The Harmonic Memory: a Knowledge Graph of harmonic patterns as a trustworthy framework for computational creativity

Jacopo de Berardinis  
Albert Meroño-Peñuela  
jacopo.deberardinis@kcl.ac.uk  
King’s College London  
London, UK

Andrea Poltronieri  
Valentina Presutti  
andrea.poltronieri2@unibo.it  
University of Bologna  
Bologna, Italy

ABSTRACT
Computationally creative systems for music have recently achieved impressive results, fuelled by progress in generative machine learning. However, black-box approaches have raised fundamental concerns for ethics, accountability, explainability, and musical plausibility. To enable trustworthy machine creativity, we introduce the Harmonic Memory, a Knowledge Graph (KG) of harmonic patterns extracted from a large and heterogeneous musical corpus. By leveraging a cognitive model of tonal harmony, chord progressions are segmented into meaningful structures, and patterns emerge from their comparison via harmonic similarity. Akin to a music memory, the KG holds temporal connections between consecutive patterns, as well as salient similarity relationships. After demonstrating the validity of our choices, we provide examples of how this design enables novel pathways for combinational creativity. The memory provides a fully accountable and explainable framework to inspire and support creative professionals – allowing for the discovery of progressions consistent with given criteria, the recomposition of harmonic sections, but also the co-creation of new progressions.

CCS CONCEPTS
- Applied computing → Sound and music computing;  
- Computing methodologies → Knowledge representation and reasoning;  
- Ontology engineering.

KEYWORDS
computational creativity, knowledge graphs, music technology

ACM Reference Format:

1 INTRODUCTION
Creativity has been defined as the ability to come up with new, surprising, and valuable ideas or artifacts [5]. These can be abstract concepts, scientific theories, solutions to real-world problems, but also new designs and artworks. In her seminal work, Boden categorised creativity into three types: (i) exploratory, where new ideas are generated by exploration of a space of concepts; (ii) combinational, which enables the creation of new ideas through the combination of familiar ones; and (iii) transformational, where the “rules” governing a space are challenged and transformed, to generate new kinds of ideas. A computational creativity theory was also formulated by Colton et al., to describe creative and generative acts (FACE model) and their potential for impact (IDEA model).

Attempts at formalising human creativity date back to the ancient Greeks, and remained up to and beyond Mozart with the “Dice Game” and Ada Lovelace – speculating that the “calculating engine” might compose elaborate and scientific pieces of music of any degree of complexity. Since then, creativity, creative reasoning and creative problem solving have been extensively researched in cognitive [4] and computational sciences [23]. A simple definition of a computationally creative system is that of a model capable to perform “generative acts” that create artefacts, concepts, or provide an aesthetic evaluation for the generated outputs [24, 40]. By harnessing recent advancements in machine learning, a variety of systems have already been implemented across several domains. Examples include computational systems for material discovery [9], molecular design [66], and more broadly, for virtual laboratories [36]; but also models for generating textual artefacts [52], images [58, 61], and even recipes [62] from a variety of prompts.

In the music domain, data-driven generative systems based on deep learning methods have achieved impressive results on symbolic music [7], and they can also produce realistic outputs when trained on the raw audio [21]. The variety of computationally creative methods for music is quite broad and diversified, and has already enabled the exploration of novel forms of artistic co-creation [33]. These range from the automatic generation, completion, and alteration of chord progressions and melodies, to the creation of mashups, and audio snippets from textual prompts [2]. Due to their success, some of these systems have already been integrated into commercial software, such as Aiva 1 and Amper 2 – allowing users to generate full music pieces based on their desiderata.

1https://www.aiva.ai  
2https://www.ampermusic.com

1.1 Fundamental concerns of music AI systems
Nonetheless, having a system that can fully generate realistic music raises ethical concerns – especially when those systems are made commercial and can potentially replace artists, rather than augmenting their possibilities [65]. Indeed, research can open highly
This fundamentally hampers a potential dialogue and synergies (e.g. autoregressive, masked prediction, generative modelling) are signal processing methods to segment the resulting sequences into project chord progressions into a musically meaningful space, and transparent, accountable, and musically plausible way.

Harmonic patterns aimed to support creative applications in a fully the Harmonic Memory potential of composers as a human-machine collaboration [11].

In addition, computationally creative models that fully learn music representations from the data by maximising a learning objective (e.g. autoregressive, masked prediction, generative modelling) are often criticised for lacking accountability, explainability, and musical plausibility. The former is related to the challenge of keeping track of where the model picks up while generating new musical content. As the model is unaware of its influences while composing, this may prevent giving recognition to real artists, which has direct implication on copyright and revenue sharing [22]. Similarly, the lack of explainability represents a technological barrier for users, as there is little or no understanding of the creative process underneath. Explainability is a desirable component for computationally creative systems, as it facilitates the interaction with artists, and particularly, the ability to control/steer the system based on domain knowledge [8, 6]. Finally, the “creative space” learned by data-driven systems is often criticised by musicologists and music experts in regard to musical plausibility [30], meaning that, solutions generated from these models may violate common notions of music theory. This fundamentally hampers a potential dialogue and synergies between music experts and AI researchers.

