that jazz musicians need a fundamental description of harmony, which they can manipulate/alter in a creative manner. The employed scheme allows for basic chord types to be represented, such as major/minor triads, dominant/7, major 7th, (half) diminished and augmented.

As mentioned earlier, the initially obtained dataset includes information only about lead sheet chords, without specific notation of actual accompanying chords. Hence, we constructed the actual accompaniment chords algorithmically by applying a basic "harmonic enrichment" process, where the lead sheet chords are transcribed into actual accompanying chords with different inversions and diverse rhythmic patterns. The enrichment process begins by assigning accompaniment chords to positions of lead sheet chord symbols. Next, inverted chords are probabilistically inserted after the initially placed chords. Aim of this process is to introduce rudimentary variability in the accompaniment channel, based on the lead sheet chord symbols and the melodic rhythm. The likelihood of inserting a chord at a particular position on the score is influenced by two factors:

1. The duration of time without a chord event (the longer the duration, the higher the probability).
2. The presence of a melodic note event (melody notes enhance the probability of chord insertion).

[3] `https://web.mit.edu/music21`

Since the melody channel is monophonic, the 128-sized vector representation of each note in the melody channel is flattened to its single non-zero value (the actual MIDI value of that note). For the accompaniment channel (i.e. the actual notes that the system is intended to learn), a dictionary of all the unique chords in the training set is created, and each chord is represented by its index in the dictionary. Practically, the "flattened" values for both the melody and the accompaniment parts allow us to apply one-hot representation of the respective data streams. Prior to the harmonic enrichment step, the initial accompaniment chord dictionary consisted of 476 chord classes. After augmentation and before transposing to all possible 12 pitches, the number of chord classes increased to 847. Finally, following the complete data preparation procedure, including augmentation and transposition, we obtained a total of 2677 unique accompaniment chord classes.

6.3.2 System Architecture and Real-time Considerations

As mentioned earlier, the generated accompaniment part should be influenced by the soloist's intended future melody notes. To achieve this, our proposal comprises two sub-systems: the Human Agent RNN (HA-RNN) and the Artificial Agent RNN (AA-RNN). These sub-systems leverage the effectiveness of RNNs with LSTM gates for modeling sequential information. An overview of the proposed system is presented in Figure 6.1.

The overall system receives as input successively overlapping windows comprising 16 time-steps, representing events within a time resolution of eighth notes. The window slides one step/eighth note at each iteration, which occurs in every eighth successively. Information for each time-step includes:

1. The metric information (b_t).
2. The soloist's melodic/solo part (h_t).
3. The accompaniment chords that are expected to be learned from the system (m_t).
4. The chord information in the abstract lead sheet style described in Subsection 6.3.1 (c_t).

Since the HA-RNN is responsible for predicting the solo melody of the successive time-step (h_{t+1}), it excludes the accompaniment channel from

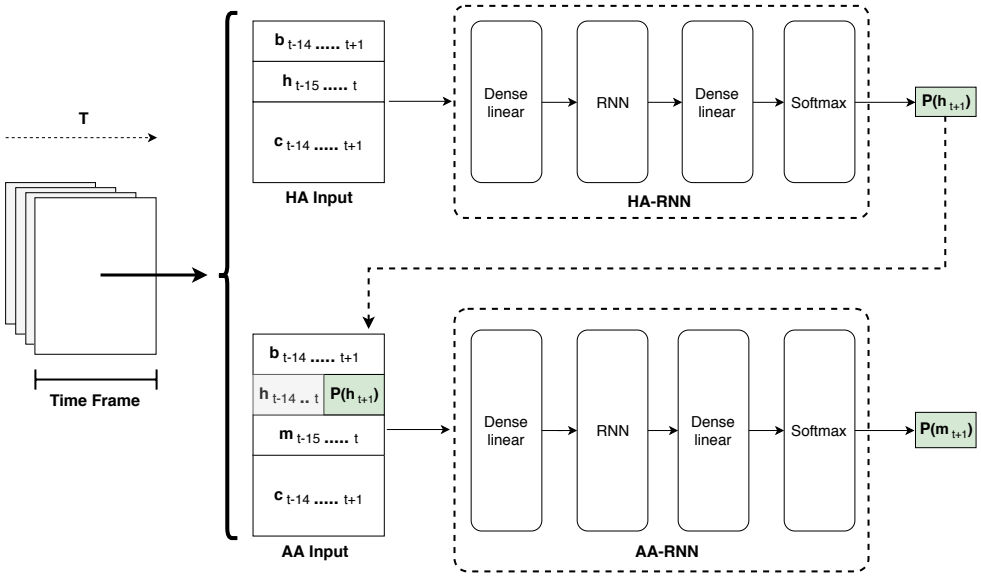


Figure 6.1: A detailed overview of the proposed system architecture. Consecutive overlapping time-frames are processed by the two sub-systems. The HA-RNN predicts the soloist melody that is later used by the AA-RNN for predicting the accompaniment chords for the following time-step.

its input, while having the beat and chord information channels one eighth ahead from the current melody. On the other hand, the AA-RNN takes under account all the information channels, in addition to the predicted $P(h_{t+1})$ of the HA-RNN, in order to anticipate the accompaniment chord for the future eighth step (m_{t+1}). Both agents at their core, implement a similar neural network architecture. Firstly, the input time frame is processed by the bottom "Dense linear" (fully connected) layer, where it gets embedded to a fixed size dimension through a linear transformation. Next, the embedded output is further encoded into a latent space through the LSTM RNN layer. Then, the top "Dense linear" layer receives the encoded LSTM output and applies a linear transformation to a space with dimensionality equal to the number of the target classes. Finally, the output of the top fully connected layer passes through a softmax function, resulting to a probability distribution for the target classes ($P(h)$ and $P(m)$). The final prediction is the class with the highest probability.

As a proof-of-concept, we train a basic system with batches of 128 samples. The embedding dimension of the bottom fully connected layer is equal to the size of the feature dimension of the input frame, whilst the RNN layer involves 64 LSTM cells. We used the Adam optimization algorithm [106] for

the minimization of the cross entropy cost function with a learning rate of 0.001. Both the HA-RNN and AA-RNN architectures were implemented using the TensorFlow 2.0 framework [217] and trained for at least 1200 epochs on a computer equipped with one NVIDIA Tesla K40c GPU, an Intel Core i7-5820K CPU at 3.30 GHz and 32 GB DDR4 RAM at 2133 Mhz.

With the aforementioned experimental setup, we observed that the average time of the overall system to predict an accompaniment chord was around $0.66\,\mathrm{ms}$ ($0.31\,\mathrm{ms}$ for the HA in addition to $0.35\,\mathrm{ms}$ for the AA). This fact indicates the feasibility of the proposed system to be adopted in real-time applications, however a thorough evaluation of the real-time capabilities of the presented method needs to be examined as future work. In this regard, we developed a prototype web application based on MIDI.js[4] and TensorFlow.js[5] javascript libraries for testing the adaptability of the proposed model to the user's soloing input in a real-time setting. An example of the GUI of the web application is illustrated in Figure 6.2. On the top side we visualize the solo improvisation performed by the user on the piano keyboard, which is placed in the middle of the screen. On the bottom side, the user can specify the chord progressions according to the selected time signature and number of bars. The accompaniment system starts to provide predictions by clicking on the "Play" button. The system implementation, source code of the LSTM models, and the real-time web application are available online[6].

6.4 Evaluation and Results

The results are oriented towards answering the research questions given in Section 6.2, i.e. whether and to what extent is the system able to capture the harmonic lead sheet constraints, to what extent is the system influenced by different soloing styles and what are possible limitations for applying this approach in real-time settings with currently available technologies. To this end, two test jazz standards, namely *"All of Me"* and *"Au Privave"*, are examined in different and diverse artificial improvisation settings, that simulate two extreme scenarios, where the human player:

[4] `https://github.com/mudcube/MIDI.js/`
[5] `https://www.tensorflow.org/js`
[6] `https://github.com/kosmasK/JazzICat`

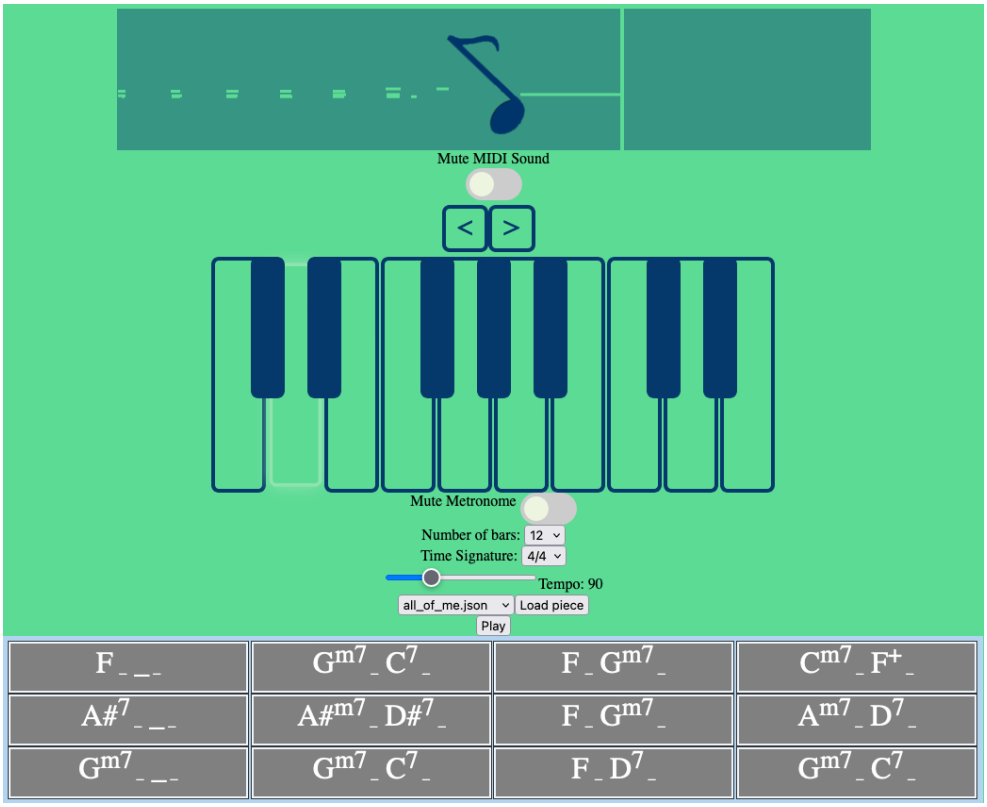


Figure 6.2: The web interface developed as a proof-of-concept to test the deployment of the proposed system in real-time setting.

1. is not playing any note during the solo performance (consecutive occurrences of pause events); and
2. is playing random notes within two octaves (as a form of extremely complex improvisation).

The responses of the system under these two settings for each piece are analyzed for different epochs of training (randomly sampled across all training epochs), providing insights about how harmonic compliance is varied and how the existence of a solo affects system responses (adaptability) at different stages of training. Since technical limitations led to building a system with limited computational power (incorporating solely a single LSTM layer with few neurons for the artificial agent) and keeping time resolution to eight notes, getting useful feedback from musicians through exhaustive real-time experiments was not possible. In this regard, a preliminary empirical evaluation based on listening tests was conducted by comparing gener-

ated and original accompaniments. We maintain, however, that the results presented herein indicate that employing more sophisticated architectures for at least the part of the artificial agent, would lead to a system that both adapts to the playing style of the user and preserves harmonic consistency according to the given lead sheet.

6.4.1 Compliance with lead sheet harmony

This section examines the ability of the system to play chords that correspond to the chord symbols on the lead sheet chart. This part of the study concerns the compliance with the basic harmonic guidelines provided by the chart and, hence, comparison is presented on the level of pitch class sets. To this end, the lead sheet chart chords are translated to their corresponding pitch classes as well as the interpretations of the system. To obtain insight on how training epochs influence the harmonic compliance of the system, results are taken from an early and a late epoch of training (59 and 1251). Tables 6.1 (epoch 59) and 6.2 (epoch 1251) we present the chord symbols and the responses of the system, regarding the "All of me" jazz standard, when no solo (top) and random solo (bottom) was provided. Similarly, Tables 6.3 and 6.4 show the responses of the system in "Au Privave" with and without random solo.

Regrading "All of Me", we can observe in Table 6.1 that in most cases the exact harmonic description in the lead sheet chart is reflected by the system. Initially, it should be noted that harmonic deviations mostly concern the first few starting measures of each piece, where the system has not incorporated any memory in its decisions. The beginning chord of the chart, $[0, 4, 7, 11]$, appears to have the most alterations, some of which are clearly erroneous (e.g. the $[4, 6, 8, 10, 11]$ interpretation that was composed for random solo). Figure 6.3a shows the first 8 measures and Figure 6.3b measures 33 to 40, composed by the system for "All of Me" in a real-time simulation setting with random solo (the solo part is not shown). The "erroneous" choices appear to be artifacts of the initial delay of the system to catch up with the constraints and start building up harmonic memory. Figure 6.3a indicates that the first three chords shown in the lower part of Table 6.1 are a result of this delay. Other harmonic deviations concern the delay of the system in complying with "unexpected" chord changes, given that most pieces in the dataset are pop songs. For instance, some

Table 6.1: System interpretations of chart chords for “All of Me” without solo (top) and with random (bottom) solo at epoch 59, shown as pitch class sets. Numbers in parentheses show the total time-steps that a system-generated PC-set occurs under the respective chart PC-set.

No solo					
Chart chord	System interpretations				
[0, 4, 7, 11]	[0, 4, 7, 11](80)	[0, 3, 8](4)	[2, 7, 10](12)	[0, 2, 4, 5, 7](24)	
[0, 4, 7, 10]	[0, 5, 7, 10](1)	[2, 6, 11](1)	[2, 4, 6, 8](4)	[2, 4, 8, 11](1)	[0, 4, 7, 10](25)
[2, 4, 8, 11]	[2, 6, 9, 11](1)	[2, 4, 8, 11](185)	[0, 4, 7](4)	[0, 4, 7, 10](18)	
[1, 4, 7, 9]	[1, 4, 7, 9](168)	[2, 5, 9](4)	[2, 7, 10](4)		
[2, 5, 9]	[2, 5, 9](128)				
[0, 4, 9]	[0, 4, 9](64)				
[0, 2, 6, 9]	[0, 2, 6, 9](64)				
[0, 2, 5, 9]	[1, 4, 7, 9](8)	[0, 2, 5, 9](72)			
[2, 5, 7, 11]	[2, 5, 7, 11](79)				
[0, 5, 9]	[0, 5, 9](32)				
[0, 5, 8]	[0, 5, 9](4)	[0, 5, 8](28)			
Random solo					
Chart chord	System interpretations				
[0, 4, 7, 11]	[4, 6, 8, 10, 11](1)	[4, 6, 8, 11](3)	[2, 6, 11](3)	[0, 4, 7, 11](80)	
		[0, 3, 8](4)	[2, 7, 10](12)	[0, 2, 4, 5, 7](24)	
[0, 4, 7, 10]	[0, 4, 7, 10](32)				
[2, 4, 8, 11]	[0, 4, 7](4)	[2, 4, 8, 11](186)	[0, 4, 7, 10](18)		
[1, 4, 7, 9]	[1, 4, 7, 9](168)	[2, 5, 9](4)	[2, 7, 10](4)		
[2, 5, 9]	[2, 5, 9](128)				
[0, 4, 9]	[0, 4, 9](64)				
[0, 2, 6, 9]	[0, 2, 6, 9](64)				
[0, 2, 5, 9]	[1, 4, 7, 9](8)	[0, 2, 5, 9](72)			
[2, 5, 7, 11]	[2, 5, 7, 11](79)				
[0, 5, 9]	[0, 5, 9](32)				
[0, 5, 8]	[2, 5, 9](4)	[0, 5, 8](28)			

misinterpretations of the E7 chord ($[2, 4, 8, 11]$) are a result of delay in “comprehending” the unexpected change. This effect can be noted in the third bars of both parts of Figure 6.3. System-generated chords for “Au Privave” follow a similar pattern in terms of harmonic compliance, but with fewer erroneous harmonic deviations, as evident in Table 6.3.

6.4.2 Variability

The chords generated by the system in each improvisation setting for each piece are expected to be different, since different improvisations from the human soloist should trigger different responses. These differences are examined by direct comparison of the system generated chords for the two improvisation modes (i.e. the chords generated by the system without human solo, and with a random solo). A general figure that describes the differences between the system-generated chords in both examined pieces with (random) and without solo, is given by computing the percentage of chords that are different per time-step for accompaniment sessions comprising four repetitions of the entire chart, with (random) and without solo.

Table 6.2: System interpretations of chart chords for “All of Me” without solo (top) and with random (bottom) solo, shown as pitch class sets. Numbers in parentheses show the total time-steps that a system-generated PC-set occurs under the respective chart PC-set.

No solo

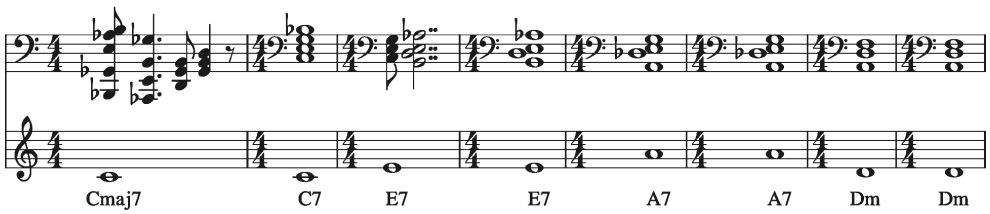
Chart chord	System interpretations				
[0, 4, 7, 11]	[3, 5, 8, 11](2)	[0, 2, 9, 10](1)	[2, 4, 9](1)	[1, 3, 10, 11](1)	[1, 3, 6, 10](1)
	[0, 3, 8](1)	[2, 5, 10](1)	[0, 4, 7, 11](103)	[3, 7, 10](13)	[0, 5, 9](3)
[0, 4, 7, 10]	[0, 3, 8](2)	[2, 5, 10](1)	[0, 4, 7, 10](23)	[2, 5, 7, 11](3)	[2, 7, 11](3)
[2, 4, 8, 11]	[2, 4, 8, 11](162)	[4, 8, 11](1)	[4, 7, 11](4)	[0, 3, 5, 9](1)	[3, 6, 8, 11](1)
	[1, 4, 9](1)	[1, 4, 7, 9](12)	[1, 4, 6, 10](3)	[2, 6, 8, 11](3)	[2, 5, 9](6)
	[0, 2, 5, 9](3)	[0, 2, 6, 9](3)			
[1, 4, 7, 9]	[1, 4, 7, 9](124)	[2, 4, 7, 11](9)	[1, 2, 6, 9](11)	[2, 4, 8, 11](7)	[1, 3, 10, 11](4)
	[2, 6, 8, 11](4)	[5, 8, 11](3)	[2, 8, 11](3)		
[2, 5, 9]	[2, 5, 7, 9](15)	[2, 7, 10](4)	[2, 5, 9](94)	[1, 3, 10, 11](3)	[1, 5, 8, 10](3)
	[3, 7, 10](6)				
[0, 4, 9]	[0, 4, 9](12)	[3, 6, 10](1)	[1, 6, 9](1)	[4, 8, 11](1)	[0, 2, 4, 9](45)
	[2, 5, 9](3)				
[0, 2, 6, 9]	[0, 2, 6, 9](54)	[0, 4, 9](5)	[0, 3, 6, 10](1)	[0, 5, 8](1)	[0, 3, 5, 8](1)
	[2, 5, 10](1)				
[0, 2, 5, 9]	[2, 5, 10](13)	[0, 2, 5, 9](36)	[0, 3, 7, 10](2)	[3, 7, 10](1)	[1, 3, 4, 11](8)
	[0, 5, 8](12)	[2, 4, 7, 11](4)	[2, 5, 7, 10](4)		
[2, 5, 7, 11]	[2, 5, 7, 11](65)	[2, 3, 7, 10](4)	[0, 3, 7](4)	[0, 5, 8](3)	[2, 5, 9, 10](3)
[0, 5, 9]	[0, 4, 5, 9](28)	[2, 5, 9](4)			
[0, 5, 8]	[0, 5, 8](28)	[3, 7, 10](4)			

Random solo

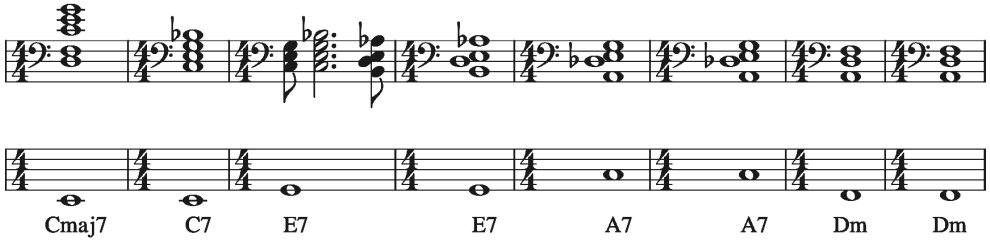
Chart chord	System interpretations				
[0, 4, 7, 11]	[3, 5, 8, 11](1)	[1, 3, 6, 8](2)	[2, 4, 9](1)	[0, 4, 5, 9](1)	[1, 3, 6, 10](1)
	[2, 7, 10](1)	[0, 5, 9](5)	[0, 4, 7, 11](86)	[3, 7, 10](5)	[0, 2, 4, 5, 7](14)
	[0, 4, 7](2)	[2, 4, 5, 9](1)	[1, 5, 8, 11](1)	[2, 5, 10](1)	[0, 2, 6, 9](1)
	[0, 3, 6, 8](1)	[1, 4, 9](2)	[2, 5, 9, 10](1)		
[0, 4, 7, 10]	[0, 4, 7, 10](28)	[1, 5, 8](1)	[0, 4, 5, 9](1)	[2, 7, 11](2)	
[2, 4, 8, 11]	[2, 4, 8, 11](177)	[1, 4, 7, 9](12)	[4, 8, 11](2)	[4, 7, 11](3)	[1, 4, 8, 11](2)
	[1, 6, 9](2)	[2, 4, 5, 9](1)	[3, 6, 9, 11](1)	[2, 4, 7, 11](1)	
[1, 4, 7, 9]	[1, 4, 7, 9](140)	[1, 2, 6, 9](5)	[2, 4, 8, 11](5)	[1, 4, 6, 10](4)	[1, 6, 8, 11](5)
	[0, 2, 6, 9](6)	[2, 4, 7, 11](4)	[2, 4, 5, 9](1)		
[2, 5, 9]	[2, 5, 9](95)	[2, 5, 7, 9](30)	[2, 7, 10](2)	[0, 4, 7](1)	
[0, 4, 9]	[0, 2, 4, 9](30)	[2, 5, 9](2)	[0, 4, 9](25)	[1, 3, 10, 11](2)	[2, 4, 6, 7](2)
	[4, 8, 11](1)				
[0, 2, 6, 9]	[0, 2, 6, 9](46)	[0, 4, 9](8)	[1, 3, 10, 11](2)	[2, 5, 7, 10](2)	[0, 4, 7, 9](2)
	[2, 7, 11](2)				
[0, 2, 5, 9]	[0, 2, 5, 9](49)	[1, 3, 10, 11](3)	[0, 2, 5, 8](2)	[0, 4, 7](6)	[0, 1, 5, 8](2)
	[2, 5, 7, 10](6)	[3, 7, 10](3)	[2, 6, 9](2)	[0, 3, 5, 8](2)	[2, 5, 10](2)
	[0, 5, 8](1)	[0, 2, 5, 7](1)			
[2, 5, 7, 11]	[2, 5, 7, 11](62)	[2, 5, 9, 10](2)	[2, 5, 10](8)	[2, 3, 7, 10](1)	[2, 7, 11](1)
	[5, 8, 11](1)	[0, 4, 7, 10](1)	[0, 2, 5, 9](1)	[0, 4, 7](1)	
[0, 5, 9]	[0, 4, 5, 9](28)	[2, 5, 9](4)			
[0, 5, 8]	[0, 5, 8](28)	[3, 7, 10](4)			

Table 6.3: System interpretations of chart chords for “Au Privave” without solo (top) and with random (bottom) solo at epoch 59, shown as pitch class sets. Numbers in parentheses show the total time-steps that a system-generated PC-set occurs under the respective chart PC-set.

