Bridging the Gap: Demonstrating the Applicability of Linguistic Analysis Tools in Digital Musicology

Sebastian Oliver Eck
Department of Musicology Weimar-Jena
University of Music Franz Liszt Weimar, Germany
sebastian.eck@hfm.uni-weimar.de

Abstract

This study introduces the novel concepts of Explicit and Implicit Musical Parameters (EMPs and IMPs) and demonstrates their application in digital musicology. Furthermore, it discusses the concept of 'musical words', that suggests representing explicit and implicit musical parameters as words or textual entities. This 'music-to-text' approach allows the application of advanced techniques and tools commonly used within the computational linguistics for the analysis of musical data, highlighting the structural parallels between music and language. Lastly, the findings of this paper not only illustrate the feasibility of this approach but also pave the way for further interdisciplinary studies and the advancement of analytical user-friendly tools that are applicable in both computational linguistics and digital musicology.

1 Introduction

In the age of digital humanities, interdisciplinary dialogue between interrelated scientific fields is becoming increasingly important. With the intentions of showing the possible benefits of interdisciplinary approaches, this paper investigates to what extent easy-to-use off-the-shelf software-tools and methods commonly used within the computational linguistics (CL) can be adopted and applied by *digital musicologists* for their own research within their respective field.

An initial assessment of the relevant scientific literature underscores the urgency and relevance of such an exploration: though the potential merits (as well the presumed deficiencies) of using computational methods in musicology have already been recognised and emphasized decades ago (Volk et al., 2011; Cook, 2005; Huron, 1999), data available on lens.org, a repository of worldwide patent and academic knowledge, reveals a significant and widening gap between the number of publications in *digital musicology* and those in *computational linguistics* (cf. Figure 1). It is reasonable to assume that this discrepancy might not only persist but potentially grow wider, as with the growing public interest and economic relevance of natural language processing and generation technologies, such as the now famous ChatGPT, computational linguistics is likely to gain even more resources, both in academia as well as the free market. However, the significance of this research paper lies in its potential to create a bridge between two seemingly disparate fields: successfully applying CL tools within the field of digital musicology could not only expand research possibilities drastically, but also facilitate a more holistic approach when researching and trying to understand the similarities between language and music.

This study is structured into four main parts: an introduction that discusses structural parallels between music and language, further introducing the novel idea of implicit and explicit musical parameters (EMPs and IMPs); a data section describing the data preparation process, encompassing the collection, tokenization and transformation of data used for this study; a methodology section, in which will be explained, how GUI-applications frequently used in computational linguistics, such as AntConc, can be adapted to analyse textual music data; and finally, a demonstration section, in which methods are presented that showcase how AntConc could be employed to address various musicological questions.

1.1 Related Work

This study positions itself in the broad field of digital musicology by explicitly discussing the concept of *musical words* to which computational linguistics (CL) tools can be applied. It establishes connections between music and language, exploring