In sum, most music AI systems cannot yet be deemed trustworthy by design (accountability, explainability, ethics, etc.) [26], which raises serious concerns related to their large scale adoption.

1.2 Our contribution
Instead of replacing artists, we believe that research should focus on leveraging the generative capabilities of these systems to design novel solutions that can support, enhance and augment the creative potential of composers as a human-machine collaboration [11].

Inspired by research in music psychology [37], here we present the Harmonic Memory (Harmory) – a Knowledge Graph (KG) of harmonic patterns aimed to support creative applications in a fully transparent, accountable, and musically plausible way. Underpinning Harmory, we also contribute and validate empirically: (i) a novel method for harmonic structure analysis in the symbolic domain, that leverages cognitive and musical models of tonal harmony; (ii) and a state of the art algorithm for harmonic similarity. Examples of possible applications for trustworthy machine creativity implemented on top of Harmory, focusing on knowledge discovery and human-machine chord generation.

2 RELATED WORK
To the best of our knowledge, most machine learning systems are explorative. Starting from different prompts, such as a priming music to continue, an incomplete passage, or a textual query, these models can generate convincing outputs by sampling from the learned distribution. These include methods based on recurrent [64], self-attention [34], and convolutional neural networks [32]. Instead, current combinational systems are dominated by variational autoencoders, which can create new ideas by interpolating between two musical passages in a latent space [60]. Transformative approaches for music have been implemented by “hacking” the former methods based on the idea of brain transplant, to provide
additional artistic stimulation [12]. These range from gentler interventions mixing up corpora, to splicing neural networks, jointly training with interference, and Frankensteinian hybrid models [67].

As pointed out before, most of these works lack trustworthy features to support and protect creative professionals. Recently, Explainable Computational Creativity (XCC) systems have been proposed, to promote a bidirectional interaction between system and user [44]. This interaction is communicative, enabling the exchange of decisions and ideas in a format that can be understood by both humans and machines. Examples of explainable systems also include [14] – presenting a real-time human-machine interaction for artwork creation: the system provides explanations for its decisions, while users can guide the creative process.

Semantic Web technologies have also been used to make creative systems more explainable. An example is [55], which proposes a system for creating innovative food combinations using a knowledge graph that describes compounds and ingredients. However, to the best of our knowledge, no such systems have been proposed in the musical domain. A notable exception is the work by Meerwaldt et al., enabling the generation of mashups by leveraging Semantic Web technologies for machine creativity [40]. Our work differs substantially in the broader intent and creative applications it enables, the musicological and cognitive basis, the scope/granularity of the interconnected musical content (patterns vs full pieces).

3 HARMORY: THE HARMONIC MEMORY

The main steps for the creation of Harmony are illustrated in Figure 1, and encompass four stages: (i) projection of harmonic sequences in the Tonal Pitch Space; (ii) novelty-based segmentation of harmonic sequences; (iii) pattern identification through similarity-based linking of harmonic segments; and (iii) KG creation.

Our workflow is defined from the harmonic analysis of a piece, which contains a sequence of chords in Harte notation [31], their onsets, and the associated local keys. Formally, let $c = \{c_1, \ldots, c_N\}$ denote a chord sequence of length $N$, where each chord figure $c_i$ is drawn from the Harte chordal set $H$. Similarly, $k = \{k_1, \ldots, k_N\}$ denotes the corresponding local keys of $c$, s.t. each $k_i$ is a tonic-mode tuple defined from $\mathbb{T} \times \mathbb{M}$, where $\mathbb{T} = \{Ab, A, A#, Bb, \ldots, G\}$ is the set of all possible tonic notes, and $\mathbb{M} = \{major, dorian, \ldots, locrian\}$ is the set of all possible modes in Western tonal music.

For simplicity, chords are expected to be temporally aligned with their onsets, meaning that $c_i$ ends when $c_{i+1}$ starts, $\forall i \in N - 1$. Hence, onsets are defined as a $(N + 1)$-th dimensional vector $t \in \mathbb{N}^{N+1}$ to compensate for the end time of the last chord ($t_{N+1}$ is the end of $c_N$). Onsets are given in seconds for harmonic analyses on audio music; or as global beats for symbolic music. For example, $c = [G, B, min, E, min7, \ldots, ]$, $k = [\{G, major\}, \{G, major\}, \{G, major\}, \ldots]$, and $t = [1, 3, 5, \ldots]$ are the first three occurrences of such vectors for a "A Day in the Life" by The Beatles.