No solo				
Chart chord	System interpretations			
[0, 5, 9]	[2, 5, 10](6)	[0, 5, 9](61)	[0, 5, 9, 10](12)	
[2, 5, 7, 10]	[2, 5, 7, 10](60)	[0, 4, 7, 9](36)	[2, 5, 9](16)	
[0, 4, 7, 10]	[0, 4, 7, 10](35)	[2, 5, 8, 10](4)	[1, 3, 7, 10](4)	[0, 2, 6, 9](4)
[0, 3, 7, 10]	[0, 3, 7, 10](16)			
[1, 3, 5, 9]	[0, 3, 5, 9](1)	[0, 3, 6, 8](2)	[5, 8, 11](9)	
[2, 5, 8, 10]	[2, 5, 10](1)	[2, 5, 8, 10](28)	[3, 6, 10, 11](3)	
[1, 5, 8, 10]	[1, 5, 8, 10](16)			
[1, 3, 7, 10]	[2, 5, 8, 10](16)			
[0, 4, 7, 9]	[2, 5, 7, 10](8)	[0, 2, 5, 9](4)	[0, 4, 7, 9](4)	
[0, 2, 6, 9]	[0, 4, 5, 9](12)	[2, 5, 9, 10](4)	[0, 2, 5, 9](4)	[0, 2, 6, 9](12)
Random solo				
Chart chord	System interpretations			
[0, 5, 9]	[2, 5, 10](3)	[0, 5, 9](73)	[0, 5, 9, 10](3)	
[2, 5, 7, 10]	[0, 5, 9](1)	[2, 5, 7, 10](83)	[0, 4, 7, 9](12)	[2, 5, 9](16)
[0, 4, 7, 10]	[0, 4, 7, 10](46)	[0, 2, 6, 9](1)		
[0, 3, 7, 10]	[0, 3, 7, 10](16)			
[1, 3, 5, 9]	[0, 3, 5, 9](5)	[0, 3, 6, 8](6)	[2, 6, 9, 11](1)	[2, 5, 9, 11](2)
[2, 5, 8, 10]	[2, 5, 10](1)	[2, 5, 8, 10](31)		
[1, 5, 8, 10]	[1, 5, 8, 10](16)			
[1, 3, 7, 10]	[2, 5, 8, 10](4)	[1, 3, 7, 10](12)		
[0, 4, 7, 9]	[0, 4, 7, 9](16)			
[0, 2, 6, 9]	[0, 2, 5, 9](2)	[0, 2, 6, 9](30)		



(a) Bars 1-8.



(b) Bars 33-40.

Figure 6.3: First 8 measures (a) and measures $33 - 40$ (b) of system-generated chords over the respective lead sheet chords for “All of Me” with random solo part (omitted in the depiction).

Table 6.4: System interpretations of chart chords for “Au Privave” without solo (top) and with random (bottom) solo at epoch 1251, shown as pitch class sets. Numbers in parentheses show the total time-steps that a system-generated PC-set occurs under the respective chart PC-set.

No solo

Chart chord	System interpretations				
[0, 5, 9]	[1, 3, 10, 11](5)	[0, 3, 5, 9](5)	[0, 3, 7, 8](1)	[3, 6, 11](1)	[2, 6, 8, 11](1)
	[1, 3, 7, 10](1)	[0, 5, 8](1)	[0, 3, 8](1)	[2, 7, 11](1)	[0, 5, 9](51)
	[5, 8, 11](4)	[2, 7, 10](4)	[0, 5, 7](3)		
[2, 5, 7, 10]	[2, 5, 8, 10](2)	[2, 5, 7, 10](59)	[0, 1, 5, 8](1)	[2, 3, 7, 10](1)	[0, 5, 8](1)
	[1, 3, 10, 11](4)	[1, 4, 6, 9](4)	[0, 4, 7, 10](10)	[1, 4, 7, 9](4)	[0, 3, 7](4)
	[2, 5, 10](4)	[0, 2, 5, 7](12)	[0, 2, 5, 9](3)	[0, 2, 7, 10](3)	
[0, 4, 7, 10]	[0, 3, 7](16)	[2, 5, 10](1)	[1, 3, 6, 10](1)	[0, 4, 7, 10](24)	
[0, 3, 7, 10]	[0, 3, 7, 10](16)				
[1, 3, 5, 9]	[0, 3, 5, 9](1)	[0, 1, 5, 8](8)	[1, 3, 7, 10](3)		
[2, 5, 8, 10]	[3, 5, 8, 11](4)	[0, 3, 7](7)	[2, 7, 10](3)	[2, 5, 8, 10](4)	[2, 5, 10](7)
	[0, 5, 8](4)	[2, 5, 9](3)			
[1, 5, 8, 10]	[3, 5, 7, 10](10)	[1, 3, 7, 10](6)			
[1, 3, 7, 10]	[1, 3, 7, 10](2)	[0, 3, 8](4)	[0, 5, 8](4)	[3, 7, 10](6)	
[0, 4, 7, 9]	[0, 4, 7, 9](16)				
[0, 2, 6, 9]	[0, 2, 6, 9](12)	[0, 3, 7](8)	[0, 5, 8](4)	[0, 5, 7, 8](4)	

Random solo

Chart chord	System interpretations				
[0, 5, 9]	[1, 3, 10, 11](4)	[2, 5, 7, 10](5)	[3, 5, 8, 10](1)	[1, 3, 7, 10](1)	[1, 3, 5, 8](1)
	[1, 5, 8](1)	[3, 5, 7, 10](2)	[2, 5, 10](2)	[3, 6, 11](6)	[0, 4, 5, 9](11)
	[2, 5, 8, 10](5)	[0, 3, 7](2)	[0, 5, 9](17)	[3, 7, 10](4)	[4, 8, 11](3)
	[0, 3, 5, 9](3)	[3, 5, 8, 11](2)	[0, 4, 7, 10](2)	[0, 5, 8](3)	[0, 3, 8](1)
[2, 5, 7, 10]	[1, 3, 7, 10](2)	[0, 3, 7](1)	[0, 3, 8, 10](1)	[2, 6, 8, 11](1)	[2, 5, 7, 10](57)
	[5, 6, 8, 11](2)	[0, 3, 5, 9](2)	[2, 5, 9](7)	[1, 3, 10, 11](4)	[1, 4, 6, 9](3)
	[1, 5, 8, 10](1)	[2, 3, 7, 10](3)	[1, 4, 9, 11](2)	[1, 4, 6, 10](3)	[0, 5, 7, 9, 10](1)
	[0, 5, 8, 10](1)	[1, 4, 9](1)	[0, 5, 9](1)	[2, 5, 10](4)	[5, 8, 11](1)
	[0, 2, 3, 5, 10](1)	[3, 6, 8, 11](1)	[0, 2, 5, 9](1)	[0, 3, 7, 8](2)	[0, 5, 8](1)
	[2, 7, 11](1)	[2, 5, 9, 10](1)			
[0, 4, 7, 10]	[0, 3, 7](6)	[3, 5, 7, 8](1)	[0, 5, 8](2)	[0, 4, 7, 10](18)	[0, 3, 8](1)
	[2, 5, 10](3)	[1, 4, 6, 9](1)	[1, 3, 6, 10](1)	[3, 6, 8, 11](1)	[2, 7, 10](1)
	[0, 3, 7, 8](2)				
[0, 3, 7, 10]	[0, 3, 7, 10](11)	[1, 4, 9, 11](1)	[1, 3, 10, 11](1)	[3, 7, 10](2)	[0, 1, 5, 8](1)
[1, 3, 5, 9]	[0, 3, 5, 9](5)	[1, 4, 6, 10](1)	[2, 6, 8, 11](1)	[0, 3, 5, 8](1)	[1, 3, 7, 10](1)
	[0, 5, 8](1)	[0, 3, 7](3)			
[2, 5, 8, 10]	[2, 7, 10](1)	[2, 5, 8, 10](19)	[0, 5, 8](5)	[1, 3, 7, 10](1)	[3, 7, 10](1)
	[2, 5, 10](3)	[2, 5, 7, 10](2)			
[1, 5, 8, 10]	[1, 3, 6, 10](1)	[1, 3, 5, 10](3)	[1, 3, 7, 10](2)	[3, 5, 7, 10](3)	[1, 5, 8, 10](6)
	[2, 5, 8, 10](1)				
[1, 3, 7, 10]	[1, 3, 7, 10](8)	[0, 5, 8](1)	[3, 6, 10](1)	[3, 7, 10](4)	[0, 3, 8](1)
[0, 4, 7, 9]	[0, 4, 7, 9](12)	[0, 2, 3, 5, 10](1)	[2, 5, 7, 10](1)	[1, 3, 5, 6, 8](1)	
[0, 2, 6, 9]	[0, 2, 6, 9](10)	[0, 3, 5, 9](2)	[0, 3, 7](5)	[0, 5, 8](3)	[0, 2, 5, 9](3)
	[2, 5, 7, 11](1)	[0, 1, 3, 8](1)			

Regarding “All of Me”, only 2% of system-generated chords are different between random and no solo for epoch 59, which increases to 60% in epoch 1251, indicating that the system decisions are affected slightly by the presence of a solo in early epochs, while the effect of solo is more evident as epochs progress. In “Au Privave” this percentage starts from 74% during epoch 59, and gets to 84% at epoch 1251, showing that system generations are more sensitive to the presence of a chord solo for this piece.

For observing the differences within each improvisation session, the system-generated chords in four repetitions of the entire chart are examined repetition-by-repetition, thus forming four quarters of the entire composition, referred to as “quartiles”. Tables 6.5-6.7 and 6.6-6.8 show the quartile similarities without and with random solo, for “All of Me” (epochs 59 and 1251) and “Au Privave” (epochs 59 and 1251) respectively. In “All of Me”, and especially in the scenario with an absence of solo, both in the early and the late epoch of training, only the first repetition is different from the remaining three, as show in the first rows and columns of both matrices in Tables 6.5 and 6.7. The insertion of the random solo does not influence the overall result in the early epoch (bottom matrix in Table 6.5), but for the late epoch the influence is evident (bottom matrix in Table 6.7). Therefore, the example of “All of Me” shows that training the system for more epochs allows some sense of responsiveness to human input, as evident by the variability that emerged from the random solo. On the other hand, in “Au Privave”, the incorporation of the random solo (see Table 6.6) influences each repetition even from early training epochs, therefore creating different variations of the chart in each of the four iterations (except repetition 3 and 4 that differ only by 1%); Variations for this test piece are even more evident in the more progressed training epoch (see Table 6.8).

A final examination of variability in the generated chords is performed by measuring the number of different voicings per chord symbol on the chart. This is a more detailed examination of how the pitch class sets presented in Tables 6.1-6.2 and 6.3-6.4, are further split down in voicing layouts, i.e. what is the variability in terms of inversions and note doublings in the chords generated by the system. Figure 6.4 illustrates the average number of different voicings composed by the system for each chord label in the chart, in form of errorbars for some random epochs sampled across all training epochs. As it regards “All of Me” (see Figure 6.4a), each chord symbol in the chart is materialized with approximately 2.5 different voicing

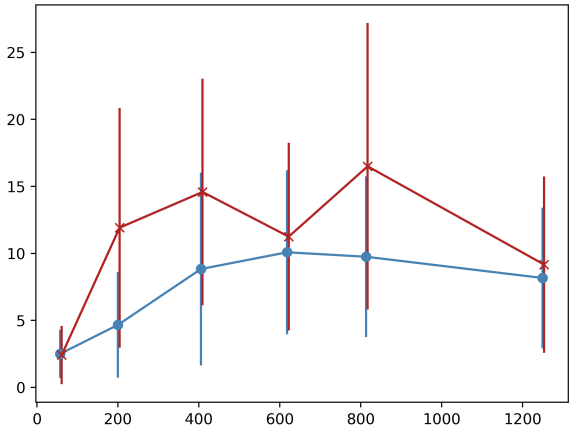
Table 6.5: "Quartile" similarity in system-generated chords in "All of Me" without (top) and with random solo (bottom) at epoch 59.

No solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	0.09	0.09	0.09
2nd qrt	0.09	0.00	0.00	0.00
3rd qrt	0.09	0.00	0.00	0.00
4th qrt	0.09	0.00	0.00	0.00
Random solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	0.05	0.05	0.06
2nd qrt	0.05	0.00	0.00	0.00
3rd qrt	0.05	0.00	0.00	0.00
4th qrt	0.06	0.00	0.00	0.00

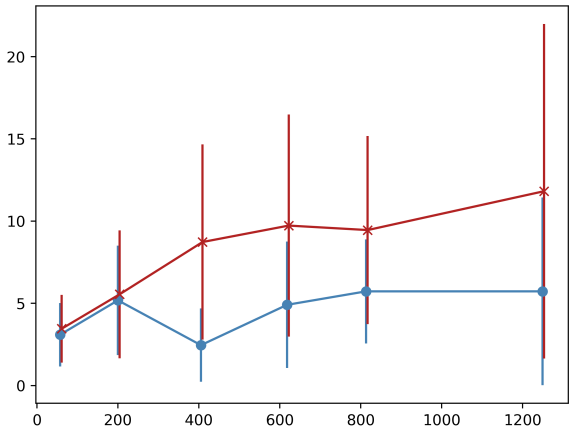
Table 6.6: "Quartile" similarity in system-generated chords in "Au Privave" without (top) and with random solo (bottom) at epoch 59

No solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	0.34	0.34	0.35
2nd qrt	0.34	0.00	0.00	0.01
3rd qrt	0.34	0.00	0.00	0.01
4th qrt	0.35	0.01	0.01	0.00
Random solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	1.00	0.99	0.99
2nd qrt	1.00	0.00	0.33	0.34
3rd qrt	0.99	0.33	0.00	0.01
4th qrt	0.99	0.34	0.01	0.00

implementations in epoch 59, almost regardless of the presence of solo (red 'x' indicates presence of random melody and blue circle absence thereof). The system presents increased voicing variability dependence on human solo input for this piece as the epochs increase. In the case of "Au Privave" (see Figure 6.4b), the tendency of the system to become more dependent on human input becomes more evident as epochs increase. The error value of the objective function on the validation set during training is presented in Figure 6.5. The typical decrease that is observed, indicates that there is a relation between error loss and system adaptability to human input, in other words, better data fitting leads to further variability.



(a) All of me



(b) Au Privave

Figure 6.4: Error-bars of different voicings employed by the system for each chord label in the chart across a sampled set of epochs.

Table 6.7: "Quartile" similarity in system-generated chords in "All of Me" without (top) and with random solo (bottom) at epoch 1251.

No solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	0.53	0.53	0.54
2nd qrt	0.53	0.00	0.00	0.00
3rd qrt	0.53	0.00	0.00	0.00
4th qrt	0.54	0.00	0.00	0.00
Random solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	0.51	0.72	0.19
2nd qrt	0.51	0.00	0.30	0.55
3rd qrt	0.72	0.30	0.00	0.72
4th qrt	0.19	0.55	0.72	0.00

Table 6.8: "Quartile" similarity in system-generated chords in "Au Privave" without (top) and with random solo (bottom) at epoch 1251.

No solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	0.40	0.40	0.41
2nd qrt	0.40	0.00	0.00	0.01
3rd qrt	0.40	0.00	0.00	0.01
4th qrt	0.41	0.01	0.01	0.00
Random solo				
	1st qrt	2nd qrt	3rd qrt	4th qrt
1st qrt	0.00	0.65	0.96	0.70
2nd qrt	0.65	0.00	0.91	0.73
3rd qrt	0.96	0.91	0.00	0.83
4th qrt	0.70	0.73	0.83	0.00

6.4.3 Listening Tests

The dataset used to train the artificial agent, contains a broad variety of popular western music melodies with simulated accompaniments derived from an augmentation process. Furthermore, the quantitative metrics presented in the previous subsections are not capable to completely capture the perceptual quality and originality of the chord accompaniments generated by the proposed system. To this end, we carried out a subjective evaluation based on listening tests, aiming to study whether the generated accompaniments are comparable to the original chords existing in the dataset.

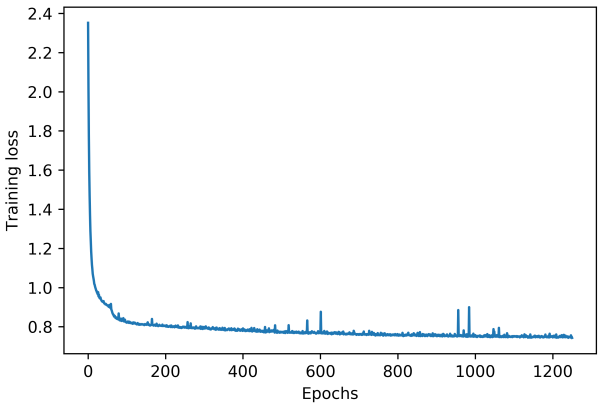


Figure 6.5: Loss of the training objective function in the validation set across multiple epochs.

For preparing the listening tests we randomly selected 10 solo melodies along with their original accompaniments from the validation set. Then we used the 10 selected melodic parts to generate their corresponding chord accompaniments with the proposed artificial agent, thus ensuring that the system receives novel input, not wielded during training. Accordingly, each participant was presented with 10 tests, where each test included 3 audio clips, starting only with the melodic part and followed by its combinations with the 2 accompaniments (original and generated), which are introduced in a random order so as to avoid any possible biases. The actual audio excerpts had a duration of around 30 seconds and looped for 6 times to reach 3 minutes of total duration. Then, the participants had to answer the following 3 questions for each accompaniment (6 questions per test) in a Likert scale from 1 (low) to 5 (high):

- **Q1:** Evaluate the overall high-level structure of the accompaniment with respect to the introduced melody.
- **Q2:** Evaluate the harmonic compliance of the accompaniment with reference to popular western music.
- **Q3:** Evaluate the rhythmical compliance of the accompaniment with reference to popular western music.

In our study 21 participants were involved, 15 male and 6 female, with the majority being 20 to 40 years old. All the participants were musicians

with different levels of expertise, having at least intermediate knowledge of music theory. Consequently, we collected a total of 1260 answers and the results are presented in Table 6.9. By inspecting only the mean values we can observe that the participants evaluated slightly better the original accompaniments in most questions. However, in order to determine whether this preference is statistically important, we performed a Wilcoxon rank sum test [107], having as null hypothesis that there is no difference between the two accompaniments. The calculated p-values demonstrated that there is statistically significant difference between the original and the generated accompaniments in examples 5, 6, 7, 9 and 10 (highlighted with bold fonts in Table 6.9), while we cannot reject the null hypothesis for the remaining examples. In other words, in 50% of the examples, we cannot be certain about whether the generated music is inferior to the original, as far as the examined qualities can define.

Overall, we can say that the accompaniments generated by the proposed artificial agent had better rhythmical compliance rather than harmonic, which might be due to the metric information that is included in the system input. Also, the poor performance in some examples indicates that the computational capabilities of a single LSTM layer are limited, thus suggesting more sophisticated architectures to be tested. We strongly encourage the reader to visit the online repository and listen to the audio files of the listening tests.

6.5 Conclusion

This Chapter presented a study on how deep neural network architectures can be employed for simulating a jazz improvisation setting between a human soloist and an artificial accompanist, based on a common chord chart. A basic implementation incorporating deep neural networks was presented and publicly available data were transformed in a way that all necessary information for the task at hand became available, i.e. information about metric structure, lead sheet chords, human-generated solo/melody and system-generated accompaniment responses. The motivation of this work is based on modeling the interplay between expectation and its violation by two improvising musicians (one human and one artificial) with implicit ML approaches (deep neural networks). The proposed methodology included the development of “a model within a model”, that allows the artificial agent to

Table 6.9: Results of our listening tests. The bold fonts indicate the statistically significant differences provided by a Wilcoxon rank sum test between the original and the generated accompaniments.

	Question	p-value	Accompaniment	Median	Mean
Example 1	Q1	0.48121	original	4.0	3.57
			generated	3.0	3.33
	Q2	0.35198	original	3.0	3.43
			generated	4.0	3.71
	Q3	0.6966	original	4.0	3.95
			generated	4.0	3.86
Example 2	Q1	0.30236	original	4.0	3.57
			generated	3.0	3.24
	Q2	0.44293	original	4.0	3.52
			generated	3.0	3.24
	Q3	0.66891	original	4.0	3.52
			generated	4.0	3.33
Example 3	Q1	0.26296	original	4.0	3.67
			generated	3.0	3.29
	Q2	0.08951	original	4.0	3.86
			generated	3.0	3.24
	Q3	0.95987	original	4.0	3.71
			generated	4.0	3.71
Example 4	Q1	0.32656	original	4.0	3.86
			generated	4.0	3.52
	Q2	0.15523	original	4.0	3.86
			generated	3.0	3.38
	Q3	0.41361	original	4.0	3.86
			generated	4.0	3.52
Example 5	Q1	**0.0543**	original	4.0	3.71
			generated	3.0	2.9
	Q2	**0.00022**	original	4.0	3.76
			generated	2.0	2.19
	Q3	**0.00766**	original	4.0	4.0
			generated	3.0	2.86
Example 6	Q1	**0.04689**	original	4.0	3.71
			generated	3.0	2.9
	Q2	**0.00025**	original	4.0	3.95
			generated	2.0	2.48
	Q3	**0.00826**	original	4.0	3.86
			generated	3.0	2.86
Example 7	Q1	**0.00007**	original	4.0	4.19
			generated	2.0	2.43
	Q2	**0.0**	original	4.0	4.19
			generated	1.0	1.62
	Q3	**0.00005**	original	5.0	4.33
			generated	2.0	2.43
Example 8	Q1	0.48907	original	4.0	3.95
			generated	4.0	3.86
	Q2	0.1159	original	4.0	4.24
			generated	4.0	3.76
	Q3	0.88003	original	4.0	3.9
			generated	4.0	4.0
Example 9	Q1	**0.03353**	original	4.0	3.76
			generated	3.0	3.0
	Q2	**0.00001**	original	5.0	4.24
			generated	2.0	2.24
	Q3	**0.00037**	original	4.0	4.1
			generated	2.0	2.57
Example 10	Q1	**0.00153**	original	4.0	3.95
			generated	2.0	2.76
	Q2	**0.0014**	original	4.0	3.71
			generated	2.0	2.29
	Q3	**0.02863**	original	4.0	3.52
			generated	2.0	2.62

have its own model of expectation for the human improviser. We also considered additional challenges, including the adaptation of large amounts of data to the desired form, leading to the development of a data enrichment process that generated variability in the accompaniment parts of the collected pieces.

Results were obtained by testing the system in 2 real-time simulation settings: (i) without any assumed human solo, and (ii) with the inclusion of a random solo. The responses of the system under these two settings in two well-known jazz standards ("All of me" and "Au Privave") indicated that harmonic compliance with the chart chords was mainly achieved, except mainly from the beginning of each accompaniment session, where the system required to "build up memory" (information context) in order to provide better accompaniment. This is possibly due to the random initialization of the states in the LSTM networks that are in the core of the presented basic implementation. Even though it was expected for the system to be influenced by the incorporation of a human solo, this was not the case in both examined pieces. Especially in "All of Me", the inclusion of a random solo did not seem to affect the output of the system, while the system-generated chords exhibited self-repetition in accompaniment sessions incorporating 4 iterations of the chart. Conversely, in "Au Privave" the inclusion of the random solo affected the system output, both by decreasing self-repetition in 4 iterations, and by increasing the number of chord voicings employed by the system for the given chart chords. In order to evaluate the perceptual quality of the generated chords, we performed subjective evaluations based on listening tests, where participants had to compare original and generated accompaniments given their corresponding melodies, by ranking their harmonic and rhythmical compliance in a liker scale. A Wilcoxon rank sum test on the responses showed that 50% of the examples were not significantly inferior to the original accompaniments.