Proceedings of the Joint 3rd NLP4DH and 8th IWCLUL, pages 248–265

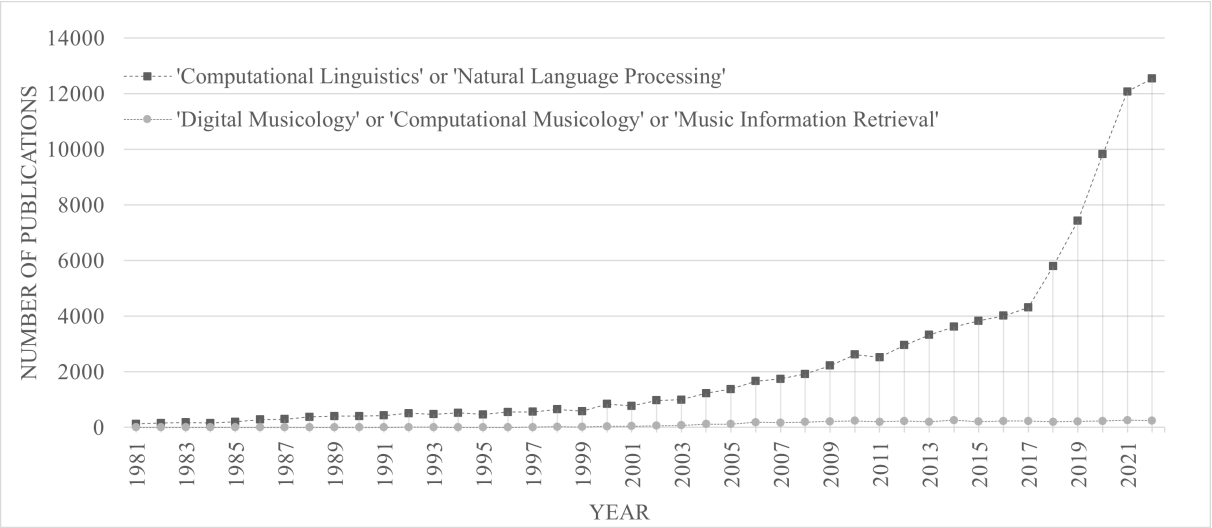


Figure 1: Scientific Publication Trends Measured by Keyword Matches in Title, Abstract, Keyword, or Field of Study (Data Source: lens.org)

their shared structural characteristics, in order to integrate CL tools into musicological research. As the work identifies a lack in studies in the application of CL tools to music data, it highlights opportunities for interdisciplinary research. This paper's findings, in particular the presented *music-to-text approach* further supports the foundational framework, set out by others (Norgaard and Römer, 2022; Wołkowicz et al., 2008), for reinterpreting monophonic musical data, e.g., folk songs, as music text, structurally comparable to written language. This paper differentiates itself by addressing theoretical limitations and philosophical interpretations of musical notation and performance, particularly through introducing the novel concepts of Explicit Musical Parameters (EMPs) and Implicit Musical Parameters (IMPs). On a final note, this study aims to further lower the barriers for interdisciplinary research by demonstrating the applicability of user-friendly GUI computational linguistics tools for musicological research.

Despite its rather low absolute quantity of publications, over the last decades research in computational musicology or, more specifically, Music Information Retrieval (MIR) has seen significant contributions in various areas. This includes work in authorship and composer recognition (Hontanilla et al., 2013; Kaliakatsos-Papakostas et al., 2010; Van Kranenburg and Backer, 2005), as well as in artist similarity recognition (Shao et al., 2008), genre recognition (Mayer and Rauber, 2011), or music recognition using 'acoustic fingerprinting' as an example (Brinkman et al., 2016).

Monophonic folk music classification, with its relatively straightforward structure, has consistently been a focal point in Music Information Retrieval (MIR) research (Huron et al., 1996). This trend continues in more recent studies (Hillewaere et al., 2014, 2009a; Taminau et al., 2009). In contrast, polyphonic music information retrieval, due to its inherent complexities, proofs to be more challenging, as evidenced in various conducted studies (Hillewaere et al., 2010, 2009b).

Additionally, MIR has made significant progress in the context of chord embeddings (Lahnala et al., 2021) and the development of chord vector representations (Madjiheurem et al., 2016).

Lastly, MIR's advancements have also extended beyond music, incorporating techniques that might seem more conventional from a linguistic viewpoint that were utilized for song lyric analysis (Mahedero et al., 2005) or classification (Fell and Sporleder, 2014).[1]

The study critically analyzes the limitations of its data and adopts both a humanities and computational perspective, enhancing the merit of its approach. This work aims to open new pathways in digital and computational musicology to pave the way for future interdisciplinary research endeavors.

[1]The author wishes to express gratitude to the peer reviewers for their insightful literature recommendations, which have significantly contributed to enriching the context and depth of this study.

1.2 Comparing the Structure of Music and Language

One key assumption of this paper is that (written) music and (written) language share enough formal similarities to make these two distinct sign systems structurally comparable and research tools mutually applicable. However, finding and defining those similarities is not an easy task: for instance, both music and spoken language follow an intrinsic logic; syntax and grammar give, to a certain degree, meanings to otherwise arbitrarily combined units, such as words, letters, or as in the case of music, musical notes. But unlike in linguistics, where many broadly available tools have already proven to rather reliably identify and give syntactic meaning to elements of, for example, a sentence, for music, in which even the definition of a phrase, a melody or the function of a chord is mostly ambiguous, such dependable instruments and methods are yet to be found and developed.[2]

One reason might be, of course, that in contrast to music, language conveys a clear, human understandable message, and therefore the correctness of the grammatical rule-set applied, can rather easily be verified by a listener familiar with the language in question. In music, the concepts of syntax and grammar seem less concrete, but equally context dependent: the defining rules of music greatly depend on factors such as the historical and cultural context in which the piece was composed, its genre, as well as the theoretical and cultural background of its composer or listener. Even though shared structural commonalities can be identified within one or across a set of several musical pieces, these, it seems, are not as strictly defined as the rules of language.