3.1 Encoding chords in the Tonal Pitch Space

Given a harmonic analysis $H = \{c, k, t\}$, the first step is to encode $c$ and $k$ as a numerical stream, so as to allow the processing of similarity/distance operations. This is necessary because chords (e) and tonalities (k) are complex elements to process, and come in symbolic format. More specifically, a chord label is a convention for describing intervals built on a root note. For example, the label of a C major seventh chord (C:maj7) represents the intervals of a major quadriad with a minor seventh built on the note C, which is equivalent to the note set $\{C, E, G, Bb\}$. Also, the harmonic function of a chord is contextual to the global (and local) key [1].

One option here is to leverage Representation Learning methods on symbolic music to learn harmonic embeddings from a large corpus of chord sequences [39, 41]. These include static embedding methods, such as Word2Vec [47] and Glove [53], as well as sequence models for contextualized representations, such as ELMo [54] and BERT [20] – which have proved their efficacy on a variety of natural language processing tasks. Nonetheless, in the music domain, representation learning methods have recently started to gain success for audio music [35], whereas little attention has been given to symbolic music. This is exacerbated by the challenge of finding musicological interpretability of the resulting embeddings, requiring new probing and evaluation methods for music [30].

We aim for an encoding of harmony that is well established, perceptually and musicologically plausible, and explainable by design. Hence, we rely on the Tonal Pitch Space [42] – a cognitive model of tonality used in music psychology and computational musicology.

The tonal pitch space

The Tonal Pitch Space (TPS) model [42] provides a scoring mechanism that predicts the proximity between musical chords. It is based on the Generative Theory of Tonal Music [43] and designed to make explicit music theoretical and cognitive intuitions about tonal organisation. The model works by comparing any possible chord to an arbitrary key, by means of the basic space. The basic space is constituted by five different levels, ordered from the most stable to the least stable: (i) the Root level; (ii) the Fifths level; (iii) the Triadic level; (iv) the Diatonic level; and (v) the Chromatic level.

Each level holds one or more notes, indexed from 0 (C) to 11 (B). The Root level holds the root of a chord ($0$ for C-major), while the Fifths level adds the fifth ($0$, 7 for C-major). The Triadic level has all the notes in the chord ($0$, 4, 7 for C major). The Diatonic level depends on the chord’s key as it holds all the notes of the diatonic scale of the key ($0$, 2, 4, 5, 7, 9, 11 for the C major key). Finally, the Chromatic level holds the chromatic scale ($0$-$11$).

The distance between two chords $c_i, c_j$ in keys $k_i, k_j$ is calculated using the basic spaces of the chords. The basic space is set to match the key of the pieces (level $iv$), and their levels (i-$iii$) are adapted to match the chords to be compared. The Chord distance rule is applied to calculate the distance. The Chord distance rule is defined as $d(x, y) = j + k$, where $d(x, y)$ is the distance between chord $x$ and chord $y$; $j$ is the minimum number of Circle-of-Fifths rule applications to shift $x$ into $y$, and $k$ is the number of non-common pitch classes divided by 2 in the levels (i-$iv$) of the basic spaces of $x$ and $y$. The Circle-of-Fifths rule consists in moving the levels (i-$iii$) four steps to the right or left on level $iv$.

For each comparison between two chords, the TPS returns a value in $[0, 13]$. TPS has been demonstrated to be sound both musically and perceptually [18, 19], and in this work, it is used to encode and compare chord-key pairs.
3.2 Novelty-based harmonic segmentation

The projection of chord-key pairs \((c_i, k_i)\) in the TPS is a fundamental requirement to perform harmonic segmentation. First, the given harmonic annotation \(H\) is used to sample a signal \(X\) of length \(d = t_{N+1} \cdot f_s\), where harmonic observation \((c_i, k_i)\) is consecutively repeated \(t_i \cdot f_s\) times (its duration), according to a sampling rate \(f_s\). Each element \(x_i \in X\) now encodes an input for the TPS model, containing the harmonic content at the \(i\)-th sample.

The resulting signal allows for the computation of two harmonic descriptors, i.e., the Harmonic Profile (or TPS time series), and the Harmonic self-similarity matrix (SSM) – the entry point for segmentation. The former is defined as a vector \(q \in \mathbb{R}^d\) s.t. \(q_i = tps(x_i, k_i)\), holding the TPS distance between each harmonic observation \(x_i\) and the global key \(k_i\) of the piece (assumed as the first key occurrence). Similarly, the Harmonic SSM is a matrix \(S \in \mathbb{R}^{d \times d}\) s.t.

\[
S(n, m) = 1 - \frac{tps(x_n, x_m)}{13},
\]

where \(x_i \in X\) is a column vector; \(n, m \in [0 : d - 1]\); 13 is a normalisation factor (the maximum TPS value); and the subtraction from 1 is used to obtain a similarity score from a distance measure.