Further research is necessary for a more thorough examination of such systems for real-time accompaniment. The results presented herein indicate that it is possible to model expectation and violation thereof for real-time jazz accompaniment with deep neural networks, however, several limitations have to be acknowledged for performing additional studies:

1. There is no proper data available with all the necessary information (lead sheet chords, metric information, solo and accompaniment). A

crucial part of the data, i.e. the accompaniment, was actually constructed algorithmically, while the solo part included melodies (rather than solos) with restricted expressional variability. The data enrichment method that was developed to construct artificial variability in the data was based on a rudimentary probabilistic implementation which is not enough for creating consistent connections that could be learned from the system.

2. The execution time of predictions might be marginally acceptable for scalable real-time systems. For the presented study, time resolution was significantly reduced for making the system safely compatible with real-time conditions, however, this fact reduced the expressional capabilities of the system. This includes not only restricted capabilities for the system responses, but also restricted capabilities for the system to identify expressional characteristics of the human soloist.
3. The prominent style found in the dataset was pop, which comprises smaller harmonic variability in comparison to jazz. Hence, the resulting accompaniment needed to undergo creative adjustments to better reflect the complex progressions found in jazz lead sheets. To delve deeper into this problem, a consistent dataset of jazz standard accompaniment sessions is essential for further study.

CHAPTER 7

Symbolic Music Encodings

7.1 Introduction

The field of *Algorithmic Music Composition* predates modern computers and the idea of AI. However, during the last twenty years it has progressed rapidly, from simple rule-based systems to complex systems (seemingly) able to compose original music. Central to this technological advancement was the implementation of deep neural networks models. Despite being a longstanding field, music generation has not experienced the same rapid advancements as disciplines such as Natural Language Processing (NLP), although it has adopted several methodologies originally developed for NLP. This can be attributed, in part, to the abstract and inherently ambiguous nature of music, which presents unique challenges compared to human language. Furthermore, the lack of standardized data encoding for musical information has also hindered progress in this field.

Current approaches often rely on transcription methods intended for human readability, such as ABC notation [218] or tablatures, or utilize communication standards like MIDI that were not specifically designed for automatic music generation purposes. The choice between these two encoding types is typically based on their historical usage and intended purpose [219]. However, it is worth noting that music data can often be effectively translated or re-encoded between the two, with minimal loss of information. This allows for the utilization of music data in contexts that differ from its original intended usage. For instance, an electronic musician may

use a Bach score re-encoded to MIDI as input for a synthesizer, or a classical ensemble may employ scores converted from video game music in the NES Sound Format (NSF) for their performances [220]. Simultaneously, it is crucial to clearly define the objective and intended usage of a system. For instance, Sturm has emphasized in multiple blog posts [221, 222] that folk-RNN[1] is specifically designed to model music transcriptions, particularly those in the ABC music format, which holds significance in the Irish folk music tradition.

In our study, we aim to avoid considerations of intended usage, such as the example mentioned above, and instead focus on encodings independently of their historical context. This approach enables us to conduct more comprehensive and general comparisons. Our research primarily focuses on notations that have been developed within the Western music tradition, spanning from the common practice period to the present. While the division between East and West is somewhat arbitrary, the term "Western music" accurately represents a historical tendency in music notation. This notation system typically incorporates octave equivalence, predominantly utilizes the 12-tone equal temperament tuning system, provides limited information regarding duration and pitch (often on a grid-like structure), and exhibits a preference for keyboard instruments. Furthermore, it treats musical compositions as unique entities. This particular type of notation, especially the traditional score, has served as a fundamental source of inspiration for the majority of the existing methods used to digitally encode music.

This Chapter specifically considers transcription methods that conceptualize music as a sequence of discrete events unfolding over time. Alternative perspectives, such as viewing a musical composition as a collection of time-independent rules, are not included in our analysis. Therefore, a music transcription is essentially represented as a sequence of events, which aligns with popular formats used to encode musical compositions, such as piano roll, tablature, scores, and MIDI. Furthermore, our focus is narrowed down to encodings where events possess a clear musical significance. While notations like ABC notation or MusicXML[2] can be seen as representing a music transcription as a sequence of characters, they often require complex interpreters to be processed by computers. On the other hand, MIDI format

[1] `https://folkrnn.org/`

[2] `https://w3c.github.io/musicxml/`

provides a straightforward interpretation of events, where note-on events initiate a note, for example. An example of this distinction is the transition made by folk-RNN v2, which shifted from a text-based encoding (pure ABC notation) to a "tokenized" version of ABC, where each token represented an interpretable event. In our view, it is highly likely that an "ideal" encoding for music generation would not resemble a sequence of musically meaningful events. Instead, it would abstract and condense crucial elements of a transcription or performance in a manner that is efficient and enhances the learning process of the music generation system. Even with the straightforward encodings utilized in our experiments, the choice of implementation can significantly impact the system's performance and characteristics. Furthermore, advancements in ML, such as learned embeddings [223, 224], have the potential to introduce novel representations for music that are not reliant on sequential structures and may not even be interpretable by humans. Nonetheless, we believe that studying simple sequential encodings is still necessary, as their resemblance to human-readable scores can offer valuable insights in the early stages of research. Despite the potential departure from these sequential encodings in the future, the examination of their properties remains relevant.

We perceive the choice of encoding not merely as another hyperparameter of computational models, arbitrarily determined in practice. Instead, it holds the potential to be an integral component of a system, with its selection guided by a theoretical and empirical understanding of its implications. Furthermore, it is important to clarify our usage of the terms "encoding" and "representation" in this Chapter, which are generally used interchangeably unless explicitly specified. The typical distinction, as proposed in related studies such as [191], is that an encoding involves the conversion of musical information of a given representation into raw data suitable for computational models. As our study encompasses both high-level and low-level organization of musical information within a programming context, we primarily employ the term "encoding", acknowledging that many of the encodings we explore herein would be referred to as "symbolic representations" in the Music Information Retrieval (MIR) research community.

The remainder of this Chapter is organized as follows. In Section 7.2 we provide an overview of the context and motivation underlying the present study. We outline our main research questions and present our general approach to addressing them. Next in Section 7.3, we explore the existing

literature in the field of AI, with a specific focus on RNN-based music generation approaches and techniques. We examine the broader landscape of music generation systems and delve into the discussion surrounding the encoding of symbolic musical data. In Section 7.4 we present an overview of various potential characteristics that are required for designing a proper musical encoding protocol. Next, Section 7.5 introduces the encoding approaches proposed in our study and categorizes them in two families. In Section 7.6 we provide an overview of the employed datasets and computational models, as well as the training and evaluation strategies applied in our study. Section 7.7 presents the experimental results based on objective metrics towards evaluating the effects of each encoding on the structure of the generated music. We also visualize the model parameters to gain deeper understanding of the representational capacity of each encoding. Section 7.8 summarizes the contributions of our study and provides a discussion on possible future research directions.

7.2 Research Scope and Contributions

The research endeavor for studying the effects of different music encodings on the latent feature representations memorized by recurrent computational models, emerged as a result of the limitations encountered in the development of the automatic jazz improvisation accompaniment system, as described in Chapter 6. Particularly the choice of a suitable musical representation was not obvious, while the employed encoding strategy obscured possible note repetitions. Furthermore, upon reviewing the existing literature, we discovered a notable absence of a comprehensive empirical study on the impact of different information encodings in music generation systems. While the rationale behind the choice of data encoding was occasionally mentioned in established generative systems, it was typically discussed in relation to specific problem domains or model architectures, often with limited detail. In many cases, there was not any information provided regarding the specific methods used for data encoding. This prompted us to question the possibility of establishing benchmark tests to compare different musical data encodings, creating a taxonomy of diverse encoding approaches, and possibly acknowledging that certain encodings are an inseparable part of a musical work, or alternatively be regarded as just another implementation detail, simply translating “pure” musical information

to representations ready for computational modeling. Either way, we recognized the value of investigating the effects and characteristics of different encodings. Either way, we argue that an investigation in their effects and characteristics would be of scientific value. Such an exploration could shed light on their role in music generation and provide insights into their broader implications.

Hence, with this study we aim to provide answers regarding the following questions. What can we deduce from the different ways to encode symbolic musical data? Is it possible to categorize them in a systematic way? What are the advantages and drawbacks for each one, especially regarding automatic music generation systems? By keeping some choices constant, such as the type of network, the dataset and the way music is being generated, can we gain some insight on the way certain computational architectures and types of musical data behave when trained with different encodings? How do they impact the form of a synthesized musical piece? In order to address these inquiries we:

- Develop a taxonomy of monophonic music encodings by considering common musical characteristics.
- Implement an automatic music generation system utilizing a LSTM RNN computational architecture.
- Train the system using a dataset of musical pieces encoded in various formats.
- Evaluate the generated output of the various models trained with different encodings and compare them to the original dataset.
- Investigate the ability of each model to learn high-level features by analyzing the network's parameters when presented with different encodings of the same musical piece.

7.3 Related Work

Although the issue of digitally encoding symbolic musical data has been a constant challenge in the design of all sorts of systems in MIR applications, there has been a limited focus on dedicated studies addressing this issue. Especially regarding the field of ML-based music generation, there have been three significant surveys conducted. In the study presented by

Fernandez *et al.* [225], although various systems are discussed, with the majority not utilizing neural networks, there is no specific focus on encodings. Similarly, Herremans *et al*. [226] attempt a taxonomy for music generation systems, exploring many of the same characteristics as our encoding taxonomy presented in Section 7.4. However, Briot *et al.* dedicate an entire chapter to music representation in their work presented in [191], providing a comprehensive explanation of important neural network models for music generation. It also includes a chapter covering the fundamentals of neural network models used in music generation, making it a valuable resource for beginners. For a more extensive and mathematically advanced reference on neural networks, we refer to the work presented by Aggarwal in [227].

Probably the first use of RNNs-based models for automatic music generation appears in [228], the design characteristics of which still appear in most sequential music generation models and whose tradition we essentially follow with this study. Some years later, Eck and Schmidhuber [197] proposed a recurrent system that employs LSTM gates, managing to synthesize long-term structured blues improvisations based on a standard blues chord progression. The availability of various RNN-based models, particularly with LSTM gates, has greatly expanded the repertoire of music generation techniques. Karpathy's "char-rnn" model and his blog post [229] played a significant role in popularizing character-based sequential autoregressive RNN models for text generation, which subsequently led to their adoption in music generation as well. Inspired by char-rnn, numerous music generation models have been developed, and one notable example is "folk-RNN" [230]. Originally trained on a corpus of pieces represented in the text-based ABC notation, folk-RNN demonstrated the potential of char-rnn-like approaches in generating folk music compositions.

While we utilized a LSTM-RNN model in our study to explore a well-established computational architecture, recent advancements in music generation systems have surpassed simple RNN architectures. Notably, the Transformer architecture introduced in the work by Vaswani et al. [176], has gained prominence. This sequential model incorporates attention mechanisms and has been employed by Google's Magenta Team in the "Music Transformer" system [231] with highly promising outcomes. Another noteworthy system developed by the same group is "MusicVAE" [232], which utilizes a hierarchical decoder within a VAE framework to adapt to sequential musical data.

7.3.1 Works exclusively about symbolic encodings

There are more than a few articles and books on symbolic music representations, most of which are fairly old and deal with representations that have since been abandoned [233]. The definitive handbook is considered the work of Byrd *et al.* [234], where they catalogue and analyze practically every major (and a few minor) symbolic representations for music at the time, including MIDI and its extensions (MIDI-like languages like Csound[3]), as well as score representations. Another important study is presented by Wiggins *et al.* in [235], where an evaluation of representations is attempted on the basis of two separate axes:

- *Expressive completeness*, denoting the range of raw musical data that can be represented.
- *Functional generality*, measuring the capacity of representing a range of high-level structures.

This is a useful approach, which somewhat echoes our historical distinction between essential digital music protocols like MIDI and human interpretable scores. The work presented by Honing in [236] provides a valuable exploration of the representation of time and structures in music, highlighting potential pitfalls and challenges that can arise in this domain. Additionally, the topic of symbolic music representation is also addressed in the brief but insightful work by Dannenberg in [237].

Upon reviewing the relevant literature, it becomes apparent that much of the research on symbolic representations predates the past two decades, with many of the cited articles being at least 20 years old. Moreover, the systems mentioned in these works have largely been abandoned by the majority of music-related research communities, without being replaced by new designs. This shift can be attributed, in part, to the predominant use of audio in digital music and sound production over the past 20 years, leaving symbolic data sources limited to recordings of MIDI performances. There is a prevailing notion that arguing for a "good choice of representation" is somewhat outdated. Although handcrafted feature extraction has been prevalent in the field of MIR until recently, it has been surpassed by automatic feature learning methods when sufficient labeled data is available [238]. Symbolic encodings often encompass a set of features, particularly when explicitly

[3] `https://csound.com/`

specified in the encoding for musical-related attributes. Therefore, selecting the appropriate encoding for a given task is akin to manual feature extraction. While our focus primarily centers on encodings intended for computational use, it is worth noting that significant efforts are being made, primarily within the field of musicology, to develop symbolic encodings for score visualization purposes. In the work by Napoles *et al.* [239], it is argued that the standard approach, "what you see is what you get", employed by most electronic notation frameworks, has inherent limitations. Identical-looking scores can possess different semantic content, which the encoding should take into consideration in order to avoid any possible discrepancies in the future.

7.3.2 Evaluating music generation systems

Evaluating generative systems poses inherent challenges [240], and the absence of semantic information in musical data, along with the lack of widely accepted performance metrics for computer generated music, further complicates finding a proper validation approach. The computational creativity research community has extensively examined deep learning music generation systems [241, 242]. The conventional approach to assess the output of a generative model involves either human expert evaluation and listening surveys (i.e objective evaluations), as well as the utilization of objective measures to evaluate different qualities of the computational models. While listening tests can be informative, many suffer from limitations such as incomplete disclosure of testing parameters, insufficient sample sizes to yield significant results [243], or improper utilization of the Turing Test approach [244]. Often, the analysis of a selection of the model's output by a music expert can provide deeper insights into its quality and characteristics [245].

Alternatively, rather than directly evaluating the output of the computational model, an evaluation based on the architectural parameters can be attempted. For instance, in the case of the folk-RNN system [246][4] and other RNN-based architectures like [247], an evaluation based on the parameters of intermediate layer weight activations and loss gradients has been explored. It is important to consider the size of the model, as larger models such as folk-RNN (containing over 5 million parameters) require ap-

[4] `https://github.com/IraKorshunova/folk-rnn`

propriate statistics and visualizations to extract interpretable information from the model's parameters.

7.4 Characteristics of Music Encodings

In this section, we present an overview of various potential characteristics that an encoding can possess. A sufficiently complex encoding has the capacity to represent and express these characteristics in different manners. Some of these traits can be explicitly stated or inferred by human readers or processing systems, while others may remain immutable throughout the entire piece, or act as a "state" until altered. Additionally, certain characteristics function as signals, requiring them to be set at every time instance. Although our experiments solely utilize monophonic encodings (i.e. allowing only one voice to be active at any given time), the following characteristics are applicable to any method capable of encoding polyphony, which encompasses a broader range than the systems we examine in our study. The characteristics discussed as follows are drawn from existing discussions in the literature on symbolic representation, primarily from [191]. These characteristics have been either expanded upon or restricted to align with the scope of our research, while also including some proposals of our own. While we cannot claim to have provided an exhaustive list, it is valuable to identify at least these characteristics when encountering or creating a new encoding. This approach allows the encoding to be placed within a larger "encoding graph", where links represent the steps necessary to modify two encodings until they become essentially the same.

Metric Information In order for a score to be easily interpreted by humans while conveying rhythmic information, it is typically necessary to incorporate a metric structure. This structure can be either static or dynamic, often indicated by a Time Signature with barlines, used to demarcate measures. The specification of these elements can be either explicit or implicitly deduced.

Key Information The key signature in staff music notation typically implies the key, either through the presence of accidentals within the score or by explicit notation. This is particularly common in traditional and popular music songs, which usually maintain a static key. Moreover, the scale mode can be explicitly specified as well.

Velocity/Volume/Dynamics Information Velocity is commonly interpreted as the "force" exerted by a performer on their musical instrument, resulting in a louder sound (thus higher in volume). Due to the absence of dedicated control knobs on physical musical instruments, velocity and volume are often used interchangeably. However, in a MIDI file, volume parameters directly influences the instrumental output volume linearly, while velocity can yield various effects based on the design of the MIDI instrument. Nonetheless, velocity primarily controls what we refer to as performance dynamics [248]. In dynamics-sensitive systems and encodings, both velocity and volume typically need explicit notation.

Time and Tempo In encodings that involve quantized note lengths or incorporate metric information, a fixed timestep is generally established based on a default length. Tempo is often expressed in Beats per Minute (BPM), where a beat typically corresponds to a quarter note. To ensure that no timing information is lost, the fixed timestep length should be the lowest common denominator among the durations of all the notes. In cases where none of the aforementioned information is provided, the progression of time is typically measured in absolute units. Altering the tempo in such situations is equivalent to multiplying or dividing the current time value at each point by a number within the range of $(0, \infty)$. Alternatively, a third option between uniform quantization and a real-valued timescale, is the utilization of variable timestep segmentation. This approach involves employing separate timesteps for each played note, as demonstrated in [249].

Onset, envelopes and Note Events Virtually, every music encoding encompasses an equivalent representation of a note onset, denoting the precise time at which a note event occurs. In most human-readable notations, there is typically no explicit information regarding how the amplitude or other characteristics of a note evolve over time. Therefore, note events are considered "distinct" and "uniform", placing the responsibility of capturing and expressing these evolving characteristics upon the performer or virtual instrument. Regarding pitch, particularly in the case of pitched instruments, it is commonly specified either as a discrete element within a specific tuning system or, more generally, as a fundamental frequency expressed in Hz. Alternatively,

pitch can also be encoded as an interval relative to a reference pitch, often employed when the data consists of pieces in the same key.

7.5 Proposed Music Encodings

In the scope of our study, we have identified and categorized music encodings into two primary families: (i) timestep-based and (ii) event-based encodings. Timestep-based encodings employ an implicit representation of time, relying on a fixed timestep length reminiscent of the piano roll representation. On the other hand, event-based encodings provide a granular representation of music by encoding individual events, allowing for more detailed and flexible representation of musical information. A detailed description is presented as follows.

7.5.1 Timestep-based encodings

Fixed timestep encodings have been extensively employed in the early applications of artificial neural networks for music generation [197, 228]. The prevailing belief was that their straightforward temporal structure facilitated more effective processing of rhythm by the computational models [250]. In all fixed-timestep encodings, it is essential to distinguish held notes from repeated ones. One approach, exemplified by *tstep1*, employs a "hold event" token, as observed in the "DeepBach" system [198]. Conversely, *tstep2* adopts a "note-off" token, as seen in [197]. Another variant, utilized in [228], incorporates an additional token to indicate the start of notes.

To ensure consistency, we set the timestep to the minimum note length we wish to represent, relative to a quarter note length. The term "resolution" is used to denote the value used for dividing one quarter note duration to determine the relative length of a timestep. For instance, a resolution of 4 corresponds to a timestep length of $1/4$ of a quarter note, equivalent to a $16th$ note. In our subsequent experiments, we select a resolution of 8 for all encodings, resulting in a minimum duration of a $32nd$ note. Alternatively, when no metric information is available, absolute time values of note subdivisions can be calculated based on the given tempo in BPM, and then quantize the piece accordingly.

tstep1

In *tstep1*, each piece is represented as an array of integers in the range $[0, 129]$, where:

- $\mathbf{0 - 127}$: Denote a note-on event, where each value corresponds to a specific pitch.
- $\mathbf{128}$: Represents a rest, indicating a period of silence.
- $\mathbf{129}$: Signifies "continue playing the last-seen note", instructing the model to sustain the previously processed pitch.

When multiple pieces are concatenated into a larger array, it is possible to insert piece start and stop symbols between them. These symbols can be represented, for example, by introducing additional tokens such as 130 and 131. For instance, let's consider the transcription of a musical sequence consisting of a quarter note C3 followed by two eighth C3 notes. In the corresponding sequence vector, it would be represented as $[60, 129, 129, 129, 60, 129, 60, 129]$. It is important to note that during the conversion process, all pieces are quantized to the smallest representable note length, which is equal to the chosen timestep length.

tstep2

In *tstep2*, each piece is represented as an array of integers within the range $[0 - 129]$, where:

- $\mathbf{0 - 127}$: Represent note-on events, with each value indicating a specific pitch.
- $\mathbf{128}$: Denotes a rest.
- $\mathbf{129}$: Indicates a note-off event, signifying the end of a sustained note.

In contrast to the *tstep1*, the *tstep2* encoding handles note durations differently. Held notes are represented by sequences of note-on events. A note may stop playing either when another note or rest appears, or when a note-off event is encountered. The note-off event continues the note for the current timestep and stops it thereafter. Note-off events are only used when the same note is played consecutively. This distinction becomes important to avoid ambiguity in representing different note durations. For example, to encode two C3 $8th$ notes, the vector sequence would be $[60, 129, 60, 129]$,

instead of $[60, 60, 60, 60]$, which would represent a C3 quarter note (assuming a $16th$ note resolution).

The inclusion of note-off events in the timestep-based notation was discussed in [197] to address the challenge of encoding notes with the minimum timestep length. However, herein we utilize the note-off event only when the same note is played successively. To fully resolve this issue, a separate note-off event would be required for cases where a note is held for only one timestep and then repeated. For instance, to encode three C3 $16th$ notes, the sequence would be $[60, 130, 130]$, where 130 represents the note-off event. During our experiments, the quantization challenges described above were not extensively addressed, as the chosen resolution proved to be sufficient in preserving note onset and duration information.

The main reason for employing a separate timestep-based encoding, aside from addressing quantization challenges, was to examine the event distribution in both datasets. In Figure 7.2, which illustrates the token distribution in the dataset used for our initial experiments, we observe that the distribution in *tstep2* is less skewed compared to *tstep1*, having significantly higher number of note-on events. Consequently, the distribution of note-on events in *tstep2* no longer aligns with the distribution of their original onsets.

7.5.2 Event-based encodings

In event-based encodings, musical data is represented as a series of discrete events or individual commands. Each event typically contains information about a specific occurrence, such as the onset, pitch, duration, velocity, or any other relevant attribute of a musical note or event. These events are arranged in a sequential order to capture the temporal structure of the musical piece. Unlike timestep-based encodings, which rely on a fixed timestep length and implicitly represent time, event-based encodings focus on the individual events themselves. This approach allows for greater flexibility in representing musical dynamics and note envelopes, as events can have varying durations and temporal relationships. Event-based encodings are particularly useful for capturing expressive performances, complex rhythms, or polyphonic music, where multiple events may occur simultaneously or in close succession.

The most widely-used encoding based on events is the MIDI format that

represents musical events as discrete messages, including note-on and note-off events, control changes, pitch bends, and more. Another famous example is the ABC notation [218] that encodes musical events using letters, numbers, and symbols to indicate notes, rhythms, and other musical information. Most event-based encodings are designed to allow for precise representation and manipulation of musical events, facilitating various music-related tasks such as analysis, composition, transcription, and generation. However, the aforementioned encoding systems still require to apply some kind of preprocessing in order to be properly employed by the underlying computational model. In this regard, we propose an event-based encoding approach that tries to bridge this gap, referred herein as *event1*, and its methodology is described as follows.

event1

The *event1* encoding draws loose inspiration from the MIDI protocol and incorporates tokens for note-on and note-off events. It explicitly represents the progression of time using special "move forward in time" events. Specifically:

- $\mathbf{0 - 127}$: Correspond to note-on events.
- $\mathbf{128 - 255}$: Denote note-off events.
- $\mathbf{256 - 356}$: Represent time-shift events.

Rests are implicitly represented in this encoding but can be made explicit by including an additional rest token with its corresponding note-off token. The time-shift events are utilized to advance time by a specific increment, which can be measured in seconds (or any subdivisions), or aligned with a predefined grid (such as quarters, $16th$ etc.). Moreover, it is common practice to use various time lengths. For example, the range $256 - 266$ can represent increments of 1 to $10\,\mathrm{ms}$, while $267 - 277$ can represent increments of 10 to $100\,\mathrm{ms}$, and so on. In our implementation, the event with a value of 256 represents the shortest time interval, and subsequent events represent multiples of that interval. This implementation simplifies comparisons with timestep-based encodings, as the length of the 256 event aligns with the chosen timestep length, enabling consistent use of the resolution parameter across all considered encodings.

A typical pattern, particularly in well-quantized pieces, involves a sequence of events, for instance: note-on, a period of waiting, note-off, another period of waiting, followed by another note-on, and so on. Consequently, each note in the original piece requires three or four distinct events to encode. This means that there is a strict limit on the number of events needed to represent a piece, and this limit does not increase with resolution. For most musical pieces, this results in significantly fewer events compared to timestep-based encodings, except for densely packed, well-quantized pieces with a high resolution, where a timestep-based encoding theoretically requires one or two events per note. This distinction becomes especially important when aiming to capture real-time performances, where the time step can be as low as $1/44100\,\mathrm{s}$. A similar encoding to *event1*, with the inclusion of velocity events, was employed in Magenta's "Performance RNN" system [251].

7.6 Methodology

In this section, we provide an overview of the datasets, computational models, training and evaluation strategies employed in our study. We conduct two major experiments, where we train a LSTM-based recurrent architectures on two datasets using different encoding approaches, in order to assess the resulting representations through subjective and objective measures. In our initial experiments we focus on a small dataset of European folk songs encoded in *tstep1*, *tstep2*, and *event1*. However, in the second experiment we employ a larger dataset of Irish folk tunes, restricting the computational model to a single music style, encoded in *tstep1*, *event1*, and a tokenized version of the ABC notation system. For further details, including the Python source code developed for the experiments, as well as generated audio examples of the trained models, please refer to the project's GitHub repository[5].

7.6.1 Datasets and preprocessing

In the first part of our experiments, we utilize a dataset consisting of 7264 original melody transcriptions in the ***kern* format. These transcriptions were sourced from the online KernScores library [252], specifically from two

[5] `https://github.com/manosplitsis/MusicRep`

Listing 7.1: A German folk piece in the Essen corpus in **kern format.

```
!!!OTL: ABSCHIED MUSS ICH NEHMEN HIER
!!!ARE: Europa, Mitteleuropa, Deutschland, Niederrhein;
   Westfalen;     Bergisches Land
!!!SCT: E0776
!!!YEM: Copyright 1995, estate of Helmut Schaffrath.
**kern
*ICvox
*Ivox
*M4/4
*k[b-e-a-]
*E-:
=1
{4e-
8e-
8g
4b-
4b-
=2
4.g
8f
4e-
4r}
```

collections: (i) approximately 6000 songs from the Essen folksong collection, and (ii) 1000 songs collected by Daimen Sagrillo[6]. The **kern notation style is a part of the Humdrum toolkit, which provides a text-based representation of music along with software for musicological analysis. An example of the first two measures of a typical **kern file from the dataset is presented in Listing 7.1. Each voice in the transcription is represented by a separate column containing note, duration, key, and metric information. Although the dataset is relatively small in terms of the number of pieces and average duration, the **kern representation offers rich structural and metadata information, facilitating the extraction of relevant features when necessary. Furthermore, the Humdrum music format can be easily parsed using the music21 MIT Python library[7] [216], which we use for all pre- and post-processing of the musical data.

Therefore, the **kern files were processed and converted into music21 Streams. We removed irrelevant information, such as titles and comments,

[6] https://kern.humdrum.org/

[7] http://web.mit.edu/music21/

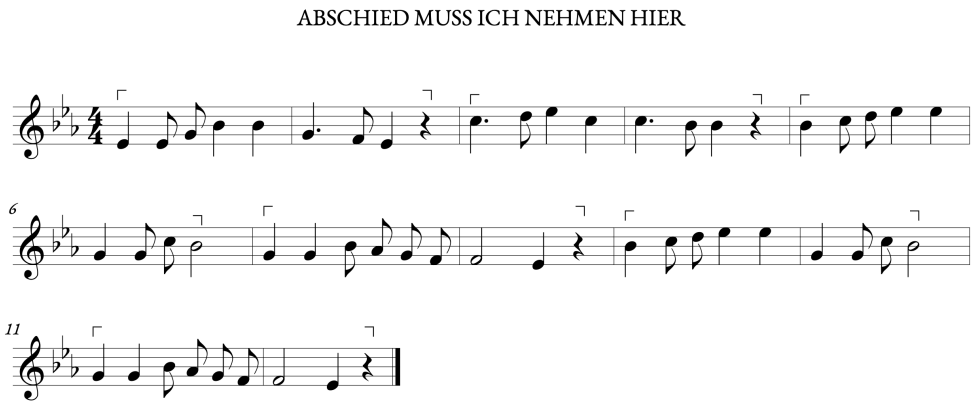


Figure 7.1: A German folk piece in the Essen corpus, in score format.

and we arranged the transcriptions into a flat structure. Subsequently, we extracted the *tstep1*, *tstep2*, and *event1* representations for each song as vectors of integers. A typical original transcription from the Essen corpus is illustrated in Figure 7.1. By plotting the distributions of the different tokens we can observe the effect of the considered encoding methods on the resulting amount of data. Figure 7.2 displays the number of occurrences in a logarithmic scale, for every token in the three encodings (from top to bottom: *event1*, *tstep1*, *tstep2*). The token numbers correspond to the descriptions provided in Section 7.5. In all plots, note-on tokens are represented in blue. As it regards *event1*, we can observe that the distribution of note-off tokens, which are highlighted in red, mirrors that of note-on tokens, as expected. On the other hand, the distribution of tokens in *tstep2* is less skewed compared to *tstep1*, having significantly higher number of note-on events. Consequently, the distribution of note-on events in *tstep2* no longer aligns with the distribution of their original onsets.

For the second part of our study, we opted to work with a larger dataset while simultaneously limiting the variation in musical styles. To achieve this, we utilized a collection of 45849 transcriptions of Irish folk tunes in ABC format, which were originally used to train folk-RNN v3 model[8]. These tunes were sourced from the weekly compilations provided by "The Session" community [9] and preprocessed in a later stage by the folk-RNN team. In the preprocessing stage the ABC dataset underwent several modifica-

[8] `https://github.com/IraKorshunova/folk-rnn/tree/master/data`
[9] `https://thesession.org/`

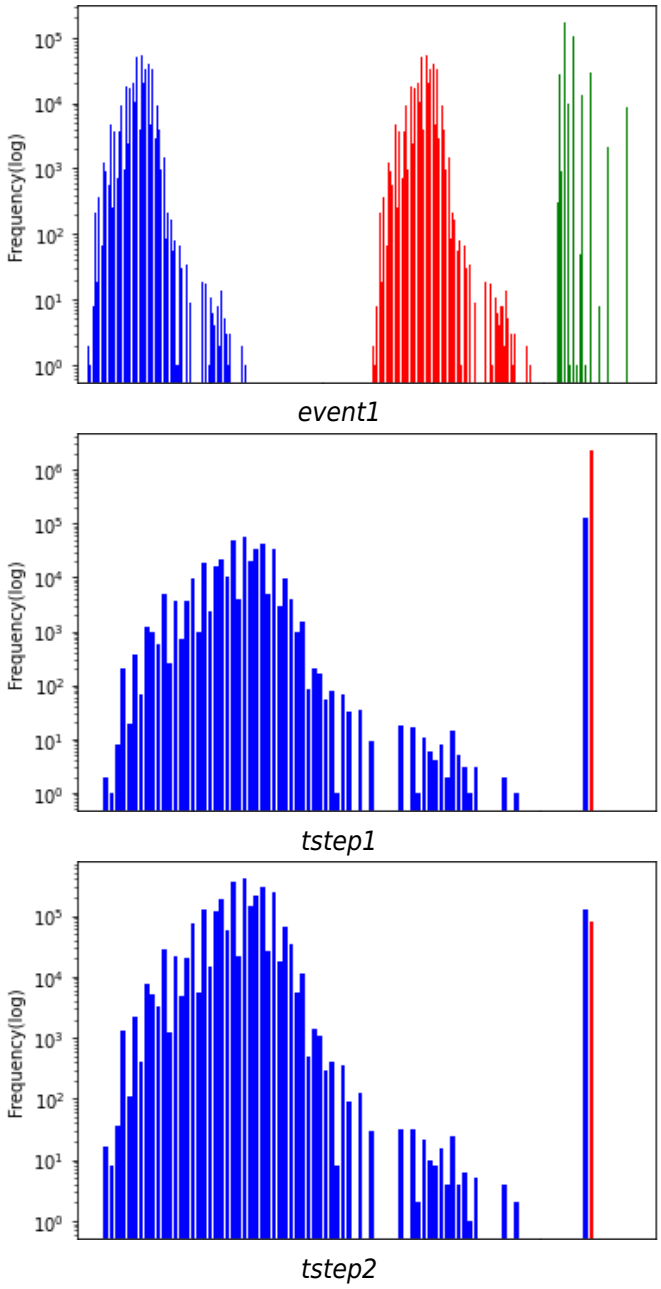


Figure 7.2: Distribution of ground truth labels for each representation (top to bottom: *event1* - *tstep1* - *tstep2*). The y axis denotes number of occurences (in log scale).

Listing 7.2: The Swallow's Nest

```
T: Swallow's Nest, The
M: 4/4
L: 1/8
K: Gmaj
|~G3A GEDE|G2BG AedB|A~E3 ~A2Bd|e2fa gedB|
~G3A GEDE|G2Bd efge|d~G3 EGDG|1FGAF GcBA:|2FGAF G2 z2|:
e~g3 edBA|GEDE G2Bd|eaag aged|~B3d e3g|
d~g3 agba|aged e2ge|d~G3 EGDG|FGAF G2 z2:|
```

tions, removing all fields, except the meter (M) and key (K) information. Additionally, the tunes were transposed to a key without accidentals, corresponding to one of the modes of the C major scale. Finally, the files were tokenized by introducing spaces between individual tokens.

Although the "pure" text-based ABC format falls outside the direct scope of our study, it is worth mentioning that the tokenized version proposed by folk-RNN introduces a transformation that groups symbols together to form new "musically meaningful" symbols. It is important to note that this tokenization process is arbitrary, as different choices could lead to distinctive encodings. For instance, the duration of a note could be included in the note tokens, or the note octave could be represented as a separate token. The tokens represent various elements, such as individual notes (e.g. 'c', 'Ĝ', '=f,'), duration adjustments (e.g. '/2', '4'), complete header expressions (e.g. 'M:6/8', 'K:maj' indicating C major, 'K:dor' indicating D dorian), and bar lines ('|', '|:'). A typical transcription of an unprocessed tune in ABC format is presented in Listing 7.2. The same tune after undergoing tokenization, with added lines after the header for readability, is displayed in Listing 7.3.

We extracted the *tstep1*, *tstep2*, and *event1* encodings from the MIDI files provided by the folk-RNN team. The MIDI files were generated by converting the ABC dataset using the abc2midi program[10]. The first bar of "The Swallow's Nest" encoded in *tstep1* is presented in Listing 7.4, while the encoding in *event1* is displayed in Listing 7.5. The resulting datasets exhibit various differences. The average sequence length required to represent one measure is 8.71 (STD 3.362) for the tokenized ABC dataset, 20 (STD 3.939) for the event-based encoding, and consistently 32 for the fixed-timestep encoding. This means that the musical context provided to the

[10] https://abcmidi.sourceforge.io/

Listing 7.3: The Swallow's Nest (ABC Tokenized)

```
<s>
| =C 3 =D =C =A, =G, =A, | =C 2 =E =C =D =A =G =E |
=D =A, 3 =D 2 =E =G | =A 2 =B =d =c =A =G =E |
=C 3 =D =C =A, =G, =A, | =C 2 =E =G =A =B =c =A |
=G =C 3 =A, =C =G, =C |1 =B, =C =D =B, =C =F =E =D :|
|2 =B, =C =D =B, =C 2 z 2 |: =A =c 3 =A =G =E =D |
=C =A, =G, =A, =C 2 =E =G | =A =d =d =c =d =c =A =G |
=E 3 =G =A 3 =c | =G =c 3 =d =c =e =d |
=d =c =A =G =A 2 =c =A | =G =C 3 =A, =C =G, =C |
=B, =C =D =B, =C 2 z 2 :|
</s>
```

Listing 7.4: The Swallow's Nest (*tstep1* - first bar)

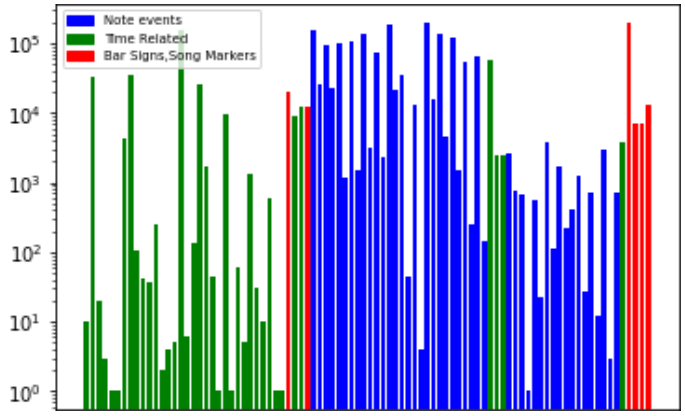
```
[60, 129, 129, 129, 129, 129, 129, 129, 129, 129, 129,
   129, 62, 129, 129, 129, 60, 129, 129, 129, 57, 129,
   129, 129, 55, 129, 129, 129, 57, 129, 129, 129]
```

Listing 7.5: The Swallow's Nest (*event1* - first bar)

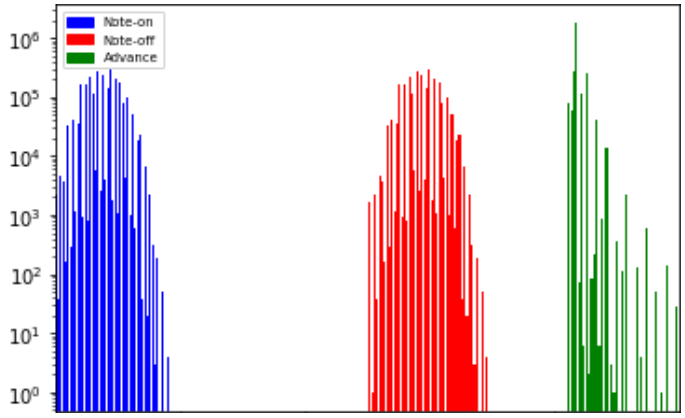
```
[60, 267, 188, 62, 259, 190, 60, 259, 188, 57, 259, 185,
   55, 259, 183, 57, 259, 185]
```

computentional model within a fixed-length sequence is three times larger when encoded in ABC compared to *tstep1*. This disparity increases further for fixed-timestep encodings as the time resolution becomes finer.

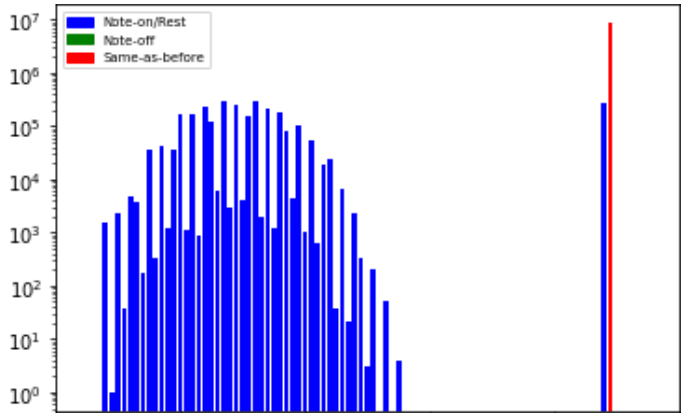
The resulting token distributions also present notable distinctions. In Figure 7.3, we plot in a logarithmic scale the number of appearances of each token for the three considered encoding approaches. Notably, there is a prominent prevalence of the "advance-by-1/8" token in *event1* (visible as a spike in the green area of the *event1* plot). This bias is a characteristic of the dataset, as further evidenced when visualizing the mean Note Length Transition Matrix in Figure 7.6. As observed in the token distributions of the dataset employed in our initial experiments, the distribution in *tstep1* is highly skewed, with two orders of magnitude more "hold" tokens (in red) compared to note-on or rest tokens (in blue).



ABC



event1



tstep1

Figure 7.3: Distribution of distinct tokens for each representation (top: ABC, left: *event1*, right: *tstep1*). The y axis denotes absolute number of occurences in the dataset, in a logarithmic scale.

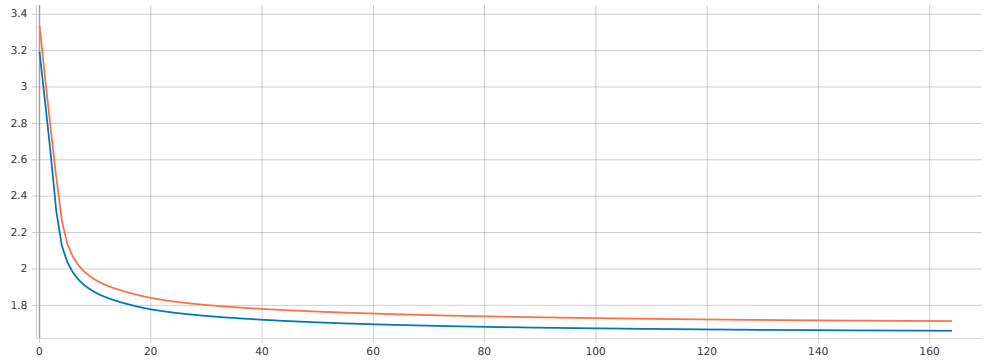
7.6.2 Model Architecture and Training

In both experiments, we deliberately opt for a small and shallow RNN architecture with a single LSTM layer and 32 hidden cells. We make this choice in order to establish a baseline architecture using a less complex model. Moreover, a shallow model with fewer parameters allows for easier inspection and visualization of the activations of the LSTM gates, that further enables us to gain insights into how the model "comprehends" each encoding.

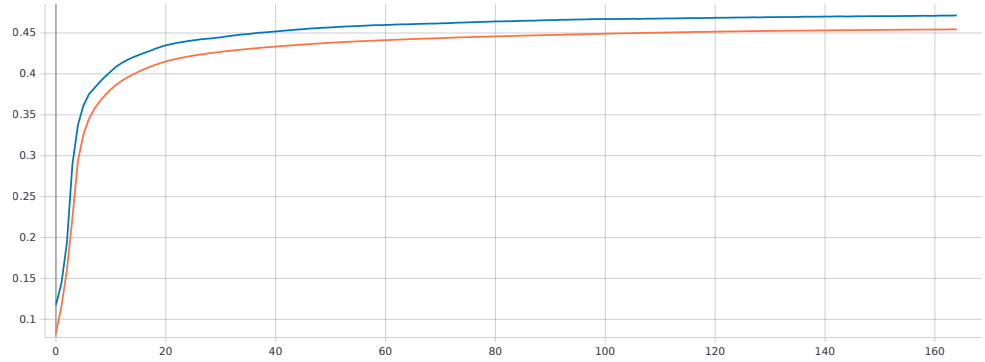
In the initial experiment, we tackle the automatic music generation task as a many-to-one sequence problem. In this setup, a sequence of tokens is fed into the LSTM layer, having as target (prediction) the token of the next timestep/event. The output of the LSTM layer is passed through a "Dense linear" layer with dimensionality equal to the size of the corresponding encoding dictionary. Next, we apply a softmax function to the output of the "Dense linear" layer, resulting in a probability distribution over the dictionary size. The final prediction is the token with the highest probability. During training, we apply a dropout rate of 0.2 on the LSTM gates (i.e. randomly setting 20% of the layer's activations to 0), aiming to enhance the model's generalization capacity [253].

The computational model is trained using partially overlapping sequences of 64 tokens. Each token is converted into a one-hot encoded vector with a dimension equivalent to the size of the dictionary accordingly. This conversion is applied by employing a sliding window technique across the length of each piece in the dataset, resulting in a total of $L - 64$ sequences for a piece with a length of L. The categorical cross-entropy loss is calculated between the true and predicted token, and the gradients are back-propagated after processing a batch of 256 sequences. The trainable parameters of the model are optimized using the Adam Adam optimization function [106], with an initial learning rate of 0.0001. Training continues until convergence is achieved, typically spanning more than 100 epochs.

In the second experiment, we address the music generation system by following a many-to-many sequence learning approach (i.e. seq2seq), where the target for a given input sequence is the entire sequence shifted by one token forward in time. This approach allows for the computation of the loss function across all predictions after processing a sequence sample. To ensure consistent number of tokens in a sequence sample, we use a sequence length of 100 tokens for all the considered encoding approaches,



(a) Loss curve.



(b) Accuracy curve.

Figure 7.4: Plots of the accuracy and loss function (red color corresponds to the training set, while blue color indicates the validation set) for the ABC dataset in the second experiment.

while smaller sequences are zero-padded to that length. During training, we monitor the validation loss and apply an early stopping when the loss value increases for more than 2 consecutive epochs. In Figure 7.4 we plot the loss value and accuracy rates on the training set (in red) and validation set (in blue), when fitting the model to the ABC dataset over 161 epochs.

7.6.3 Generation

In both experiments, the generation of new pieces follows an autoregressive process, where every prediction is generated based on past tokens, up to the current time instance. Especially in the first experiment, our model architecture only allows for a priming input sequence of fixed length, equal to the sequence length employed during training. Therefore, the resulting input seeds are relatively large compared to the average piece length in the dataset. Moreover, we apply a sampling function on the softmax probabili-

ties to select the next token, in contrast to the argmax approach employed during training. The sampled token is then appended to the seed sequence, while we discard the first token in the seed, ensuring a constant sequence length during generation. As a result, the model "disregards" the beginning of a piece once it generates the subsequent part, as its past context extends as far back as the sequence length.

In the second experiment, our model implementation allows for sequences of arbitrary length. Hence, we start with an initial priming sequence of exactly one bar, the length of which varies significantly for the three encoding methods examined herein. After each step, the predicted token is appended to the seed, effectively preserving the context of the entire generated piece at all times (at least in theory). However, it is important to note that LSTM models, despite being more optimized than regular RNNs, still are prone to vanishing gradient issues. As the gradient is back-propagated through time after each training batch, it diminishes over a finite number of timesteps.

During our experiments, we sample directly from the softmax distribution to obtain the index of the predicted tokens, in order to minimize the effect of autoregressive "Exposure Bias" as much as possible. In practical music generation scenarios, RNNs often employ sampling strategies to control the level of randomness introduced in the generation process. These strategies can involve dividing the "Dense linear" logits by a fixed value, known as the "temperature" factor, before applying the softmax function during inference, or sampling from only the top-K probability tokens.

7.7 Experimental Results and Evaluation

Designing an experiment that effectively measures the impact of music encoding, especially in the context of music generation, poses a nontrivial challenge. This is further compounded by the fact that evaluating music generation systems remains an ongoing research problem. To this end, our objective is to employ music-specific objective measures to evaluate the effect of each encoding on the statistical similarity between the generated output and the original dataset, similar to the approach presented by Yang and Lerch [243]. Furthermore, we aim to conduct an analysis on the model's parameters to gain a deeper understanding of its learning capacity for each encoding. This analysis draws inspiration from the methodologies employed

in [254] and [246]. Additionally, we evaluate empirically the quality of the generated music.