Self-similarity matrices have been extensively used for structure analysis, due to their ability to reveal nested structural elements [27, 17]. As can be seen from Figure 2, block-like structures are observed when the underlying sequence shows homogeneous features over the duration of the corresponding segment. Often, such a homogeneous segment is followed by another homogeneous segment that stands in contrast to the previous one.

To identify the boundary between two homogeneous but contrasting segments (2D corner points), we slide a checkerboard kernel \(K\) along the main diagonal of the SSM and sum up the element-wise product of \(K\) and \(S\). A checkerboard kernel can be simply defined as a box kernel \(K_B \in \mathbb{Z}^{M \times M}\) where \(M = 2L + 1\) is the size of the kernel, defined as \(K_B = sgn(k) \cdot sgn(l)\) \(\forall k, l \in [-L, L]\), where \(sgn\) is the sign function. This yields a novelty function \(\Delta_{kernel}(n)\) for each index \(n \in [1 : d]\) of \(X\) as follows:

\[
\Delta_{kernel}(n) = \sum_{k,l \in [-L,L]} K(k, l) \cdot S(n + k, n + l)
\]

for \(n \in [L + 1 : d - L]\). When \(K\) is located within a relatively uniform region of \(S\), the positive and negative values of the product tend to sum to zero (small novelty). Conversely, when \(K\) is at the crux of a checkerboard-like structure of \(S\), the values of the product are all positive and sum up to a large value (high novelty) [50].

Local maxima of the novelty curve are then used to detect the boundaries of neighbouring segments that correspond to contrasting harmonic parts. For this, we use a pick peaking method that applies a smoothing filter to the novelty function (to reduce the effect of noise-like fluctuations) and uses adaptive thresholding to select a peak when novelty exceeds a local average [51]. The detected segment boundaries are used to split \(X\) and the corresponding \(q\) into a number of non-overlapping harmonic structures. This yields \(\bar{q} = \bar{q}^1, \ldots, \bar{q}^P\), where \(P\) denotes the number of structures.

3.3 Linking harmonic segments via similarity

Each harmonic structure \(\bar{q}^i\) is then considered for harmonic similarity. Since \(\bar{q}^i\) is still a time series (a partition of \(q\), the Harmonic Profile), we formulate the harmonic similarity between two harmonic structures by comparing their time series. This is done using Dynamic Time Warping (DTW) – an algorithm for comparing time series, which has been widely used across various domains, including speech recognition [49], pattern recognition [63], and bioinformatics [71]. In our case, DTW has desirable properties, as it is invariant to time shifts, and robust to local variations.

Vanilla DTW compares two time series by calculating the cumulative distances between each point/observation. It allows for non-linear alignment between the time series by considering the local warping path. The cost matrix, holding the cumulative distance between each corresponding point, is constructed using the Euclidean distance, and is formalised as:

\[
d_{DTW}(\bar{q}^i, \bar{p}^j) = \sqrt{\sum_{(v,w) \in \pi} ||\bar{q}_{v} - \bar{p}_{w}||^2}
\]

where \(\pi\) is the optimal warping path – the shortest cumulative distance between the time series (found via dynamic programming).

As the computational complexity of vanilla DTW is quadratic in the sequence length, here we use the Sakoe-Chiba variant. The latter achieves linear complexity \(O(N \cdot w)\), by constraining the warping path within a window of size \(w\), rather than using all points (\(N\)).

Prior to the computation of similarities, time series are normalised are resampled to meet the same length, and standardised to zero mean and unit variance. This has the effect of comparing time series by looking at their shapes in an amplitude-invariant manner – which brings us closer to the identification of harmonic patterns.

The latter emerge after retrieving the \(k\) most similar structures for each segment \(\bar{q}^i\), and applying a filtration to retain only those...
structures $p^i$ whose $d_{DTW}(q^i, p^i)$ is below a given threshold. Structures sharing the same (normalised) TPS time series ($d_{DTW} = 0$) define a distinct harmonic pattern; whereas segments with similar time series can be grouped within the same pattern family/cluster.

### 3.4 Knowledge graph creation

An ontology, called Harmory Ontology, was developed for the creation of the Knowledge Graph (KG). The ontology re-uses the Core module from the Polifonia Ontology Network (PON) [10], a network of ontologies for the semantic annotation of musical content. This allows to link Harmory to ChoCo$^3$ [16]. We also align to the Music Ontology [59] – a widely used ontology model in the music domain. The ontology was realised by applying the eXtreme Design methodology [56], which relies on Ontology Design Patterns (ODP) and on the iterative testing of the produced model against a predefined set of Competency Questions (CQs).

For each piece, the ontology allows to: (i) store its metadata, such as title, genre, and artist; (ii) hold the harmonic segmentation (see Section 3.2); and (iii) relate similar segments (see Section 3.3). This enables semantic access to the aforementioned data via SPARQL.