7.7.1 Initial Experiments

As mentioned in Section 7.6.1, in our initial experiment we utilized a dataset consisting of 7264 European folk songs originally encoded in the Humdrum format as **kern files. These files were used to generate three distinct tokenized datasets in the *event1*, *tstep1*, and *tstep2* encodings. Following the evaluation strategy proposed by [243] for music generation networks, we selected 200 random pieces from our validation set. Using the first segment of each piece as a seed melody, we generated a set of 200 melodies, each consisting of 8 bars (exactly 16 seconds in duration at a tempo of 120 BPM), for each encoding.

Objective Measures

Subsequently, we computed 12 music-specific statistical features for each set, including Pitch Count (PC), Note Count (NC), Pitch Class Histogram (PCH), Pitch Class Transition Matrix (PCTM), Pitch Range (PR), average Pitch Shift (PS), average Inter-onset Interval (IOI), Note Length Histogram (NLH), and Note Length Transition Matrix (NLTM). To facilitate comparison among different systems, we performed an exhaustive cross-validation by measuring the distance between each sample within a set (i.e. intra-set distance) and comparing it to all samples from another set (i.e. inter-set distance). These relative measures provided histograms for each feature, from which we computed continuous Probability Density Functions (PDFs) using the Kernel Density Estimation (KDE) approach. To evaluate the similarity between generated and original data, we computed the Kullback–Leibler Divergence (KLD) and Overlapping Area (OA) between the intra-set PDF of the generated data, as well as the inter-set PDF between the generated and original data. A low KLD indicates similarity in the shape of the compared distributions, while a high OA indicates a higher probability density overlap. These similarity metrics allowed us to assess how closely the generated outputs of each model resembled the original dataset across different measures. The results for the KLD and OA for the three sets are presented in Table 7.1. Our main focus was to answer the question: which of the three representations produced pieces that closely resembled the initial dataset, and why? While

Table 7.1: Intra-set similarity measures for the three encodings in the initial experiment.

		PC	PC/bar	NC	NC/bar	PCH	PCH/bar	PCTM	PR	PS	IOI	NLH	NLTM
tstep1	KLD	0.738	0.059	0.286	0.162	0.054	0.310	0.112	0.770	0.001	0.038	0.045	0.400
	OA	0.831	0.589	0.569	0.532	0.844	0.847	0.390	–	0.935	0.829	0.763	0.461
tstep2	KLD	0.272	0.403	0.028	0.110	1.094	1.491	0.326	0.020	0.008	0.031	1.525	0.058
	OA	0.819	0.943	0.944	0.847	0.339	0.357	0.638	0.864	0.881	0.847	0.058	0.236
event1	KLD	0.501	0.069	0.017	0.022	0.035	0.132	0.079	0.164	0.004	0.006	0.043	0.061
	OA	0.908	0.925	0.906	0.933	0.815	0.828	0.877	0.892	0.860	0.922	0.909	0.891

the absolute measures (as shown in Table 7.2) provided some insights, they only exhibited a weak correlation or failed to provide a comprehensive perspective.

Upon analyzing the relative measures, several trends become apparent. In Figure 7.5, we present the computed PDFs for the intra-set and inter-set distances of the PCH, comparing the original dataset with each generated set. The blue and green curves represent the variation within the dataset and the generated output, respectively, providing insights into the distribution characteristics of each measure. For instance, in Figure 7.5, the intra-set PDF of the dataset exhibits a bell-like shape, indicating that the variation of the pitch class histograms follows a Gaussian distribution with a mean value of 0.38 and standard deviation of 0.130, as shown in Table 7.2. It is worth noting that relying solely on the numerical values from Table 7.2 may be insufficient when the distribution deviates from a Gaussian shape, as observed in the case of *tstep2*. In this case, the distribution takes the form of a mixture of Gaussian distributions with distinct means.

Considering the plots for all features collectively, we observe that the intra-set distribution of *event1* consistently aligns closely with that of the original dataset. This finding is also reflected in the similarity measures presented in Table 7.1, where *event1* consistently exhibits high OA and low KLD values, indicating a high degree of similarity to the original dataset. On the other hand, the performance of the other two encodings (*tstep1* and *tstep2*) varies, depending on the specific measure being examined. Furthermore, the aforementioned findings align with subjective evaluations provided by 3 human experts. Upon listening to the generated samples, several observations become apparent. As it regards samples generated with the model that was trained with the *event1*-based encodings demonstrate su-

Table 7.2: Absolute and intra-set differences for all measures in the initial experiment.

	Dataset				**tstep1**				**tstep2**				**event1**			
	Abs.Value		Intra-Set		Abs.Value		Intra-Set		Abs.Value		Intra-Set		Abs.Value		Intra-Set	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
PC	7.755	1.586	1.760	1.399	6.960	1.435	1.591	1.306	7.830	2.464	2.780	2.116	8.645	1.860	2.084	1.615
PC/bar			4.186	1.501			4.070	1.538			6.223	1.918			4.639	1.446
NC	42.820	9.826	11.022	8.520	21.360	10.161	11.367	10.118	31.920	12.857	14.711	10.763	49.515	15.975	16.562	15.448
NC/bar			6.224	2.389			6.045	3.172			11.886	3.835			8.321	4.776
PCH			0.387	0.130			0.482	0.175			0.770	0.421			0.339	0.124
PCH/bar			2.052	0.314			2.548	0.397			2.699	0.948			1.904	0.308
PCTM			13.333	3.439			7.518	2.816			10.764	3.710			13.455	4.324
PR	13.205	4.332	3.753	4.861	11.120		3.500	2.834	12.895	5.298	5.506	5.108	13.895	4.127	4.290	3.980
PS	2.144	0.620	0.649	0.594	2.962	0.834	0.907	0.699	3.248	0.959	1.078	0.830	2.814	0.639	0.711	0.562
IOI	0.398	0.104	0.109	0.099	0.720	0.235	0.271	0.227	0.577	0.349	0.295	0.397	0.372	0.129	0.133	0.126
NLH			0.420	0.245			0.428	0.193			0.238	0.111			0.505	0.284
NLTM			19.279	9.150			11.014	6.903			10.868	4.734			25.218	15.592

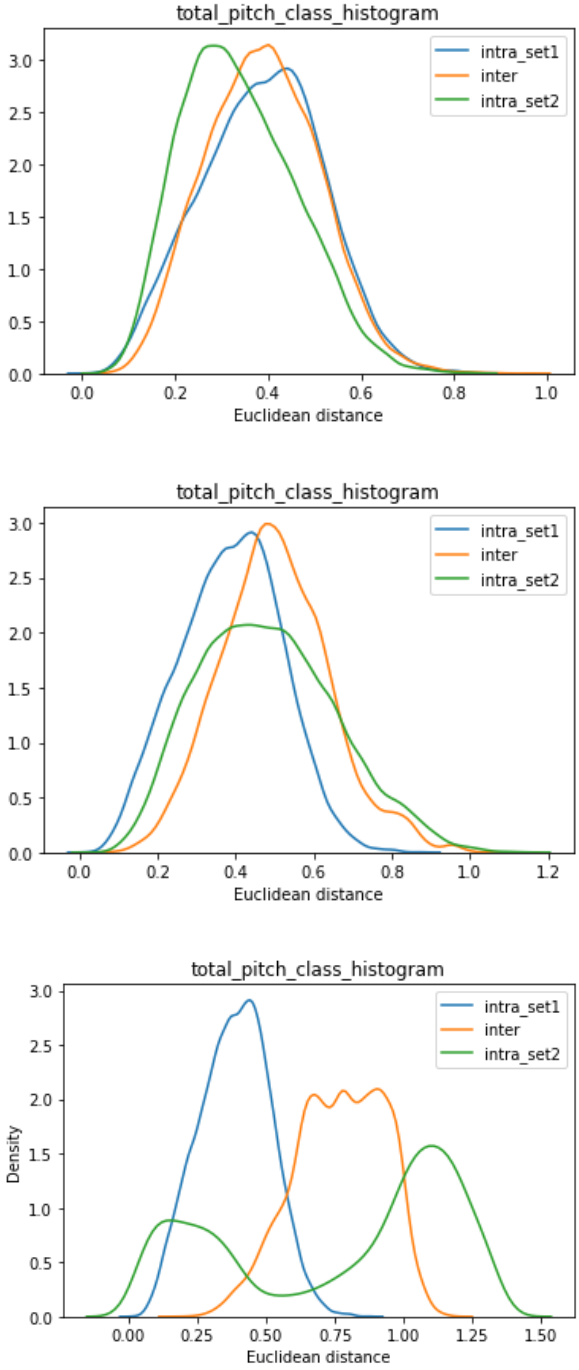


Figure 7.5: Pitch Class Histogram Intra- (blue and green curves) and inter-set (orange curve) difference distributions for the three generated sets in the initial experiment. Set1 is always the Dataset (in blue), while set2 is (from top to bottom) *event1*, *tstep1*,*tstep2* (in green).

perior quality, conforming to the stylistic rhythm and tonality of the dataset. The only discrepancy, which the statistical measures fail to capture, is the occasional lack of adherence to a metric structure, resulting in syncopations or, at worst, complete arrhythmia. This may be attributed to directly sampling the softmax distribution, which occasionally generates inappropriate time-skip events, particularly given the distribution of time skip events (see Figure 7.2). Many of these syncopations are imperceptible when listening without a metronome or rhythmic accompaniment. In contrast, *tstep1*, while generally maintaining a consistent metric structure, produces numerous off-key notes and irregularly spaced intervals (partially attributable also to the sampling approach). Examples from *tstep2* appear to be the weakest among the three encodings, both rhythmically and in terms of pitch. They feature prolonged notes (as expected from the data distribution) followed by rapid successions of short notes, likely due to sampling-based generation. Audio examples from the generated pieces can be found online in the repository of the project[11].

7.7.2 Second Experiment

In our second experiment, we incorporated the tokenized version of ABC notation provided by the folk-RNN project [230], along with the *event1* and *tstep1* encodings, according to the results from the initial experiment. The dataset we employed comprises 12058 Irish folk tunes in $4/4$ time signature, all transposed to the key of C major or its modes.

Objective Measures

Following the approach outlined in [243], we generated 200 music pieces using the models trained with each encoding. To serve as priming sequences for the generation process, we kept the first bar of 200 randomly selected tunes from the validation set, which vary in length to precisely represent one measure in $4/4$ time signature. We generated exactly 8 measures for each song, resulting in a total of 200 samples, consisting of 9 bars each, equivalent to a duration of 18 seconds at 120 BPM, for each encoding. For comparison purposes, we also considered the first 9 bars from the randomly selected 200 original pieces. Subsequently, all samples were converted to

[11] https://github.com/manosplitsis/MusicRep

MIDI format, allowing us to extract the same statistical measures, as outlined in Section 7.7.1.

The statistical measures for the three models, along with the corresponding objective metrics of the original tunes from the validation set, are presented in Table 7.3. However, interpreting the results directly from these measures can be challenging. To gain a deeper understanding of the computed metrics, we visualize some of the absolute measures in Figure 7.6, where we plot the mean NLTM and mean PCTM for the original validation set in the top row, while in the bottom row we present the NLTM and PCTM of a random sample from the original validation set. The mean NLTM indicates that, on average, the samples predominantly consist of $8th$ notes. As expected, the mean PCTM reveals that the dataset contains only notes from the C major scale, providing insights into prevalent melodic motions such as stepwise motion being more common, while interval skips of a $4th$ or $5th$ note are less frequent. Comparing the intra-set distances of the PCTM and NLTM measure between the original and the generated sequences, we observe that there is considerable variance in the original data, while the generated sets exhibit smaller variances. This observation aligns with the typical behavior of RNNs to converge towards the mean.

By analyzing the results presented in Table 7.4, we observe that all three models exhibit relatively low KLD values and high OA values, when comparing the PDF of inter-set distances in the generated sequences, to the intra-set distances between the original and generated sets for most of the computed objective measures. Based on the KLD and OA statistics alone, it is difficult to determine if one model is better at capturing the characteristics of the original dataset compared to the others. However, the ABC encoding appears to demonstrate slightly better performance across most measures. Nevertheless, further examination of the generated sequences by domain experts is necessary to provide a comprehensive assessment. Additionally, it is important to consider that the advantage reported by the ABC encoding may be attributed to a single re-encoding process, compared to the other encodings that involve multiple transformation steps from ABC to the target encoding and finally to MIDI, which could introduce artifacts or alter timing information that may not be present in the original transcriptions.

Table 7.3: Absolute and intra-set differences for all measures in the second experiment.

	Dataset				**tstep1**				**abc**				**event1**			
	Abs.Value		Intra-Set		Abs.Value		Intra-Set		Abs.Value		Intra-Set		Abs.Value		Intra-Set	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
PC	9.590	1.689	1.906	1.449	13.075	2.371	2.606	2.122	11.720	2.283	2.544	2.001	12.530	12.530	2.294	1.804
PC/bar			4.528	1.772			4.770	1.298			4.709	1.475			4.575	1.180
NC	61.520	10.112	10.50	9.761	62.810	6.621	6.918	6.345	62.785	7.782	8.277	7.295	58.345	58.345	6.168	5.632
NC/bar			4.953	2.973			5.367	2.155			5.315	2.350			4.919	1.894
PCH			0.301	0.103			0.191	0.059			0.233	0.079			0.191	0.060
PCH/bar			0.093	0.376			0.062	0.307			0.732	0.789			1.492	0.186
PCTM			16.933	3.674			11.225	1.623			13.182	2.551			10.855	1.684
PR	15.810	3.208	3.595	2.787	21.365	4.143	4.597	3.656	19.455	4.370	4.955	3.719	20.220	20.220	4.044	3.095
PS	2.967	0.615	0.669	0.560	2.853	0.341	0.379	0.301	2.908	0.453	0.501	0.402	2.827	2.827	0.404	0.323
IOI	0.302	0.061	0.056	0.066	0.290	0.039	0.036	0.041	0.285	0.038	0.037	0.039	0.294	0.294	0.038	0.040
NLH			0.376	0.336			0.287	0.282			0.328	0.323			0.299	0.310
NLTM			24.677	17.009			20.136	15.491			22.569	17.336			18.892	15.395

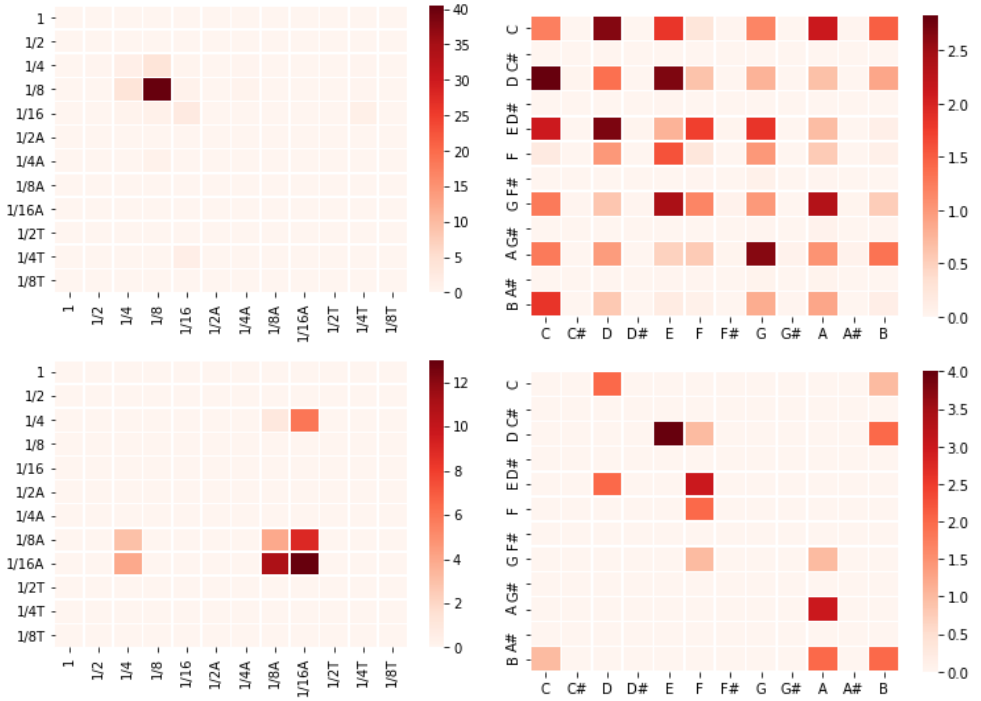


Figure 7.6: Top row: Mean NLTM (left) and PCTM (right) for the original dataset. Bottom row: NLTM (left) and PCTM (right) for a random tune in the dataset of employed in the second experiment.

Table 7.4: Intra-set similarity measures for the three encodings in the second experiment.

		PC	PC/bar	NC	NC/bar	PCH	PCH/bar	PCTM	PR	PS	IOI	NLH	NLTM
tstep	KLD	.1369	.2657	.0078	.0924	.0080	.1686	.0456	.0718	.0141	.0650	.0230	.0133
	OA	.7419	.9351	.8675	.8942	.6580	.4799	.4780	.7893	.8306	.8649	.8586	.9067
abc	KLD	.1746	.1046	.0088	.0334	.0234	.1947	.1120	.0075	.0043	.1280	.0151	.0129
	OA	.8775	.9549	.9062	.8981	.8251	.5556	.7153	.9267	.9048	.8709	.9074	.9314
event	KLD	.2605	.5395	.0395	.3559	.0552	.9893	.4550	.0296	.0085	.3785	.0397	.0194
	OA	.7358	.9366	.7323	.9041	.6768	.7352	.4495	.8132	.8464	.8615	.8529	.8536

Examining model parameters

In this experiment we also aimed to investigate the parameters of the computational models, to determine if they have captured valuable information regarding the musical structure of a piece, with respect to the underlying encoding. To achieve this, we utilized the trained models corresponding to each encoding and employed complete pieces from the respective validation set, as input to the models. The last token representing the "end of piece", is excluded from the input sequences. Then, we generated sequences of predictions, matching the length of the input sequences by utilizing the contextual information from preceding predictions (i.e. autoregressively). Throughout the generation process, we stored the probability distributions provided by the softmax function, along with the states of the LSTM layer that employs a $\tanh$ function, limiting the state values within the range of $[-1, 1]$.

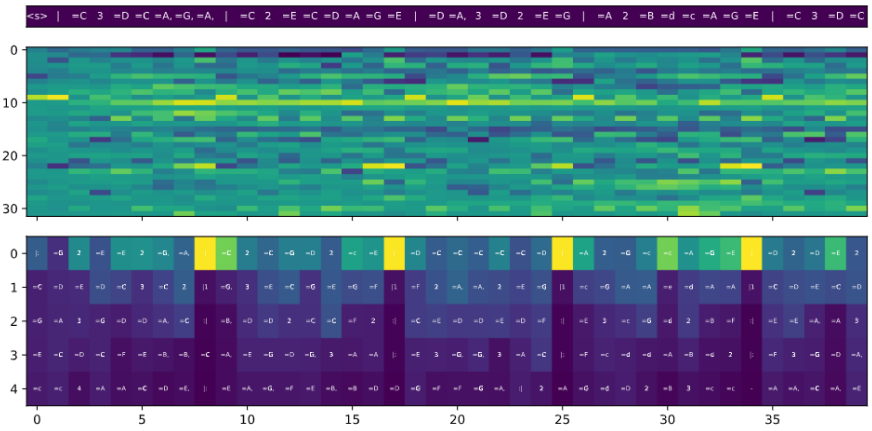
During listening-based examination of the generated music, a common characteristic observed in all models, despite their overall adherence to the correct tonal key, is their consistent ability to maintain rhythmic regularity, predominantly producing music perceived as following a $4/4$ time signature. Since the metric structure is implicit in *event1* and *tstep1* encodings, we conducted a visual analysis of the states for each of the 32 cells within the LSTM layer, focusing on the onset of a melody in all encodings. Our objective was to identify potential correlations between the activation patterns, and metric or other musical information. In Figure 7.7 we present these visualizations for each encoding approach considered in our study, where the top layer represents the input sequence, depicted as a dark blue bar, that corresponds to the beginning of a tokenized ABC tune. In the middle layer, the cell activations are visualized using a color heat map, where yellow indicates higher activation values (closer to $+1$), dark blue represents lower activation values (closer to -1), and light green represents intermediate activation levels. On the bottom part we show the top-5 probability predictions, as computed from the softmax function. We can draw several notable observations from these visualizations. In the case of ABC notation, it appears that cells 9 and 22 exhibit distinct activation patterns associated with barlines. Specifically, the $9th$ cell gets activated when a barline is processed by the model, while the $22nd$ cell shows anticipatory activation, preceding the occurrence of a barline (as indicated by the elevated probability of a

barline following these activations). Although the precise interpretation of other cell behaviors may not be immediately apparent, there is some indication of non-local memory retention, exemplified by the behavior of the $10th$ cell.

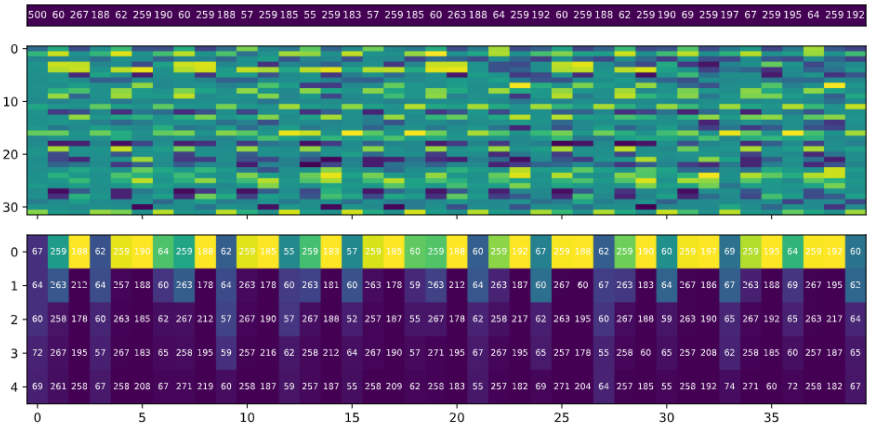
When examining the other encodings that lack explicit reference to barlines, it becomes challenging to discern whether the model possesses an inherent sense of measuring time. Instead, the focus shifts to identifying local patterns that indicate the presence of melodic phrases. Especially in the case of *event1*, we can observe that following a note-on event, the model consistently generates an "advance-time" event, followed by the appropriate note-off event for the currently playing note. Notably, the "advance-time" events with the highest probabilities correspond precisely to the duration of an $8th$ note, which is the default note length for all ABC pieces. A similar preference for $8th$ note durations can be observed in the softmax output of the *tstep1* model. This remark also aligns with the results from the mean NLTM presented in Figure 7.6, indicating that the model has learned to consistently output note lengths that are representative of the dataset's average. For both *event1* and *tstep1* encodings, note-on tokens exhibit fewer activations, with only a single cell consistently activating upon processing this token. Moreover, in all three models, we can observe that in the case of generating a note-on token, there is no discernible bias towards a particular note prediction. Consequently, the melodic output demonstrates significant variability, as a well-designed sampling strategy would mitigate repetitive patterns.