The model is illustrated in Figure 3, using the Grafoo notation$^4$. A piece of music is expressed by means of the class har:MusicalWork, which is aligned with mo:MusicalWork in the Music Ontology. The metadata of a work is stored via core:hasTitle, core:hasGenre, and core:hasArtist, which describe the title, musical genre and composer or performer of the piece, respectively.

A musical work has a har:SegmentationSituation – a specialisation of the Situation Pattern [28] describing a segmentation performed by a specific har:SegmentationAlgorithm that produces one or more has:Segments. In this context, a harmonic sequence is split/partitioned into a number of segments, with their ordering allowing for reconstruction. Each sequence also holds its chords, using the class mf:Chord. Each segment is linked to its corresponding har:SegmentPattern – an abstraction of the TPS pattern normalised on the temporal axis. Hence, several har:Patterns may have the same har:SegmentPattern. Similarity relations are expressed via the class har:SegmentPatternSimilarity, which relates two Segment Patterns and holds their similarity value via the datatype property har:hasSimilarityValue.

### 4 EXPERIMENTS

To validate Harmory, we tested the efficacy of the two central components underpinning its creation: the DTW harmonic similarity (Section 3.3), and the harmonic segmentation (Section 3.2).

#### 4.1 Evaluation of harmonic similarity

We evaluated the DTW harmonic similarity by comparing our implementation with other algorithms for the cover song detection task – a common benchmark for similarity algorithms in the symbolic music domain [18, 19].

In this comparison, performance is evaluated using two standard metrics: First Tier and Second Tier. The former measures the ratio of correctly retrieved songs within the top $(C_t - 1)$ matches to $(C_t - 1)$, where $C_t$ is the size of the song class (e.g. the same composition, or performance) for track $t$. The First Tier can be formalised as:

$$
FirstTier(D) = \frac{\sum_{t=0}^{N} \frac{|L_t(C_t - 1)| \cap C_t|}{||C_t - 1||}}{N}
$$

where $N$ is the set of all tracks in the dataset having at least a "cover", and $L_t(C_t - 1)$ denotes the list of matches for track $t$ ranked by similarity – where only the first $(C_t - 1)$ occurrences are considered. Similarly, the Second Tier is defined as the ratio of correctly retrieved songs within the best $(2C_t - 1)$ matches to $(C_t - 1)$.

$$
SecondTier(D) = \frac{\sum_{t=0}^{N} \frac{|L_t(2C_t - 1)| \cap C_t|}{||C_t - 1||}}{N}
$$

$^3$ChoCo SPARQL endpoint: https://polifonia.disi.unibo.it/choco/sparql

$^4$Grafoo Notation: https://essepuntato.it/graffoo/
Table 1: Performance of similarity algorithms on cover song detection. The highlighted lines denote the best performing algorithms, while results in bold indicate the best performance obtained for the First Tier and Second Tier, respectively.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>TPS Mode</th>
<th>Stretch</th>
<th>Constraint</th>
<th>Normalise</th>
<th>First Tier</th>
<th>Second Tier</th>
<th>First Tier</th>
<th>Second Tier</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPSD</td>
<td>offset</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.49</td>
<td>0.63</td>
<td>0.62</td>
<td>0.68</td>
</tr>
<tr>
<td>TPSD</td>
<td>profile</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.53</td>
<td>0.74</td>
<td>0.76</td>
<td>0.83</td>
</tr>
<tr>
<td>DTW</td>
<td>offset</td>
<td>stretch</td>
<td>-</td>
<td>-</td>
<td>0.94</td>
<td>0.98</td>
<td>0.53</td>
<td>0.67</td>
</tr>
<tr>
<td>DTW</td>
<td>profile</td>
<td>stretch</td>
<td>-</td>
<td>-</td>
<td>0.97</td>
<td>0.99</td>
<td>0.6</td>
<td>0.69</td>
</tr>
<tr>
<td>DTW</td>
<td>offset</td>
<td>stretch</td>
<td>sakoe_chiba</td>
<td>-</td>
<td>0.96</td>
<td>0.99</td>
<td>0.62</td>
<td>0.7</td>
</tr>
<tr>
<td>DTW</td>
<td>profile</td>
<td>stretch</td>
<td>sakoe_chiba</td>
<td>-</td>
<td>0.97</td>
<td>0.99</td>
<td>0.69</td>
<td>0.77</td>
</tr>
<tr>
<td>DTW</td>
<td>offset</td>
<td>stretch</td>
<td>itakura</td>
<td>-</td>
<td>0.96</td>
<td>0.99</td>
<td>0.55</td>
<td>0.65</td>
</tr>
<tr>
<td>DTW</td>
<td>profile</td>
<td>stretch</td>
<td>sakoe_chiba</td>
<td>yes</td>
<td>0.97</td>
<td>0.99</td>
<td>0.7</td>
<td>0.76</td>
</tr>
<tr>
<td>LCSS</td>
<td>offset</td>
<td>-</td>
<td>sakoe_chiba</td>
<td>-</td>
<td>0.38</td>
<td>0.61</td>
<td>0.03</td>
<td>0.07</td>
</tr>
<tr>
<td>LCSS</td>
<td>offset</td>
<td>-</td>
<td>itakura</td>
<td>-</td>
<td>0.7</td>
<td>0.8</td>
<td>0.14</td>
<td>0.23</td>
</tr>
<tr>
<td>SoftDTW</td>
<td>offset</td>
<td>stretch</td>
<td>-</td>
<td>-</td>
<td>0.93</td>
<td>0.97</td>
<td>0.55</td>
<td>0.69</td>
</tr>
<tr>
<td>SoftDTW</td>
<td>profile</td>
<td>stretch</td>
<td>sakoe_chiba</td>
<td>-</td>
<td>0.98</td>
<td>0.99</td>
<td>0.62</td>
<td>0.73</td>
</tr>
</tbody>
</table>

Methods. We compare our DTW similarity (c.f. Section 3.3) with the following algorithms for harmonic and time series similarity:

- **Tonal Pitch Step Distance (TPSD)** [18, 19], a state of the art method that measures the difference between the Harmonic Profiles (see \( q \) in Section 3.2) of the given harmonic sequences. The difference is determined as the minimal area between the two time series, after considering all possible horizontal shifts. TPSD can handle sequences of different length, and has a time complexity of \( O(nm \log (n + m)) \), where \( n \) and \( m \) denote the length of the compared chord sequences [3]:

- **Longest Common Subsequence (LCSS)** [68], a method expressing time series similarity based on their longest common subsequence. Similarity is calculated as the relative length of the longest common subsequence compared to the length of the shortest time series, thus ranging in \([0, 1]\). Using dynamic programming, LCSS is bounded in \( O(n^2) \);

- **Soft Dynamic Time Warping (Soft DTW)** [15], a variant of DTW that allows for non-binary (fuzzy) alignments between time series, by using a sof-constraint. Soft DTW can be computed with quadratic time/space complexity.

All experiments are performed on the Harmonic Profile, in addition to an alternative formulation of the TPS time series, called offsets, where \( q_i = tps(x_i, x_{i-1}) \) (chord offset distance).

For DTW, LCSS and Soft DTW, two types of constraints were also tested: Sakoe-Chiba and Itakura. Analogously to Sakoe-Chiba, the Itakura constraint sets a maximum distance for each point in the time series, making the algorithm more efficient, and reducing the risk of being trapped in local minima. Several parameter settings for the Sakoe-Chiba radius and Itakura band were tested, and the best results were obtained by setting them to 5 and 4, respectively. This parametrisation turned out to be optimal across all our experiments.

Each method was tested on sequences of original length (no-stretch) and after resampling to the shortest sequence. We also experimented with normalised time series (Section 3.3).

Dataset. We two subsets of ChoCo [16] containing cover tracks: Schubert Winterreise [69] and Chordify Annotator Subjectivity Dataset (CASD) [38]. The former provides harmonic annotations for each of the 9 different performances of the same musical piece by Schubert. Similarly, CASD contains four annotations of the same performance, contributed by four different annotators. Chords from Isophonics Dataset [45] and Jazz Audio-Aligned Harmony (JAAH) [25] are also added to the reference dataset in order to add heterogeneity (different genres) and increase the complexity of the task.

Results. Table 1 shows the results of this comparison and highlights the best performing algorithms. Results are presented for Schubert and CASD separately, and also in a third merged setup (Schubert+CASD). Notably, the performance of the DTW algorithm is significantly better for Schubert (one piece, multiple performances), while for CASD (one performance, multiple annotations), TPSD performs slightly better. The best results for the third setup are obtained using the Sakoe-Chiba DTW, using normalisation and resampling on the shortest sequence. It is also worth remarking that our implementation, besides being the most accurate overall, is also the most efficient approach, due to its linear complexity.

4.2 Structural coverage of known patterns

To validate our harmonic segmentation (Section 3.2), we measure the overlap between the resulting structures with a collection of well-known chordal patterns. This exemplifies the hypothesis that a good segmentation would maximise the “reuse” of harmonic patterns – as building blocks that can be found in other pieces.