7.8 Conclusions

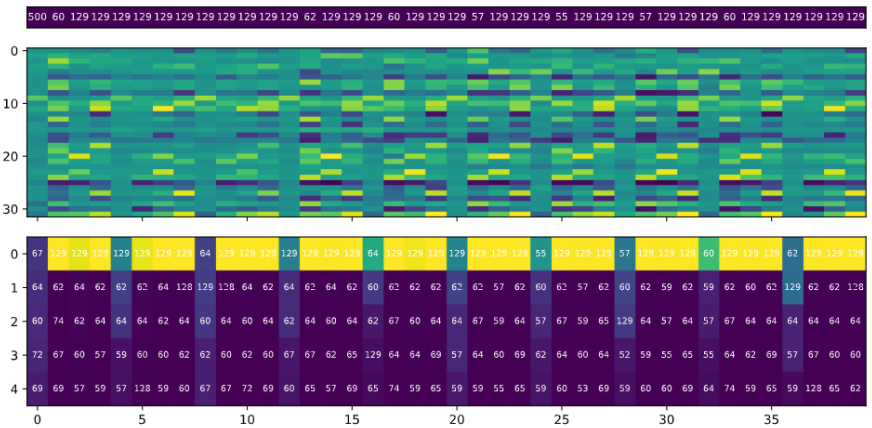
To the best of our knowledge, this study represents the first empirical evaluation of the impact of symbolic musical data encodings, on automatic music generation systems based on recurrent computational architectures. Our approach focused on investigating various encodings within a well-established baseline model, employing a shallow LSTM model for autoregressively generating monophonic melodies. Specifically, we examined a selection of widely-used symbolic encodings. Our experimental results conclusively demonstrated that the choice of data encoding employed in training a music-generating model, significantly influences the resulting musical



(a) The piece in tokenized ABC.



(b) The *event1* piece.



(c) The piece in *tstep1*.

Figure 7.7: Activations and predictions for the same song. For each of the three, the top row shows the input sequence, middle part shows the LSTM activations for each neuron (Dark blue is -1, Yellow is +1), the bottom part shows the top-5 probability predictions.

structure, including aspects such as musical phrasing and metric organization.

However, our study is subject to several inherent limitations. Generalizing results from such a small-scale study is challenging. Even within the confines of our research scope, there exists a multitude of potential encodings, many of which are not readily translatable across different formats and possess distinct characteristics that influence each model type in unique ways. It is worth noting that implementation issues and the availability of reliable parsing and conversion software tools, further compound these challenges. Additionally, compared to audio representations, there is a scarcity of symbolic musical data that encompasses sufficient metadata and exhibits the necessary diversity to facilitate large-scale experiments. Most readily available symbolic music data, primarily comprises human-made transcriptions of scores, predominantly representing Western common practice music, or a variety of curated or uncurated MIDI files.

Nonetheless, this should not discourage further research in this area. There are still valuable challenges to be explored. One possible direction is to conduct further experiments on popular encoding methods, which surprisingly are not as numerous as one might expect, using CNNs, transformer-based models or other state-of-the-art computational architectures. This line of research has the potential to shed light on how different systems process and retain information for creative tasks. In the long run, the objective could be to develop innovative approaches to efficiently encode musical information into abstract representations, potentially eliminating the need for manual transcriptions, while enabling translation into human-readable formats or plain MIDI events. Additionally, the convergence of musical notation and performance opens up possibilities for designing AI-enabled musical instruments capable of processing symbolic notations and interpreting them without explicit information on absolute timing or velocity, thereby introducing their own variations and styles during the interpretation process.

Part IV

Conclusions

CHAPTER 8

Conclusions and Discussion

8.1 Overall Conclusions

In this dissertation we have explored various aspects of computational music generation and interaction, contributing to our understanding of the field and paving the way for future advancements. The research endeavors unfolded with a focus on different tasks, encompassing musical gesture recognition, virtual music instrument interaction, audio-driven dance motion synthesis, jazz improvisation accompaniment generation, and symbolic music encodings. Throughout the various experimental campaigns described herein, the performance of recurrent computational architectures (i.e. LSTM RNNs), and convolutional models were examined, providing valuable insights into their respective capabilities and implications.

Our research began with the task of musical gesture recognition, where different computational models were evaluated. The initial architecture, utilizing a single LSTM layer, served as the foundation for a real-time gesture recognition system. However, subsequent experimentation using CNNs for computing feature embeddings, such as the CNN-LSTM approach and the deep CNN (dCNN) architecture described in Chapter 3, proved to be more effective. Both models outperformed the shallow LSTM-based model, demonstrating improved recognition accuracy and reduced computation time. These findings suggest that convolutional models, with their ability to capture spatial dependencies and exploit pattern recognition capabilities, offer advantages in the domain of musical gesture recognition.

Building upon the recognition models, in Chapter 4 the thesis delves into the development of a web-based system for real-time interaction with virtual musical instruments in a 3D environment. By leveraging the Leap Motion sensor and state-of-the-art web technologies, the system provided realistic visual and audio feedback, enhancing accessibility and cross-platform deployment capabilities. User evaluations confirmed the positive feedback regarding usability, audio realism, and design. Overall, the integration of convolutional and recurrent architectures showcased the potential for deploying various computational model types into interactive music systems, facilitating a seamless user experience.

Subsequently, our research in Chapter 5 tackled the task of audio-driven dance motion synthesis, where the use of deep CNN architectures proved to be highly effective. The proposed models, incorporating the DCHC layer along with a conditional decoder, outperformed the baseline LSTM models in terms of generating diverse and realistic skeletal motion sequences. These findings demonstrated the advantages of convolutional models in capturing complex temporal and spatial correlations within dance motion data. Furthermore, the DCHC layer allowed for a wider temporal receptive field, enabling the generation of longer sequences, while the conditional decoder enhanced the creativity of the generated sequences through stochastic control. We also considered two different training schemes, including classical teacher-forcing along with a curriculum-learning approach, demonstrating the effectiveness of self-supervised models to handle prediction error accumulation during the autoregressive synthesis process.

As it regards the scenario of jazz improvisation, in Chapter 6 our study focused on simulating the interplay between a human soloist and an artificial accompanist, showcasing the challenges and prospects of modeling music improvisational interactions using implicit ML approaches. The proposed system demonstrated harmonic compliance with chart chords information and contextual adaptability. However, limitations related to data availability and real-time design considerations were acknowledged, indicating the need for further research and dataset enrichment. The successful integration of RNN-based models highlighted the potential of combining different architectures to model complex musical interactions.

In this regard, we subsequently examined various symbolic music information encodings in the task of automatic music generation, emphasizing their impact on the resulting musical structure, as described in Chap-

ter 7. The choice of encoding method significantly influenced aspects such as melodic phrasing and metric organization. These findings highlighted the importance of careful consideration when designing music generation models. The findings highlighted the need for careful consideration of encoding methods in designing computational music generation models.

Overall, the performance of RNN and CNN models varied across the different tasks and architectures explored in our research. While LSTM-based models presented promising results in capturing temporal dependencies and modeling sequential data, the convolutional models, such as the CNN-LSTM, dCNN and DCHC architectures, exhibited superior performance in tasks such as gesture recognition and dance motion synthesis. Moreover, the findings from Chapter 7 highlight the significance of encoding methods, which suggest the potential benefits of convolutional models in shaping music generation outcomes. It is important to note that the performance of these models can be influenced by various factors such as dataset characteristics, model architecture, hyperparameter settings, and the specific requirements of the task at hand. To this end, visualization approaches were utilized in several chapters to aid in comprehending the outcomes of the various models. They provide intuitive representations of complex data and facilitate the analysis and performance evaluation of the developed systems. By visualizing various aspects of the model parameters and output, we gained valuable insights into their processing behavior, identified patterns and trends, and informed our decisions regarding model improvements or refinements.

Therefore, it is crucial to carefully evaluate and select the appropriate architecture based on the specific goals and constraints of each task in the field of computational music generation and interaction. In conclusion, our research has contributed to the field of computational music generation and interaction by advancing our understanding of the capabilities and implications of different model types. The successful integration of different computational architectures showcased the potential for combining their strengths to model complex musical interactions. These findings provide a foundation for future research, where further advancements can be made by exploring advanced architectures, larger datasets, and more diverse tasks. By continuing to push the boundaries of computational music generation and interaction, we can unlock new possibilities for creative expression, human-computer collaborations, and the evolution of music technology.

8.2 Future Directions

Here we discuss and organize in a list some potential future research directions and provide recommendations based on the findings and limitations identified in our research as follows:

Gesture recognition systems: Future research in gesture recognition can focus on exploring recent deep learning architectures, such as Transformer models, Graph Neural Networks, or diffusion models, to capture more complex spatio-temporal patterns and improve gesture recognition performance. Additionally, investigating the integration of additional modalities, such as audio or inertial sensors, can enhance gesture recognition models and enable a more comprehensive understanding of musical gestures. Techniques for transfer learning and domain adaptation can be explored to transfer knowledge between gesture recognition tasks or adapt models to new users or musical contexts, reducing the need for extensive training on new datasets. Furthermore, developing user-independent gesture recognition systems that accurately recognize gestures from different individuals without requiring individual training data would be valuable in increasing the versatility and applicability of such systems.

Virtual musical interaction: To advance virtual musical interaction systems, further research may explore multiple directions. One area of focus is real-time performance optimization, developing techniques to enhance system performance across diverse hardware configurations and platforms, ensuring seamless and responsive experiences. Another aspect is the pursuit of enhanced realism and expressiveness in virtual musical instruments, encompassing both visual and audio feedback, and capturing the intricate nuances of instrumental performance. Additionally, the integration of machine learning and user feedback can be leveraged to create adaptive systems that learn from user input, enabling virtual musical instruments to personalize their responses based on individual preferences and playing styles. Lastly, collaborative virtual environments present an exciting opportunity to facilitate remote musical collaboration and improvisation, allowing multiple users to interact and perform together.

Audio-driven dance motion synthesis: Future research in dance motion synthesis models may explore several techniques to capture and synthesize motion sequences with enhanced realism and expressiveness, focusing on fine-grained details like subtle body movements, joint rotations, and expressive qualities of different dance styles. The integration of dynamics and physics-based modeling can be explored to enhance dance motion synthesis by accurately capturing the physical characteristics and dynamics of human movement. Additionally, the fusion of additional modalities, such as textual cues, can be explored to improve the generation of dance motion sequences and enable synchronized and meaningful interactions between music and dance. Furthermore, methods can be developed to personalize dance motion synthesis systems, allowing users to customize the generated sequences to match their preferred dance style, level of expertise, or artistic preferences.

Jazz accompaniment generation: Future diractions in this area may incude advanced harmonic and melodic analysis techniques, which can be integrated into the improvisation model to enhance the artificial accompanist's understanding of the musical context and enable more musically coherent responses to the human soloist. Multimodal approaches can be investigated to model and simulate the expressive variations inherent in jazz improvisation, capturing the nuances of different artists, improvisational styles, and characteristic phrasing. Additionally, the development of collaborative improvisation systems can be explored, creating platforms that facilitate interaction and interplay between multiple human and artificial improvisers, fostering creative collaborations and providing new musical experiences.

Data availability and information encoding: To overcome the limitations related to data availability and encoding, future research can explore methods to curate diverse and comprehensive datasets that encompass a wide range of musical genres, styles, and cultural contexts. These datasets should include both symbolic and audio information, accompanied by sufficient metadata, to facilitate extensive and generalizable research. Another area of focus is the investigation of advanced symbolic encoding approaches that can capture more nuanced musical information, preserving and representing elements

such as musical structure, dynamics, and expressiveness more effectively. Additionally, researchers may explore the integration of multimodal architectures with audio representations, such as spectrograms or waveforms, enabling direct synthesis or analysis of music without relying solely on symbolic annotations. Lastly, collaborative efforts and standardization should be promoted in data collection, encoding, and sharing practices to foster the development of larger and more diverse datasets. This will facilitate reproducibility and comparability across research studies, ultimately advancing the field as a whole.

Overall, these research directions hold immense potential to push the boundaries of creativity, realism, and interactive musical experiences. They can revolutionize the way we interact with music, towards enabling new forms of expression, collaboration, and artistic possibilities, where technology seamlessly intertwines with human creativity, shaping the future of music.

Bibliography

[1] Harry McGurk and John MacDonald. “Hearing lips and seeing voices”. In: *Nature* 264.5588 (1976), pp. 746–748.

[2] John MacDonald. “Hearing lips and seeing voices: the origins and development of the ‘McGurk Effect’and reflections on audio–visual speech perception over the last 40 years”. In: *Multisensory Research* 31.1-2 (2018), pp. 7–18.

[3] Perceptual Science Lab. *McGurk Effect: auditory /ba/ + visual /ga/ = /da/*. `https://mambo.ucsc.edu/demos/`. Accessed: 2023-2-06. 2015.

[4] John MacDonald and Harry McGurk. “Visual influences on speech perception processes”. In: *Perception & psychophysics* 24.3 (1978), pp. 253–257.

[5] Virginie Van Wassenhove, Ken W Grant, and David Poeppel. “Temporal window of integration in auditory-visual speech perception”. In: *Neuropsychologia* 45.3 (2007), pp. 598–607.

[6] Julian Keil, Nadia Müller, Niklas Ihssen, and Nathan Weisz. “On the variability of the McGurk effect: audiovisual integration depends on prestimulus brain states”. In: *Cerebral Cortex* 22.1 (2012), pp. 221–231.

[7] Annalisa Setti, Kate E Burke, RoseAnne Kenny, and Fiona N Newell. “Susceptibility to a multisensory speech illusion in older persons is driven by perceptual processes”. In: *Frontiers in psychology* 4 (2013), p. 575.

[8] Kerry P Green and Linda W Norrix. “Acoustic cues to place of articulation and the McGurk effect: the role of release bursts, aspiration, and formant transitions”. In: *Journal of Speech, Language, and Hearing Research* 40.3 (1997), pp. 646–665.

[9] Jintao Jiang and Lynne E Bernstein. “Psychophysics of the McGurk and other audiovisual speech integration effects.” In: *Journal of Experimental Psychology: Human Perception and Performance* 37.4 (2011), p. 1193.

[10] Michael J Crosse, John S Butler, and Edmund C Lalor. “Congruent visual speech enhances cortical entrainment to continuous auditory speech in noise-free conditions”. In: *Journal of Neuroscience* 35.42 (2015), pp. 14195–14204.

[11] Gregory Hickok et al. “Neural networks supporting audiovisual integration for speech: A large-scale lesion study”. In: *Cortex* 103 (2018), pp. 360–371.

[12] Dominic William Massaro. *Speech perception by ear and eye: A paradigm for psychological inquiry.* Lawrence Erlbaum Associates, Inc, 1987.

[13] Dominic W Massaro and Daniel Friedman. “Models of integration given multiple sources of information.” In: *Psychological review* 97.2 (1990), p. 225.

[14] DW Massaro and MM Cohen. “The paradigm and the fuzzy logical model of perception are alive and well”. In: *Journal of experimental psychology. General* 122.1 (1993), pp. 115–124.

[15] Dominic W Massaro and David G Stork. “Speech recognition and sensory integration: a 240-year-old theorem helps explain how people and machines can integrate auditory and visual information to understand speech”. In: *American Scientist* 86.3 (1998), pp. 236–244.

[16] Dominic W Massaro. *Perceiving talking faces: From speech perception to a behavioral principle*. Mit Press, 1998.

[17] Dominic W Massaro. “The McGurk effect: Auditory visual speech perception’s piltdown man”. In: *Proceedings of the 14th International Conference on Auditory-Visual Speech Processin (AVSP 2017), Stockholm, Sweden*. Aug. 2017, pp. 131–136.

[18] Michael C. Corballis. “Language as gesture”. In: *Human Movement Science* 28.5 (2009), pp. 556–565. ISSN: 0167-9457. DOI: 10.1016/j.humov.2009.07.003.

[19] David F Armstrong, William C Stokoe, and Sherman E Wilcox. *Gesture and the nature of language*. Cambridge University Press, 1995.

[20] Roel M. Willems and Peter Hagoort. “Neural evidence for the interplay between language, gesture, and action: A review”. In: *Brain and Language* 101.3 (2007). Gesture, Brain, and Language, pp. 278–289. ISSN: 0093-934X. DOI: 10.1016/j.bandl.2007.03.004.

[21] Nathan Oesch. “Music and language in social interaction: Synchrony, antiphony, and functional origins”. In: *Frontiers in Psychology* 10 (2019), p. 1514.

[22] Alessandro Dell’Anna, Marc Leman, and Annamaria Berti. “Musical interaction reveals music as embodied language”. In: *Frontiers in Neuroscience* 15 (2021), p. 667838.

[23] Karen M. Ludke and Hanna Weinmann. *European Music Portfolio: A Creative Way into Languages TEACHER’S HANDBOOK*. English. European Commission, Sept. 2012.

[24] Mónica Rodríguez-Bonces. “A basis for the design of a curriculum incorporating music and drama in children’s english language instruction”. In: *Profile Issues in TeachersProfessional Development* 19.2 (2017), pp. 203–223.

[25] Sandra E Trehub, Judith Becker, and Iain Morley. “Cross-cultural perspectives on music and musicality”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 370.1664 (2015), p. 20140096.

[26] John Laver and Laver John. *Principles of phonetics*. Cambridge university press, 1994.

[27] Fred Lerdahl and Ray S Jackendoff. *A Generative Theory of Tonal Music, reissue, with a new preface*. MIT Press, 1996.

[28] Thomas Fritz et al. “Universal recognition of three basic emotions in music”. In: *Current biology* 19.7 (2009), pp. 573–576.

[29] Patrick E Savage, Steven Brown, Emi Sakai, and Thomas E Currie. “Statistical universals reveal the structures and functions of human music”. In: *Proceedings of the National Academy of Sciences* 112.29 (2015), pp. 8987–8992.

[30] Daniel Weinstein, Jacques Launay, Eiluned Pearce, Robin IM Dunbar, and Lauren Stewart. “Singing and social bonding: changes in connectivity and pain threshold as a function of group size”. In: *Evolution and Human Behavior* 37.2 (2016), pp. 152–158.

[31] Patrik Vuilleumier and Wiebke Trost. “Music and emotions: from enchantment to entrainment”. In: *Annals of the New York Academy of Sciences* 1337.1 (2015), pp. 212–222.

[32] Steven J Mithen. *The singing Neanderthals: The origins of music, language, mind, and body*. Harvard University Press, 2006.

[33] Ian Cross. “Music and communication in music psychology”. In: *Psychology of music* 42.6 (2014), pp. 809–819.

[34] Steven Brown, Michael J Martinez, and Lawrence M Parsons. “Passive music listening spontaneously engages limbic and paralimbic systems”. In: *Neuroreport* 15.13 (2004), pp. 2033–2037.

[35] Valorie N Salimpoor, Iris Van Den Bosch, Natasa Kovacevic, Anthony Randal McIntosh, Alain Dagher, and Robert J Zatorre. “Interactions between the nucleus accumbens and auditory cortices predict music reward value”. In: *Science* 340.6129 (2013), pp. 216–219.

[36] F Richard Moore. *Elements of computer music*. Prentice-Hall, Inc., 1990.

[37] Dave Collins. *The Act of Musical Composition: Studies in the creative process*. Routledge, 2016.

[38] Arthur Cropley. “In praise of convergent thinking”. In: *Creativity research journal* 18.3 (2006), pp. 391–404.

[39] François Delalande. “Towards an Analysis of Compositional Strategies 1”. In: *Circuit* 17.1 (2007), pp. 11–26.

[40] Marc Leman. “An embodied approach to music semantics”. In: *Musicae Scientiae* 14.1_suppl (2010), pp. 43–67.

[41] António Baía Reis and Mark Ashmore. “From video streaming to virtual reality worlds: an academic, reflective, and creative study on live theatre and performance in the metaverse”. In: *International Journal of Performance Arts and Digital Media* 18.1 (2022), pp. 7–28.

[42] Anna Xambó, Alexander Lerch, and Jason Freeman. “Music information retrieval in live coding: a theoretical framework”. In: *Computer Music Journal* 42.4 (2018), pp. 9–25.

[43] Baptiste Caramiaux and Marco Donnarumma. “Artificial intelligence in music and performance: a subjective art-research inquiry”. In: *Handbook of Artificial Intelligence for Music: Foundations, Advanced Approaches, and Developments for Creativity* (2021), pp. 75–95.

[44] Marc Leman. *Embodied music cognition and mediation technology*. MIT Press, 2007.

[45] Adrian C North, David J Hargreaves, and Jon J Hargreaves. “Uses of music in everyday life”. In: *Music perception* 22.1 (2004), pp. 41–77.

[46] Andrea Schiavio, Dylan van der Schyff, Julian Cespedes-Guevara, and Mark Reybrouck. “Enacting musical emotions. Sense-making, dynamic systems, and the embodied mind”. In: *Phenomenology and the Cognitive Sciences* 16 (2017), pp. 785–809.

[47] Wayne Bowman. “Cognition and the body: Perspectives from music education”. In: *Knowing bodies, moving minds: Towards embodied teaching and learning* (2004), pp. 29–50.

[48] Andrea Schiavio, Dylan Van der Schyff, Michele Biasutti, Nikki Moran, and Richard Parncutt. “Instrumental technique, expressivity, and communication. A qualitative study on learning music in individual and collective settings”. In: *Frontiers in psychology* 10 (2019), p. 737.

[49] Eleonora Concina. “The role of metacognitive skills in music learning and performing: theoretical features and educational implications”. In: *Frontiers in Psychology* 10 (2019), p. 1583.

[50] Marissa Silverman. “Sense-making, meaningfulness, and instrumental music education”. In: *Frontiers in Psychology* 11 (2020), p. 837.

[51] Atau Tanaka and Marco Donnarumma. “The body as musical instrument”. In: *The Oxford handbook of music and the body* (2019), pp. 79–96.

[52] John Granzow, Matias Vilaplana, and Anıl Çamcı. “Capturing kinetic wave demonstrations for sound control”. In: *Proceedings of the 15th International Audio Mostly Conference*. 2020, pp. 273–276.

[53] Federico Ghelli Visi and Atau Tanaka. “Interactive machine learning of musical gesture”. In: *Handbook of artificial intelligence for music: Foundations, advanced approaches, and developments for creativity* (2021), pp. 771–798.

[54] Marc Leman, Pieter-Jan Maes, Luc Nijs, and Edith Van Dyck. “What is embodied music cognition?” In: *Springer handbook of systematic musicology* (2018), pp. 747–760.

[55] Judith Lynne Hanna. *Dancing to learn: The brain’s cognition, emotion, and movement*. Rowman & Littlefield, 2014.

[56] Matthew Turk. “Multimodal interaction: A review”. In: *Pattern recognition letters* 36 (2014), pp. 189–195.

[57] Alejandro Jaimes and Nicu Sebe. “Multimodal human–computer interaction: A survey”. In: *Computer vision and image understanding* 108.1-2 (2007), pp. 116–134.

[58] Francesco Cutugno, Vincenza Anna Leano, Roberto Rinaldi, and Gianluca Mignini. “Multimodal Framework for Mobile Interaction”. In: *Proceedings of the International Working Conference on Advanced Visual Interfaces*. AVI ’12. Capri Island, Italy: Association for Computing Machinery, 2012, pp. 197–203. ISBN: 9781450312875. DOI: 10.1145/2254556.2254592.

[59] Théo Jourdan and Baptiste Caramiaux. “Machine Learning for Musical Expression: A Systematic Literature Review”. preprint. Apr. 2023. URL: https://hal.science/hal-04075492.

[60] Torgrim Rudland Næss and Charles Patrick Martin. “A Physical Intelligent Instrument using Recurrent Neural Networks”. In: *Proceedings of the International Conference on New Interfaces for Musical Expression*. Universidade Federal do Rio Grande do Sul. 2019, pp. 79–82.

[61] Charles Patrick Martin, Kyrre Glette, Tønnes Frostad Nygaard, and Jim Torresen. “Understanding musical predictions with an embodied interface for musical machine learning”. In: *Frontiers in Artificial Intelligence* 3 (2020), p. 6.

[62] Shulei Ji, Jing Luo, and Xinyu Yang. “A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions”. In: *arXiv preprint arXiv:2011.06801* (2020).

[63] Carlos Hernandez-Olivan and Jose R Beltran. “Music composition with deep learning: A review”. In: *Advances in Speech and Music Technology: Computational Aspects and Applications* (2022), pp. 25–50.

[64] Jean-Pierre Briot and François Pachet. “Deep learning for music generation: challenges and directions”. In: *Neural Computing and Applications* 32.4 (2020), pp. 981–993.

[65] Miguel Civit, Javier Civit-Masot, Francisco Cuadrado, and Maria J Escalona. “A systematic review of artificial intelligence-based music generation: Scope, applications, and future trends”. In: *Expert Systems with Applications* (2022), p. 118190.

[66] Nan Jiang, Sheng Jin, Zhiyao Duan, and Changshui Zhang. “RL-Duet: Online Music Accompaniment Generation Using Deep Reinforcement Learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.01 (Apr. 2020), pp. 710–718. DOI: 10.1609/aaai.v34i01.5413.

[67] Fatemeh Jamshidi, Daniela Marghitu, and Richard Chapman. “Developing an online music teaching and practicing platform via machine learning: a review paper”. In: *Universal Access in Human-Computer Interaction. Access to Media, Learning and Assistive Environments: 15th International Conference, UAHCI 2021, Held as Part of the 23rd HCI International Conference, HCII 2021, Virtual Event, July 24–29, 2021, Proceedings, Part II*. Springer. 2021, pp. 95–108.

[68] Michael Zbyszyński, Balandino Di Donato, Federico Ghelli Visi, and Atau Tanaka. “Gesture-Timbre Space: Multidimensional Feature Mapping Using Machine Learning and Concatenative Synthesis”. In: *Perception, Representations, Image, Sound, Music: 14th Interna-*

tional Symposium, CMMR 2019, Marseille, France, October 14–18, 2019, Revised Selected Papers 14. Springer. 2021, pp. 600–622.

[69] Ming Jin Cheok, Zaid Omar, and Mohamed Hisham Jaward. “A review of hand gesture and sign language recognition techniques”. In: *International Journal of Machine Learning and Cybernetics* 10 (2019), pp. 131–153. DOI: 10.1007/s13042-017-0705-5.

[70] Beatrice van Amsterdam, Matthew J. Clarkson, and Danail Stoyanov. “Gesture Recognition in Robotic Surgery: A Review”. In: *IEEE Transactions on Biomedical Engineering* 68.6 (2021), pp. 2021–2035. DOI: 10.1109/TBME.2021.3054828.

[71] Stefania Serafin, Cumhur Erkut, Juraj Kojs, Niels C. Nilsson, and Rolf Nordahl. “Virtual Reality Musical Instruments: State of the Art, Design Principles, and Future Directions”. In: *Computer Music Journal* 40.3 (Sept. 2016), pp. 22–40. ISSN: 0148-9267. DOI: 10.1162/COMJ_a_00372.

[72] Alvaro Sarasua, Baptiste Caramiaux, and Atau Tanaka. “Machine Learning of Personal Gesture Variation in Music Conducting”. In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. San Jose, California, USA: Association for Computing Machinery, 2016, pp. 3428–3432. ISBN: 9781450333627. DOI: 10.1145/2858036.2858328.

[73] Said Yacine Boulahia, Eric Anquetil, Franck Multon, and Richard Kulpa. “Dynamic hand gesture recognition based on 3D pattern assembled trajectories”. In: *7th IEEE International Conference on Image Processing Theory, Tools and Applications (IPTA 2017), Montreal, QC, Canada*. Dec. 2017.

[74] George ElKoura and Karan Singh. “Handrix: animating the human hand”. In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association. 2003, pp. 110–119.

[75] W. Lu, Z. Tong, and J. Chu. “Dynamic Hand Gesture Recognition With Leap Motion Controller”. In: *IEEE Signal Processing Letters* 23.9 (Sept. 2016), pp. 1188–1192.

[76] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz. “Online Detection and Classification of Dynamic Hand Gestures with Recurrent 3D Convolutional Neural Networks”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA*. June 2016, pp. 4207–4215.

[77] Alexandre Bouënard, Marcelo MM Wanderley, and Sylvie Gibet. “Gesture control of sound synthesis: Analysis and classification of percussion gestures”. In: *Acta Acustica united with Acustica* 96.4 (2010), pp. 668–677. DOI: 10.3813/AAA.918321.

[78] Jihyun Han and Nicolas Gold. “Lessons Learned in Exploring the Leap Motion(TM) Sensor for Gesture-based Instrument Design”. In: *Proceedings of the International Conference on New Interfaces for Musical Expression*. London, United Kingdom: Goldsmiths, University of London, June 2014, pp. 371–374.

[79] Marcella Mandanici, Antonio Rodà, and Sergio Canazza. “A conceptual framework for motion based music applications”. In: *2nd IEEE VR Workshop on Sonic Interactions for Virtual Environments (SIVE@VR 2015), Arles, France*. Mar. 2015, pp. 9–13.

[80] Yingxue Zhang, Siqi Liu, Lu Tao, Chun Yu, Yuanchun Shi, and Yingqing Xu. “ChinAR: Facilitating Chinese Guqin learning through interactive projected augmentation”. In: *Proceedings of the 3rd International Symposium of Chinese CHI (Chinese CHI ’15), Seoul, Republic of Korea*. Apr. 2015, pp. 23–31.

[81] C. P. Martin, K. Olav Ellefsen, and J. Torresen. “Deep Predictive Models in Interactive Music”. In: *ArXiv e-prints* (Jan. 2018).

[82] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak. “Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks”. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, QLD, Australia*. Apr. 2015, pp. 4580–4584.

[83] Baptiste Caramiaux and Atau Tanaka. “Machine Learning of Musical Gestures”. In: *Proceedings of the International Conference on New Interfaces for Musical Expression*. Daejeon, Republic of Korea, May 2013, pp. 513–518.

[84] Grigoris Bastas, Kosmas Kritsis, and Vassilis Katsouros. “Air-Writing Recognition using Deep Convolutional and Recurrent Neural Network Architectures”. In: *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2020, pp. 7–12. DOI: 10.1109/ICFHR2020.2020.00013.

[85] S. Mitra and T. Acharya. “Gesture Recognition: A Survey”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37.3 (May 2007), pp. 311–324.

[86] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. “Hand gesture recognition with leap motion and kinect devices”. In: *Image Processing (ICIP), 2014 IEEE International Conference on*. IEEE. 2014, pp. 1565–1569.

[87] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. “Hand gesture recognition with jointly calibrated leap motion and depth sensor”. In: *Multimedia Tools and Applications* 75.22 (2016), pp. 14991–15015.

[88] Edgar Hemery, Sotiris Manitsaris, Fabien Moutarde, Christina Volioti, and Athanasios Manitsaris. “Towards the design of a natural user interface for performing and learning musical gestures”. In: *Procedia Manufacturing* 3 (2015), pp. 6329–6336.

[89] Enkhtogtokh Togootogtokh, Timothy K Shih, WGCW Kumara, Shih-Jung Wu, Shih-Wei Sun, and Hon-Hang Chang. “3D finger tracking and recognition image processing for real-time music playing with depth sensors”. In: *Multimedia Tools and Applications* (2017), pp. 1–16.

[90] Jiayue Wu, Mark Rau, Yun Zhang, Yijun Zhou, and Matt Wright. “Towards Robust Tracking with an Unreliable Motion Sensor Using Machine Learning”. In: *Proceedings of the International Conference on New Interfaces for Musical Expression*. Aalborg University Copenhagen, 2017, pp. 42–47.

[91] Robert McCartney, Jie Yuan, and Hans-Peter Bischof. “Gesture recognition with the leap motion controller”. In: *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp). 2015, pp. 3–9.

[92] P. Molchanov, S. Gupta, K. Kim, and J. Kautz. “Hand gesture recognition with 3D convolutional neural networks”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. June 2015, pp. 1–7.

[93] Guillaume Devineau, Fabien Moutarde, Wang Xi, and Jie Yang. “Deep Learning for Hand Gesture Recognition on Skeletal Data”. In: *13th IEEE International Conference on Automatic Face & Gesture Recognition, FG 2018, Xi’an, China*. May 2018, pp. 106–113.

[94] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vandeborre, J. Guerry, B. Le Saux, and D. Filliat. “3D Hand Gesture Recognition Using a Depth and Skeletal Dataset”. In: *Eurographics Workshop on 3D Object Retrieval*. Ed. by Ioannis Pratikakis, Florent Dupont, and Maks Ovsjanikov. The Eurographics Association, 2017. ISBN: 978-3-03868-030-7. DOI: 10.2312/3dor.20171049.

[95] Huy-Hieu Pham, Louahdi Khoudour, Alain Crouzil, Pablo Zegers, and Sergio A. Velastin. “Exploiting deep residual networks for human action recognition from skeletal data”. In: *Computer Vision and Image Understanding* 170 (2018), pp. 51–66. DOI: 10.1016/j.cviu.2018.03.003.

[96] Rodrigo Schramm, Cláudio Rosito Jung, and Eduardo Reck Miranda. “Dynamic time warping for music conducting gestures evaluation”. In: *IEEE Transactions on Multimedia* 17.2 (2015), pp. 243–255.

[97] Di Wu et al. “Deep dynamic neural networks for multimodal gesture segmentation and recognition”. In: *IEEE transactions on pattern analysis and machine intelligence* 38.8 (2016), pp. 1583–1597.

[98] Ayanava Sarkar, Alexander Gepperth, Uwe Handmann, and Thomas Kopinski. “Dynamic Hand Gesture Recognition for Mobile Systems Using Deep LSTM”. In: *International Conference on Intelligent Human Computer Interaction*. Springer. 2017, pp. 19–31.

[99] Juan C. Núñez, Raúl Cabido, Juan J. Pantrigo, Antonio S. Montemayor, and José F. Vélez. “Convolutional Neural Networks and Long Short-Term Memory for skeleton-based human activity and hand gesture recognition”. In: *Pattern Recognition* 76 (2018), pp. 80–94. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2017.10.033.

[100] Chinmaya R. Naguri and Razvan C. Bunescu. “Recognition of Dynamic Hand Gestures from 3D Motion Data Using LSTM and CNN Architectures”. In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE. 2017, pp. 1130–1133. DOI: 10.1109/ICMLA.2017.00013.

[101] Koustav Mullick and Anoop M. Namboodiri. “Learning deep and compact models for gesture recognition”. In: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2017, pp. 3998–4002. DOI: 10.1109/ICIP.2017.8297033.

[102] Sergio Escalera et al. “ChaLearn Looking at People Challenge 2014: Dataset and Results”. In: *Computer Vision - ECCV 2014 Workshops*. Ed. by Lourdes Agapito, Michael M. Bronstein, and Carsten Rother. Springer International Publishing, 2015, pp. 459–473. ISBN: 978-3-319-16178-5. DOI: 10.1007/978-3-319-16178-5_32.

[103] Ultraleap. *Leap Motion Hand Hierarchy*. https://blog.leapmotion.com/getting-started-leap-motion-sdk/hand-hierarchy/. Accessed: 2023-2-06. 2014.

[104] Kosmas Kritsis, Aggelos Gkiokas, Maximos Kaliakatsos-Papakostas, Vassilis Katsouros, and Aggelos Pikrakis. *Leap Motion Hand Gestures for Interaction with 3D Virtual Music Instruments (LMHGIf3DVMI)*. June 2018. DOI: 10.5281/zenodo.1260336.

[105] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[106] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. DOI: 10.48550/arXiv.1412.6980.

[107] F. Wilcoxon. “Individual comparisons by ranking methods”. In: *Biometrics Bulletin* 1.6 (1945), pp. 80–83.

[108] Shawn Trail et al. “Non-invasive sensing and gesture control for pitched percussion hyper-instruments using the Kinect”. In: *12th International Conference on New Interfaces for Musical Expression, NIME 2012, Ann Arbor, Michigan, USA, May 21-23, 2012*. 2012. DOI: 10.5281/zenodo.1178435.

[109] Teemu Mäki-Patola, Aki Kanerva, Juha Laitinen, and Tapio Takala. “Experiments with Virtual Reality Instruments”. In: *New Interfaces for Musical Expression, NIME-05, Proceedings, Vancouver, May 26-28, 2005*. 2005, pp. 11–16. DOI: 10.5281/zenodo.1176780.

[110] Martin Hachet. “3D User Interfaces, from Mobile Devices to Immersive Virtual Environments”. PhD thesis. Université Sciences et Technologies-Bordeaux I, 2010.

[111] R. Medeiros, F. Calegario, G. Cabral, and G. Ramalho. “Challenges in Designing New Interfaces for Musical Expression”. In: *Design, User Experience, and Usability. Theories, Methods, and Tools for Designing the User Experience - Third International Conference, DUXU 2014, Held as Part of HCI International 2014*. 2014, pp. 643–652. DOI: 10.1007/978-3-319-07668-3_62.

[112] Ulysse Rosselet and Alain Renaud. “Jam On: a new interface for web-based collective music performance”. In: *13th International Conference on New Interfaces for Musical Expression, NIME 2013, Daejeon, Republic of Korea, May 27-30, 2013*. 2013, pp. 394–399. DOI: 10.5281/zenodo.1178650.

[113] Benjamin Taylor and Jesse T. Allison. “Gesture Capture, Processing, and Asynchronous Playback within Web Audio Instruments”. In: *Looking Back, Looking Forward: Proceedings of the 41st International Computer Music Conference, ICMC 2015, Denton, TX, USA, September 25 - October 1, 2015*. 2015. DOI: 2027/spo.bbp2372.2015.078.

[114] Gerard Roma, Anna Xambó, and Jason Freeman. “Handwaving: Gesture Recognition for Participatory Mobile Music”. In: *Proceedings of the 12th International Audio Mostly Conference on Augmented and Participatory Sound and Music Experiences*. AM ’17. London, United Kingdom: Association for Computing Machinery, 2017. ISBN: 9781450353731. DOI: 10.1145/3123514.3123538.

[115] Álvaro Sarasúa, Baptiste Caramiaux, Atau Tanaka, and Miguel Ortiz. "Datasets for the Analysis of Expressive Musical Gestures". In: *Proceedings of the 4th International Conference on Movement Computing, London, United Kingdom, June 28-30, 2017*. 2017, 13:1–13:4. DOI: 10.1145/3077981.3078032.

[116] Balandino Di Donato, James Dooley, Jason Hockman, Simon Hall, and Jamie Bullock. "MyoSpat: A hand-gesture controlled system for sound and light projections manipulation". In: *Proceedings of the 43rd International Computer Music Conference*. Ed. by Margaret Schedel. Oct. 2017, pp. 335–340. DOI: 2027/spo.bbp2372.2017.056.

[117] Marcella Mandanici and Sergio Canazza. "The hand composer: gesture-driven music composition machines". In: *1st international workshop on computer and robotic Systems for Automatic Music Performance (SAMP14)*. 2014, pp. 553–560.

[118] Jules Françoise, Olivier Chapuis, Sylvain Hanneton, and Frédéric Bevilacqua. "SoundGuides: Adapting Continuous Auditory Feedback to Users". In: *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA '16. San Jose, California, USA: ACM, 2016, pp. 2829–2836. ISBN: 978-1-4503-4082-3. DOI: 10.1145/2851581.2892420.

[119] Lamtharn Hantrakul and Konrad Kaczmarek. "Implementations of the Leap Motion in sound synthesis, effects modulation and assistive performance tools". In: *Music Technology meets Philosophy - From Digital Echos to Virtual Ethos: Joint Proceedings of the 40th International Computer Music Conference, ICMC 2014, and the 11th Sound and Music Computing Conference, SMC 2014, Athens, Greece, September 14-20, 2014*. 2014. DOI: 2027/spo.bbp2372.2014.100.

[120] Álvaro Sarasúa, Baptiste Caramiaux, and Atau Tanaka. "Machine Learning of Personal Gesture Variation in Music Conducting". In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, San Jose, CA, USA, May 7-12, 2016*. 2016, pp. 3428–3432. DOI: 10.1145/2858036.2858328.

[121] Jules Françoise, Norbert Schnell, and Frédéric Bevilacqua. “A multimodal probabilistic model for gesture-based control of sound synthesis”. In: *ACM Multimedia Conference, MM ’13, Barcelona, Spain, October 21-25, 2013*. 2013, pp. 705–708. DOI: 10.1145/2502081.2502184.

[122] James Leonard, Claude Cadoz, Nicolas Castagne, Jean-Loup Florens, and Annie Luciani. “A Virtual Reality Platform for Musical Creation: GENESIS-RT”. In: *Sound, Music, and Motion: 10th International Symposium, CMMR 2013, Marseille, France, October 15-18, 2013. Revised Selected Papers*. Marseille, France: Springer-Verlag, 2014, pp. 346–371. ISBN: 978-3-319-12975-4. DOI: 10.1007/978-3-319-12976-1_22.

[123] Florent Berthaut, Cagan Arslan, and Laurent Grisoni. “Revgest: Augmenting Gestural Musical Instruments with Revealed Virtual Objects”. In: *Proceedings of the International Conference on New Interfaces for Musical Expression* (Copenhagen, Denmark). Zenodo, June 2017, pp. 180–185. DOI: 10.5281/zenodo.1176213.

[124] K. Kilteni, I. Bergstrom, and M. Slater. “Drumming in Immersive Virtual Reality: The Body Shapes the Way We Play”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.4 (Apr. 2013), pp. 597–605. ISSN: 1077-2626. DOI: 10.1109/TVCG.2013.29.

[125] Rene Emile Causse, Joel Bensoam, and Nicholas Ellis. “Modalys, a physical modeling synthesizer: More than twenty years of researches, developments, and musical uses”. In: *The Journal of the Acoustical Society of America* 130.4 (2011), pp. 2365–2365. DOI: 10.1121/1.3654475.

[126] Andreas Aristidou, Daniel Cohen-Or, Jessica K. Hodgins, Yiorgos Chrysanthou, and Ariel Shamir. “Deep Motifs and Motion Signatures”. In: *ACM Trans. Graph.* 37.6 (Dec. 2018). ISSN: 0730–0301. DOI: 10.1145/3272127.3275038.

[127] Andreas Aristidou, Ariel Shamir, and Yiorgos Chrysanthou. “Digital Dance Ethnography: Organizing Large Dance Collections”. In: *J. Comput. Cult. Herit.* 12.4 (Nov. 2019). ISSN: 1556–4673. DOI: 10.1145/3344383.

[128] Adrienne L. Kaeppler. “Dance Ethnology and the Anthropology of Dance”. In: *Dance Research Journal* 32 (2000), pp. 116–125.

[129] H. Tachibana, K. Uenoyama, and S. Aihara. “Efficiently Trainable Text-to-Speech System Based on Deep Convolutional Networks with Guided Attention”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 4784–4788. DOI: 10.1109/ICASSP.2018.8461829.

[130] Jonathan Shen et al. “Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 4779–4783. DOI: 10.1109/ICASSP.2018.8461368.

[131] Rachel Manzelli, Vijay Thakkar, Ali Siahkamari, and Brian Kulis. “Conditioning Deep Generative Raw Audio Models for Structured Automatic Music”. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference* (Paris, France). Paris, France: ISMIR, Sept. 2018, pp. 182–189. DOI: 10.5281/zenodo.1492375.

[132] Lucas Ferreira and Jim Whitehead. “Learning to Generate Music With Sentiment”. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference* (Delft, The Netherlands). Delft, The Netherlands: ISMIR, Nov. 2019, pp. 384–390. DOI: 10.5281/zenodo.3527824.

[133] Wang Xi, Guillaume Devineau, Fabien Moutarde, and Jie Yang. “Generative Model for Skeletal Human Movements Based on Conditional DC-GAN Applied to Pseudo-Images”. In: *Algorithms* 13.12 (Dec. 2020), p. 319. ISSN: 1999–4893. DOI: 10.3390/a13120319.

[134] Slim Essid et al. “An advanced virtual dance performance evaluator”. In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2012, pp. 2269–2272. DOI: 10.1109/ICASSP.2012.6288366.

[135] Ruiqi Wu et al. “Towards Deep Learning Based Robot Automatic Choreography System”. In: *Intelligent Robotics and Applications - 12th International Conference, ICIRA 2019, Shenyang, China, August 8-11, 2019, Proceedings, Part IV*. Ed. by Haibin Yu, Jinguo Liu,

Lianqing Liu, Zhaojie Ju, Yuwang Liu, and Dalin Zhou. Vol. 11743. Lecture Notes in Computer Science. Springer, 2019, pp. 629–640. DOI: 10.1007/978-3-030-27538-9_54.

[136] Jacky C.P. Chan, Howard Leung, Jeff K.T. Tang, and Taku Komura. "A Virtual Reality Dance Training System Using Motion Capture Technology". In: *IEEE Transactions on Learning Technologies* 4.2 (2011), pp. 187–195. DOI: 10.1109/TLT.2010.27.

[137] Katerina El Raheb, Marina Stergiou, Akrivi Katifori, and Yannis Ioannidis. "Dance Interactive Learning Systems: A Study on Interaction Workflow and Teaching Approaches". In: *ACM Comput. Surv.* 52.3 (June 2019). ISSN: 0360-0300. DOI: 10.1145/3323335.

[138] Hsin-Ying Lee et al. "Dancing to Music". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019.

[139] Shuhei Tsuchida, Satoru Fukayama, Masahiro Hamasaki, and Masataka Goto. "AIST Dance Video Database: Multi-Genre, Multidancer, and Multi-Camera Database for Dance Information Processing". In: *Proceedings of the 20th International Society for Music Information Retrieval Conference* (Delft, The Netherlands). Delft, The Netherlands: ISMIR, Nov. 2019, pp. 501–510. DOI: 10.5281/zenodo.3527854.

[140] Taoran Tang, Jia Jia, and Hanyang Mao. "Dance with Melody: An LSTM-Autoencoder Approach to Music-Oriented Dance Synthesis". In: *Proceedings of the 26th ACM International Conference on Multimedia*. MM '18. Seoul, Republic of Korea: Association for Computing Machinery, 2018, pp. 1598–1606. ISBN: 9781450356657. DOI: 10.1145/3240508.3240526.

[141] Hyemin Ahn, Jaehun Kim, Kihyun Kim, and Songhwai Oh. "Generative Autoregressive Networks for 3D Dancing Move Synthesis From Music". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 3501–3508. DOI: 10.1109/LRA.2020.2977333.

[142] Z. Cao, G. Hidalgo, T. Simon, S. -E. Wei, and Y. Sheikh. "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields".

In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.1 (2021), pp. 172–186. DOI: 10.1109/TPAMI.2019.2929257.

[143] Juheon Lee, Seohyun Kim, and Kyogu Lee. "Automatic Choreography Generation with Convolutional Encoder-decoder Network". In: *Proceedings of the 20th International Society for Music Information Retrieval Conference* (Delft, The Netherlands). Delft, The Netherlands: ISMIR, Nov. 2019, pp. 894–899. DOI: 10.5281/zenodo.3527958.

[144] Yu Qi, Yazhou Liu, and Quansen Sun. "Music-Driven Dance Generation". In: *IEEE Access* 7 (2019), pp. 166540–166550. DOI: 10.1109/ACCESS.2019.2953698.

[145] Ruozi Huang, Huang Hu, Wei Wu, Kei Sawada, Mi Zhang, and Daxin Jiang. "Dance Revolution: Long-Term Dance Generation with Music via Curriculum Learning". In: *International Conference on Learning Representations*. 2021.

[146] Luka Crnkovic-Friis and Louise Crnkovic-Friis. "Generative Choreography using Deep Learning". In: *Proceedings of the Seventh International Conference on Computational Creativity (ICCC)*. 2016.

[147] Benedikte Wallace, Charles P. Martin, and Kristian Nymoen. "Tracing from Sound to Movement with Mixture Density Recurrent Neural Networks". In: *Proceedings of the 6th International Conference on Movement and Computing*. MOCO '19. Tempe, AZ, USA: Association for Computing Machinery, 2019. ISBN: 9781450376549. DOI: 10.1145/3347122.3371376.

[148] Benedikte Wallace, Charles P. Martin, Jim Tørresen, and Kristian Nymoen. "Towards Movement Generation with Audio Features". In: *Proceedings of the Eleventh International Conference on Computational Creativity, ICCC 2020, Coimbra, Portugal, September 7-11, 2020*. Association for Computational Creativity (ACC), 2020, pp. 284–287.