Given a segmentation \( q = q_1^{}, \ldots, q_p^{} \) of a piece, with each \( q_i^{} \) containing a TPS time series, the overlap of \( q \) with a dataset of known harmonic patterns \( P \) is computed as:

\[
\sigma_M(q) = \frac{1}{T} \sum_{i=1}^{T} \min_{p \in P} d_{DTW}(q_i^{}, p), \quad (6)
\]

\[
\sigma_B(q) = \min_{q' \in P} \min_{p \in P} d_{DTW}(q'_i^{}, p), \quad (7)
\]

which differ in the aggregation function. The former measures the average pattern distance contributed by each structure in the segmentation; the latter, instead, only retains the distance of the most similar pattern that was matched to one of the structures. When \( \sigma_M = 0 \), all segments are fully matched/found in \( P \); whereas \( \sigma_M \) is minimal when at least a segment matches a pattern in \( P \).

Methods. We compare our method (denoted as harmov) to fast low-cost unipotent semantic segmentation (FLUSS) [29] – a state of the art algorithm for time series segmentation defined on the Matrix.
We envisage various applications of Harmory across different tasks and use cases, ranging from music information retrieval and computational musicology, to creativity support for artistic workflows. The latter is the main focus of this work. However, we do not aim at improving the state of the art in music generation, but rather to provide a transparent system to support creative workflows [11].

Here, we show examples of trustworthy applications for pattern discovery, human-machine chord generation, and harmonic similarity. The latter is more of musicological interest, whereas the former are both creative use cases. Each application is demonstrated through a set of Prompts, expressed in natural language, which correspond to SPARQL query templates to interrogate the KG (Section 3). The latter are fully available on our repository6.

5.1 Pattern discovery

The traversal of the Harmonic Memory makes it possible to obtain granular information of the harmonic structure of songs. In particular, it is possible to explore the harmonic segments of each song, the patterns related to each segment, and the similarities with other patterns/segments found in other pieces.

A composer may start with a harmonic pattern mind, and initiate a creative exploration of the KG by leveraging music and metadata.

Prompts

Prompt 1 For a given pattern, which are the tracks (titles, artists and genres) in which the pattern can be found?

Prompt 2 Given a music genre, what are the most frequent patterns?

To support creative exploration, more complex prompts can be formulated in order to narrow down the search, and eventually discover surprising or unexpected outputs, if present.

Prompt 3 Which harmonic patterns are used in “Michelle” by The Beatles, but also in a classical composition?

Prompt 4 Which patterns used by The Beatles in “Michelle” but not in “Hey Jude” contain at least a B flat major seventh chord?

In the Harmory KG, we have included known patterns (as described in Section 4.2), which are labelled in such a way as to indicate their origin, mood, or harmonic function within the progression. These labelled harmonic fragments can be used as input for a query, e.g. for searching songs that contain them:

Prompt 5 Which tracks include a dominant cycle in seven steps?

Profile [70]. FLUSS annotates the time series with information about the likelihood of a regime change (a segment boundary); and is parameterised by a fixed window size $m$, and the number of segments to detect $r$. We also include a baseline splitting a time series in $r$ uniform segments. Both methods operate on the TPS Profile of $h$, and are optimised via grid search.

Datasets. We compute and evaluate the harmonic segmentations on a dataset comprising 320 chord progressions, obtained from randomly sampling 40 pieces per audio partition in ChoCo (isophonics, billboard, casd, schubert-winterreise, rwc-pop, uspop-2002, jaah, robbie-williams). This yields a diversified (several genres, durations, etc.) yet representative sample of Harmory (≈ 2% of ChoCo); which prevents larger partitions from biasing the overall results. For $\mathcal{P}$, we assembled a dataset of known harmonic patterns from Impro-Visor [57], which is available on GitHub5. After filtration of trivial occurrences (e.g. chord uni-grams, sequences with repeated chord occurrences, etc.), the dataset counts 300 unique patterns spanning from 3 to 10 chord occurrences per pattern (the length of a chordal pattern).

Results. The structural coverage, computed for each segmentation method and aggregated for all known harmonic patterns of the same length, is reported in Figure 4. For both measures $e_M$, $e_B$, the segmentations produced by our method (harmov) produce the lowest distances – meaning that they show the highest overlap with the known harmonic patterns in $\mathcal{P}$. This behaviour is preserved for all pattern groups (the x-axis), and the gap with the other methods increases with pattern’s length. The second performing method is FLUSS, using $r = 14$ split regions and a window size of $m = 3$. However, for longer patterns, the latter performs comparably with a fixed sequence split (the other baseline). Finally, it is worth remarking that results for all baselines are first optimised on a grid search; whereas we use the default parameterisation for harmov.

5 AVENUES FOR MACHINE CREATIVITY

We envisage various applications of Harmory across different tasks and use cases, ranging from music information retrieval and computational musicology, to creativity support for artistic workflows. The latter is the main focus of this work. However, we do not aim at improving the state of the art in music generation, but rather to provide a transparent system to support creative workflows [11].