[149] Zijie Ye et al. "ChoreoNet: Towards Music to Dance Synthesis with Choreographic Action Unit". In: *Proceedings of the 28th ACM International Conference on Multimedia*. MM '20. Seattle, WA, USA: Association for Computing Machinery, 2020, pp. 744–752. ISBN: 9781450379885. DOI: 10.1145/3394171.3414005.

[150] Xuanchi Ren, Haoran Li, Zijian Huang, and Qifeng Chen. “Self-Supervised Dance Video Synthesis Conditioned on Music”. In: *Proceedings of the 28th ACM International Conference on Multimedia*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 46–54. ISBN: 9781450379885. DOI: 10.1145/3394171.3413932.

[151] Guofei Sun, Yongkang Wong, Zhiyong Cheng, Mohan S. Kankanhalli, Weidong Geng, and Xiangdong Li. “DeepDance: Music-to-Dance Motion Choreography With Adversarial Learning”. In: *IEEE Transactions on Multimedia* 23 (2021), pp. 497–509. DOI: 10.1109/TMM.2020.2981989.

[152] Trieu H. Trinh, Andrew M. Dai, Thang Luong, and Quoc V. Le. “Learning Longer-term Dependencies in RNNs with Auxiliary Losses”. In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 4972–4981.

[153] J. Martinez, M. J. Black, and J. Romero. “On Human Motion Prediction Using Recurrent Neural Networks”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4674–4683. DOI: 10.1109/CVPR.2017.497.

[154] Divya Saxena and Jiannong Cao. “Generative Adversarial Networks (GANs): Challenges, Solutions, and Future Directions”. In: *ACM Comput. Surv.* 54.3 (May 2021). ISSN: 0360–0300. DOI: 10.1145/3446374.

[155] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. “Convolutional Sequence to Sequence Learning”. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML’17. Sydney, NSW, Australia: JMLR.org, 2017, pp. 1243–1252.

[156] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. “Training Very Deep Networks”. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. NIPS’15. Montreal, Canada: MIT Press, 2015, pp. 2377–2385.

[157] Rui Li, Zhenyu Liu, and Jianrong Tan. “Human motion segmentation using collaborative representations of 3D skeletal sequences”. In: *IET Comput. Vis.* 12.4 (2018), pp. 434–442. DOI: 10.1049/iet-cvi.2016.0385.

[158] Kosmas Kritsis, Maximos Kaliakatsos-Papakostas, Vassilis Katsouros, and Aggelos Pikrakis. “Deep Convolutional and LSTM Neural Network Architectures on Leap Motion Hand Tracking Data Sequences”. In: *2019 27th European Signal Processing Conference (EUSIPCO)*. 2019, pp. 1–5. DOI: 10.23919/EUSIPCO.2019.8902973.

[159] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[160] Daniel Groos, Heri Ramampiaro, and Espen AF Ihlen. “EfficientPose: Scalable single-person pose estimation”. In: *Applied Intelligence* 51.4 (2021), pp. 2518–2533.

[161] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. “DensePose: Dense Human Pose Estimation in the Wild”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018, pp. 7297–7306.

[162] Nikos Kolotouros, Georgios Pavlakos, Michael J. Black, and Kostas Daniilidis. “Learning to Reconstruct 3D Human Pose and Shape via Model-Fitting in the Loop”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019, pp. 2252–2261.

[163] Muhammed Kocabas, Nikos Athanasiou, and Michael J. Black. “VIBE: Video Inference for Human Body Pose and Shape Estimation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020, pp. 5253–5263.

[164] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. “Recurrent Network Models for Human Dynamics”. In: *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. ICCV ’15. USA: IEEE Computer Society, 2015, pp. 4346–4354. ISBN: 9781467383912.

[165] Bochen Li, Akira Maezawa, and Zhiyao Duan. “Skeleton Plays Piano: Online Generation of Pianist Body Movements from MIDI Performance”. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference* (Paris, France). Paris, France: ISMIR, Sept. 2018, pp. 218–224. DOI: 10.5281/zenodo.1492387.

[166] Eli Shlizerman, Lucio Dery, Hayden Schoen, and Ira Kemelmacher-Shlizerman. “Audio to Body Dynamics”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2018, pp. 7574–7583. DOI: 10.1109/CVPR.2018.00790.

[167] Vedran Vukotić, Christian Raymond, and Guillaume Gravier. “Bidirectional Joint Representation Learning with Symmetrical Deep Neural Networks for Multimodal and Crossmodal Applications”. In: *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*. ICMR ’16. New York, New York, USA: Association for Computing Machinery, 2016, pp. 343–346. DOI: 10.1145/2911996.2912064.

[168] Aggelos Gkiokas and Vassilios Katsouros. “Convolutional Neural Networks for Real-Time Beat Tracking: A Dancing Robot Application.” In: *Proceedings of the 18th International Society for Music Information Retrieval Conference* (Suzhou, China). Suzhou, China: ISMIR, Oct. 2017, pp. 286–293. DOI: 10.5281/zenodo.1417737.

[169] Chris Donahue, Zachary C. Lipton, and Julian McAuley. “Dance Dance Convolution”. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, Aug. 2017, pp. 1039–1048.

[170] Andreas Aristidou et al. “Emotion Control of Unstructured Dance Movements”. In: *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. SCA ’17. Los Angeles, California: Association for Computing Machinery, 2017. DOI: 10.1145/3099564.3099566.

[171] Omid Alemi, Jules Françoise, and Philippe Pasquier. “Groovenet: Real-time music-driven dance movement generation using artificial

neural networks". In: *SIGKDD 2017 Workshop on Machine Learning for Creativity (ML4Creativity 2017)*. 2017, pp. 1–6.

[172] Jiaman Li et al. "Learning to Generate Diverse Dance Motions with Transformer". In: *CoRR* abs/2008.08171 (2020). arXiv: 2008.08171.

[173] Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. "Music Conditioned 3D Dance Generation With AIST++". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 13401–13412. DOI: 10.1109/ICCV48922.2021.01315.

[174] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[175] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[176] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.

[177] J. R. Medina and J. Kalita. "Parallel Attention Mechanisms in Neural Machine Translation". In: *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. Dec. 2018, pp. 547–552. DOI: 10.1109/ICMLA.2018.00088.

[178] Anna Currey and Kenneth Heafield. "Incorporating Source Syntax into Transformer-Based Neural Machine Translation". In: *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 24–33. DOI: 10.18653/v1/W19-5203.

[179] Ulme Wennberg and Gustav Eje Henter. "The Case for Translation-Invariant Self-Attention in Transformer-Based Language Models". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Online:

Association for Computational Linguistics, Aug. 2021, pp. 130–140. DOI: 10.18653/v1/2021.acl-short.18.

[180] Tshephisho J. Sefara, Skhumbuzo G. Zwane, Nelisiwe Gama, Hlawulani Sibisi, Phillemon N. Senoamadi, and Vukosi Marivate. "Transformer based Machine Translation for Low-resourced Languages embedded with Language Identification". In: *2021 Conference on Information Communications Technology and Society (ICTAS)*. 2021, pp. 127–132. DOI: 10.1109/ICTAS50802.2021.9394996.

[181] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. "Structured Prediction Helps 3D Human Motion Modelling". In: *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*. IEEE, 2019, pp. 7143–7152. DOI: 10.1109/ICCV.2019.00724.

[182] Brian McFee et al. "librosa: Audio and Music Signal Analysis in Python". In: *Proceedings of the 14th Python in Science Conference*. Ed. by Kathryn Huff and James Bergstra. 2015, pp. 18–24. DOI: 10.25080/Majora-7b98e3ed-003.

[183] Sebastian Böck, Florian Krebs, and Gerhard Widmer. "A Multi-model Approach to Beat Tracking Considering Heterogeneous Music Styles". In: *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*. Ed. by Hsin-Min Wang, Yi-Hsuan Yang, and Jin Ha Lee. 2014, pp. 603–608.

[184] Aäron van den Oord et al. "WaveNet: A Generative Model for Raw Audio". In: *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*. ISCA, 2016, p. 125. DOI: 10.48550/arXiv.1609.03499.

[185] Julian Georg Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. "Recurrent Highway Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 4189–4198.

[186] Aäron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. “Conditional Image Generation with PixelCNN Decoders”. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS’16. Barcelona, Spain: Curran Associates Inc., 2016, pp. 4797–4805. ISBN: 9781510838819.

[187] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. DOI: 10.48550/arXiv.1912.01703.

[188] Yi Zhou, Zimo Li, Shuangjiu Xiao, Chong He, Zeng Huang, and Hao Li. “Auto-Conditioned Recurrent Networks for Extended Complex Human Motion Synthesis”. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[189] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.

[190] Florian Schmidt. “Generalization in Generation: A closer look at Exposure Bias”. In: *Proceedings of the 3rd Workshop on Neural Generation and Translation*. Hong Kong: Association for Computational Linguistics, Nov. 2019, pp. 157–167. DOI: 10.18653/v1/D19-5616.

[191] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. *Deep Learning Techniques for Music Generation*. Springer International Publishing, 2019. DOI: 10.1007/978-3-319-70163-9.

[192] Lejaren A Hiller Jr and Leonard M Isaacson. “Musical Composition with a High Speed Digital Computer”. In: *Audio Engineering Society Convention 9*. Audio Engineering Society. 1957, pp. 154–160.

[193] I. Xenakis. *Formalized music: Thought and Mathematics in Composition*. Indiana University Press, 1963.

[194] Hao-Min Liu and Yi-Hsuan Yang. “Lead Sheet Generation and Arrangement by Conditional Generative Adversarial Network”. In: *17th IEEE International Conference on Machine Learning and Applications (ICMLA 2018)*. IEEE, 2018, pp. 722–727. DOI: 10.1109/ICMLA.2018.00114.

[195] Ziyu Wang and Gus Xia. “A Framework for Automated Pop-song Melody Generation with Piano Accompaniment Arrangement”. In: *6th Conference on Sound and Music Technology (CSMT 2018)*. 2018, pp. 1–9. DOI: 10.48550/arXiv.1812.10906.

[196] Michael C Mozer. “Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing”. In: *Connection Science* 6.2-3 (1994), pp. 247–280. DOI: 10.1080/09540099408915726.

[197] Douglas Eck and Juergen Schmidhuber. “Finding temporal structure in music: Blues improvisation with LSTM recurrent networks”. In: *Proceedings of the 12th IEEE workshop on neural networks for signal processing*. IEEE. 2002, pp. 747–756. DOI: 10.1109/NNSP.2002.1030094.

[198] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. “DeepBach: A Steerable Model for Bach Chorales Generation”. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML’17. Sydney, NSW, Australia, 2017, pp. 1362–1371. DOI: 10.48550/arXiv.1612.01010.

[199] Natasha Jaques, Shixiang Gu, Dzmitry Bahdanau, José Miguel Lobato, Richard E. Turner, and Douglas Eck. “Sequence Tutor: Conservative Fine-Tuning of Sequence Generation Models with KL-control”. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, Aug. 2017, pp. 1645–1654.

[200] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. “Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription”. In: *Proceedings of the 29th International Coference on International*

Conference on Machine Learning. ICML'12. Edinburgh, Scotland: Omnipress, 2012, pp. 1881–1888. ISBN: 9781450312851.

[201] Keunwoo Choi, George Fazekas, and Mark Sandler. "Text-based LSTM networks for automatic music composition". In: *arXiv preprint arXiv:1604.05358* (2016). DOI: 10.48550/arXiv.1604.05358.

[202] Gino Brunner, Yuyi Wang, Roger Wattenhofer, and Jonas Wiesendanger. "JamBot: Music theory aware chord based generation of polyphonic music with LSTMs". In: *IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI 2017)*. 2017, pp. 519–526. DOI: 10.1109/ICTAI.2017.00085.

[203] Hang Chu, Raquel Urtasun, and Sanja Fidler. "Song from PI: A musically plausible network for pop music generation". In: *arXiv preprint arXiv:1611.03477* (2016). DOI: 10.48550/arXiv.1611.03477.

[204] Daniel D. Johnson, Robert M. Keller, and Nicholas Weintraut. "Learning to Create Jazz Melodies Using a Product of Experts". In: *8th International Conference on Computational Creativity (ICCC 2017)*. 2017, pp. 151–158.

[205] Nicholas Trieu and Robert Keller. "JazzGAN: Improvising with Generative Adversarial Networks". In: *Proceedings of the 6th International Workshop on Musical Metacreation*. Salamanca, Spain: MUME, June 2018, p. 8. DOI: 10.5281/zenodo.4285166.

[206] Jingling Wang, Xiaodong Wang, and Juanjuan Cai. "Jazz Music Generation Based on Grammar and LSTM". In: *2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*. 2019, pp. 115–120. DOI: 10.1109/IHMSC.2019.00035.

[207] Roberto De Prisco, Delfina Malandrino, Gianluca Zaccagnino, Rocco Zaccagnino, and Rosalba Zizza. "A Kind of Bio-inspired Learning of mUsic stylE". In: *Computational Intelligence in Music, Sound, Art and Design*. Ed. by João Correia, Vic Ciesielski, and Antonios Liapis. Springer International Publishing, 2017, pp. 97–113. DOI: 10.1007/978-3-319-55750-2_7.

[208] Clelia De Felice, Roberto De Prisco, Delfina Malandrino, Gianluca Zaccagnino, Rocco Zaccagnino, and Rosalba Zizza. "Chorale Music Splicing System: An Algorithmic Music Composer Inspired by

Molecular Splicing". In: *Evolutionary and Biologically Inspired Music, Sound, Art and Design*. Springer International Publishing, 2015, pp. 50–61. DOI: 10.1007/978-3-319-16498-4_5.

[209] Takeshi Hori, Kazuyuki Nakamura, and Shigeki Sagayama. "Jazz piano trio synthesizing system based on HMM and DNN". In: *Proceedings of the 14th Sound and Music Computing Conference 2017, SMC 2017*. Ed. by Tapio Lokki, Jukka Patynen, and Vesa Valimaki. Proceedings of the 14th Sound and Music Computing Conference 2017, SMC 2017. Aalto University, Jan. 2019, pp. 153–158.

[210] Hsiao-Tzu Hung, Chung-Yang Wang, Yi-Hsuan Yang, and Hsin-Min Wang. "Improving Automatic Jazz Melody Generation by Transfer Learning Techniques". In: *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 339–346. DOI: 10.1109/APSIPAASC47483.2019.9023224.

[211] Maximos A Kaliakatsos-Papakostas, Andreas Floros, and Michael N Vrahatis. "Intelligent real-time music accompaniment for constraint-free improvisation". In: *2012 IEEE 24th International Conference on Tools with Artificial Intelligence*. Vol. 1. IEEE. 2012, pp. 444–451. DOI: 10.1109/ICTAI.2012.67.

[212] Patrick Hutchings and Jon McCormack. "Using Autonomous Agents to Improvise Music Compositions in Real-Time". In: *Computational Intelligence in Music, Sound, Art and Design*. Ed. by João Correia, Vic Ciesielski, and Antonios Liapis. Springer International Publishing, 2017, pp. 114–127. DOI: 10.1007/978-3-319-55750-2_8.

[213] Laura Ferguson. "Band-in-a-Box for the General Music Classroom". In: *General Music Today* 18.2 (2005), pp. 7–13. DOI: 10.1177/10483713050180020103.

[214] David Brian Huron. *Sweet anticipation: Music and the psychology of expectation*. MIT press, 2006. DOI: 10.7551/mitpress/6575.001.0001.

[215] Dimos Makris, Maximos Kaliakatsos-Papakostas, Ioannis Karydis, and Katia Lida Kermanidis. "Conditional neural sequence learners for generating drums' rhythms". In: *Neural Computing and Applica-*

tions 31.6 (2019), pp. 1793–1804. DOI: 10.1007/s00521-018-3708-6.

[216] Michael Scott Cuthbert and Christopher Ariza. "Music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data." In: (Aug. 2010), pp. 637–642. DOI: 10.5281/zenodo.1416114.

[217] Martín Abadi et al. "Tensorflow: A system for large-scale machine learning". In: *12th Symposium on Operating Systems Design and Implementation (OSDI 2016)*. 2016, pp. 265–283. DOI: 10.48550/arXiv.1605.08695.

[218] Chris Walshaw. *ABC notation*. https://abcnotation.com/. Accessed: 2023-2-06. 1995.

[219] Piero Weiss and Richard Taruskin. *Music in the Western World*. Cengage Learning, 2007. ISBN: 9780534585990.

[220] Chris Donahue, Huanru Henry Mao, and Julian McAuley. "The NES music database: A multi-instrumental dataset with expressive performance attributes". In: *arXiv preprint arXiv:1806.04278* (2018).

[221] Bob L. T. Sturm. *Benchmarking "music generation systems"?* 2017. URL: https://highnoongmt.wordpress.com/2017/03/19/benchmarking-music-generation-systems/.

[222] Bob L. T. Sturm. *On folk-rnn cheating and music generation*. 2017. URL: https://highnoongmt.wordpress.com/2017/06/15/on-folk-rnn-cheating-and-music-generation/.

[223] Sebastian Garcia-Valencia. "Embeddings as representation for symbolic music". In: *arXiv preprint arXiv:2005.09406* (2020). DOI: 10.48550/arXiv.2005.09406.

[224] Sephora Madjiheurem, Lizhen Qu, and Christian Walder. "Chord2vec: Learning musical chord embeddings". In: *Proceedings of the constructive machine learning workshop at 30th conference on neural information processing systems (NIPS2016), Barcelona, Spain*. 2016.

[225] Jose David Fernández and Francisco Vico. "AI Methods in Algorithmic Composition: A Comprehensive Survey". In: *Journal of Artificial Intelligence Research* 48.1 (Oct. 2013), pp. 513–582. ISSN: 1076-9757. DOI: 10.1613/jair.3908.

[226] Dorien Herremans, Ching-Hua Chuan, and Elaine Chew. “A Functional Taxonomy of Music Generation Systems”. In: *ACM Comput. Surv.* 50.5 (Sept. 2017). ISSN: 0360-0300. DOI: 10.1145/3108242.

[227] Charu C. Aggarwal. *Neural Networks and Deep Learning. A Textbook*. Cham: Springer, 2018, p. 497. ISBN: 978-3-319-94462-3. DOI: 10.1007/978-3-319-94463-0.

[228] Peter M. Todd. “A Connectionist Approach to Algorithmic Composition”. In: *Computer Music Journal* 13.4 (1989), pp. 27–43. ISSN: 01489267, 15315169. URL: http://www.jstor.org/stable/3679551.

[229] Andrej Karpathy. *The Unreasonable Effectiveness of Recurrent Neural Networks*. Blog Post. May 2015. URL: http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

[230] Bob Sturm, Joao Felipe Santos, and Iryna Korshunova. “Folk music style modelling by recurrent neural networks with long short term memory units”. In: *16th International Society for Music Information Retrieval Conference*. 2015.

[231] Cheng-Zhi Anna Huang et al. “Music Transformer: Generating Music with Long-Term Structure”. In: *arXiv preprint arXiv:1809.04281* (2018).

[232] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. “A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 4364–4373. DOI: 10.48550/arXiv.1803.05428.

[233] Mitch Harris, Alan Smaill, and Geraint Wiggins. *Representing music symbolically*. University of Edinburgh, Department of Artificial Intelligence, 1991.

[234] Donald Byrd, Roger D Boyle, Ulf Berggren, David Bainbridge, et al. *Beyond MIDI - The Handbook of Musical Codes*. MIT press, 1997.

[235] Geraint Wiggins, Eduardo Miranda, Alan Smaill, and Mitch Harris. “A Framework for the Evaluation of Music Representation Systems”. In: *Computer Music Journal* 17.3 (1993), pp. 31–42. ISSN: 01489267, 15315169.

[236] Henkjan Honing. “Issues on the representation of time and structure in music”. In: *Contemporary Music Review* 9.1-2 (1993), pp. 221–238. DOI: 10.1080/07494469300640461.

[237] Roger B Dannenberg. “A Brief Survey of Music Representation Issues, Techniques, and Systems”. In: (Mar. 2000). DOI: 10.1184/R1/6587204.v1.

[238] Eric J. Humphrey, Juan Pablo Bello, and Yann LeCun. “Moving Beyond Feature Design: Deep Architectures and Automatic Feature Learning in Music Informatics.” In: *Proceedings of the 13th International Society for Music Information Retrieval Conference*. Porto, Portugal: ISMIR, Oct. 2012, pp. 403–408. DOI: 10.5281/zenodo.1415726.

[239] Néstor Nápoles, Gabriel Vigliensoni, and Ichiro Fujinaga. “Encoding Matters”. In: *Proceedings of the 5th International Conference on Digital Libraries for Musicology*. DLfM ’18. Paris, France: Association for Computing Machinery, 2018, pp. 69–73. ISBN: 9781450365222. DOI: 10.1145/3273024.3273027.

[240] Lucas Theis, Aäron van den Oord, and Matthias Bethge. *A note on the evaluation of generative models*. 2015. arXiv: 1511.01844 [stat.ML].

[241] Kat Agres, Jamie Forth, and Geraint A. Wiggins. “Evaluation of Musical Creativity and Musical Metacreation Systems”. In: *Comput. Entertain.* 14.3 (Dec. 2016). DOI: 10.1145/2967506.

[242] Carolyn Lamb, Daniel G. Brown, and Charles L. A. Clarke. “Evaluating Computational Creativity: An Interdisciplinary Tutorial”. In: *ACM Comput. Surv.* 51 (2018), 28:1–28:34.

[243] Li-Chia Yang and Alexander Lerch. “On the evaluation of generative models in music”. In: *Neural Computing and Applications* (2018), pp. 1–12.

[244] Christopher Ariza. “The Interrogator as Critic: The Turing Test and the Evaluation of Generative Music Systems”. In: *Computer Music Journal* 33.2 (2009), pp. 48–70. DOI: 10.1162/comj.2009.33.2.48.

[245] Bob L Sturm and Oded Ben-Tal. “Taking the models back to music practice: Evaluating generative transcription models built using deep learning”. In: *Journal of Creative Music Systems* 2.1 (2017).

[246] Bob Sturm. “What do these 5,599,881 parameters mean?: An analysis of a specific LSTM music transcription model, starting with the 70,281 parameters of its softmax layer”. In: *International Conference on Computational Creativity*. 2018.

[247] Tian Cheng, Satoru Fukayama, and Masataka Goto. “Comparing RNN Parameters for Melodic Similarity.” In: *ISMIR*. 2018, pp. 763–770.

[248] Roger B. Dannenberg. “The Interpretation of MIDI Velocity”. In: *International Conference on Mathematics and Computing*. 2006.

[249] MICHAEL C. MOZER. “Neural Network Music Composition by Prediction: Exploring the Benefits of Psychoacoustic Constraints and Multiscale Processing”. In: *Connection Science* 6.2-3 (1994), pp. 247–280. DOI: 10.1080/09540099408915726.

[250] Felix A Gers and Jürgen Schmidhuber. “Recurrent nets that time and count”. In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*. Vol. 3. IEEE. 2000, pp. 189–194. DOI: 10.1109/IJCNN.2000.861302.

[251] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. “This time with feeling: Learning expressive musical performance”. In: *Neural Computing and Applications* 32.4 (2020), pp. 955–967. DOI: 10.1007/s00521-018-3758-9.

[252] Craig Stuart Sapp. “Online Database of Scores in the Humdrum File Format”. In: *ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, Proceedings*. Sept. 2005, pp. 664–665.

[253] Yarin Gal and Zoubin Ghahramani. “A Theoretically Grounded Application of Dropout in Recurrent Neural Networks”. In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*. NIPS’16. Barcelona, Spain: Curran Associates Inc., Dec. 2016, pp. 1019–1027. ISBN: 9781510838819.

[254] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. “Visualizing and understanding recurrent networks”. In: *arXiv preprint arXiv:1506.02078* (2015). DOI: https://doi.org/10.48550/arXiv.1506.02078.