Here, we show examples of trustworthy applications for pattern discovery, human-machine chord generation, and harmonic similarity. The latter is more of musicological interest, whereas the former are both creative use cases. Each application is demonstrated through a set of Prompts, expressed in natural language, which correspond to SPARQL query templates to interrogate the KG (Section 3). The latter are fully available on our repository6.

5.1 Pattern discovery

The traversal of the Harmonic Memory makes it possible to obtain granular information of the harmonic structure of songs. In particular, it is possible to explore the harmonic segments of each song, the patterns related to each segment, and the similarities with other patterns/segments found in other pieces.

A composer may start with a harmonic pattern mind, and initiate a creative exploration of the KG by leveraging music and metadata.

Prompts

Prompt 1 For a given pattern, which are the tracks (titles, artists and genres) in which the pattern can be found?

Prompt 2 Given a music genre, what are the most frequent patterns?

To support creative exploration, more complex prompts can be formulated in order to narrow down the search, and eventually discover surprising or unexpected outputs, if present.

Prompt 3 Which harmonic patterns are used in “Michelle” by The Beatles, but also in a classical composition?

Prompt 4 Which patterns used by The Beatles in “Michelle” but not in “Hey Jude” contain at least a B flat major seventh chord?

In the Harmory KG, we have included known patterns (as described in Section 4.2), which are labelled in such a way as to indicate their origin, mood, or harmonic function within the progression. These labelled harmonic fragments can be used as input for a query, e.g. for searching songs that contain them:

Prompt 5 Which tracks include a dominant cycle in seven steps?

5https://github.com/Impro-Visor/Impro-Visor

6SPARQL queries: https://github.com/smashub/harmory/tree/main/queries
5.2 Human-machine chord generation

Harmory also enables combinational creativity use cases. New progressions are generated by moving across patterns through temporal and similarity links, based on the given creative requirements. At generation time, this has the advantage of giving recognition to all artists that contributed to the new creation, as shown in Figure 5.

First, it can provide statistical information regarding variations of a given harmonic sequence. As these variations come from real pieces, it is also possible to leverage metadata for controlling the generation. To do this, a prompt can be formulated from a given (possibly new) harmonic sequence (or a part of it), to retrieve all the all harmonic sequences using the same pattern.

Prompt 6 Given a chord sequence, which are its variations, and which tracks these variations belong to?

Similarly, it is also possible to query the most similar (or most distant) harmonic sequences to a given one:

Prompt 7 Given a chord sequence, which are its most similar chord sequences, sorted by similarity?

These simple constructs already allow to generate new harmonic sequences, starting from either a known harmonic idea/pattern, or a full progression. If starting from a full progression, one way is to identify the first harmonic segment that makes it up. From this point, transitions can be made using similarity relations, while taking into account the order of the different segments (temporal connections) and their tonality. For example, starting from the first harmonic segment of a song (a priming sequence), one can then generate a continuation by identifying similar sequences to the next sequence, filtering them by tonality (or/and by artist, genre, title) and repeat this process recursively for a number of steps, criteria, or with the supervision/control of the user.

Prompt 8 Create a progression starting with “Michelle” by The Beatles, continuing with a segment found in a classical piece of music, and then continuing with another by Chet Baker.

5.3 Harmonic similarity

From a musico-logical perspective, the KG can also be used to analyse similarity relations between tracks – by leveraging the local information relating harmonic structures. This also allows for the formal definition of similarity functions (depending on a genre- or task-specific notions) by using logical operators (SPARQL syntax) over harmonic segments/patterns. An example is given below.

Prompt 9 Given a track, which tracks contain patterns with a distance of less than 0.2, each having the same order?

As expected, the results of this query are almost exclusively cover songs of the given track. Nevertheless, a similarity function can be defined to be less strict, and hence more explorative. For instance, the similarity function below uses a higher similarity threshold for patterns, and does not constrain on the order of segments.

Prompt 10 Given a track, which tracks contain patterns with a distance of less than 0.5, regardless of their order?

6 CONCLUSIONS

Our work contributes a Web resource aimed to support the design of trustworthy systems for computational music creativity. This is a central requirement for the large scale adoption of these systems, which is often neglected in generative machine learning research. To this vision, we leverage a corpus of harmonic annotations on the Web, to design the Harmonic Memory (Harmory) – a Knowledge Graph of interconnected chordal patterns which is perceptually and musico-logically grounded. After demonstrating the validity of our framework, we showed how Harmony can enable transparent, explainable, and accountable applications for human-machine creativity – ranging from pattern discovery and chord generation, to harmonic similarity. Future work includes linking Web resources with musical pieces in Harmony, to achieve the alignment to other ontologies and Knowledge Graphs in the music domain. We also envisage the inclusion of heterogeneous data to enrich and complement the harmonic information, such as perceptual metadata, musical content (melodies), and complexity measures.

ACKNOWLEDGMENTS

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 101004746.