[2]While the task of defining and finding similarities between music and language is complex, notable attempts have been made in this area. For instance, a previous, rather extensive publication (Granroth-Wilding, 2013) demonstrates the application of Combinatory Categorial Grammar (CCG) to analyze the hierarchical structure of chord sequences. This approach introduces a formal language, similar to first-order predicate logic, to express the tonal harmonic relationships between chords, serving as a mechanism to map unstructured chord sequences into structured analyses. Additionally, a subsequent study (Granroth-Wilding and Steedman, 2014) successfully showed the effectiveness of applying machine learning techniques to the identification of musical grammar. It describes the use of a formal grammar of jazz chord sequences, combined with statistical modelling techniques, for parsing musical structure, demonstrating that these NLP-adapted statistical techniques can be profitably applied to the analysis of generally ambiguous harmonic structure in music. Further, as a side note, James R. Meehan's work (Meehan, 1979) is notable as it provides a rather historic, however interesting comparison between language and music originating from the early days of Artificial Intelligence (AI).

1.3 The Limitations of Sheet Music Notation

Despite this clear lack of an universally defined intrinsic musical structure, a vast number of musical parameters can still be extracted and, consequently, patterns can, presuming their existence, be identified. However, asking the question of how to extract those parameters is particularly intriguing when considering the complex nature of music.

Sheet music notation is best described as a textual, time-continuous, simplified representation of musical reality. Whereas musical reality - due to its subjective nature - still lacks a clear definition, from a purely physical perspective, music is generally understood as a series of sound events that we interpret and *understand* as music. As a consequence, any sheet music notation, digital or analogue, is by nature a mere simplification or abstraction of an indefinitely *complex musical reality*. To get back to and support the initial assumption of this paper, based on these observations it seems reasonable to argue that sheet music notation, in other words *written music*, is as much a simplified description of *performed music* as *written text* is a simplified description of *spoken language* - therefore, no representation, of musical or literary nature, will ever be able to entirely represent physical reality in its entire complexity; as a general verdict, we can presume that some information loss is *inevitable*.

This, of course, comes with difficulties, as much of the information - that had existed or will exist in the exact moment text turns into sound, is not (yet) contained in our textual source material. But this exact apparent shortcoming, on the other side, reduces the information that we need to work with to a humanly graspable, but in context of this study even more importantly, computationally manageable amount. More over, notation systems bear the potential to enrich descriptions as they give space for including information that is not inherent in the object or phenomenon that they describe (e.g., performance directions, cross-references, subdivisions etc.).

Due to these obvious limitations, music notation systems generally represent only a very limited set of musical parameters. The amount as well as the specific set chosen are usually determined by its anticipated scope of application scenarios. As the number of specific scenarios is virtually

unlimited, over the past decades, parallel to the development of various computational methods for music analysis, various digital formats for storing music (related) information have already been, are expected to be invented or further developed. For a long time, some of the most commonly used music notation systems have been Standard MIDI (.mid) (Loy, 1985)[3] as well as the much younger MusicXML (.mxl/.musicxml) (Good, 2001),[4] both formats focusing on a rather practical aspect of storing music information for playback, or as in case of musicXML for representation in music notation software. Nowadays, the Music Encoding Initiative's schema MEI (Roland, 2000; Hankinson et al., 2011),[5] has established itself as the gold standard for academic music notation. Analogue to the Text Encoding Initiative's format TEI (Aguera et al., 1987),[6] MEI was invented in particular with the intentions to standardise music encoding for scientific and archival use.[7]

1.4 Introducing the Concept of Explicit and Implicit Musical Parameters

When trying to retrieve musical parameters from any form of textual music representation, analogue or digital, it seems reasonable to differentiate between explicit and implicit musical parameters contained within the given source material. As a consequence, the following classification is proposed:

In the context of Music Information Retrieval (MIR) the term explicit musical parameter (EMP) must refer to a musical attribute that is explicitly notated or indicated in the given sheet music notation. The term implicit musical parameter (IMP), on the other hand, should refer to a musical attribute that is not directly stated but can be inferred or deduced, i.e., by comparison or calculation.

EMPs are musical parameters that are explicitly defined and communicated within the given musical notation system. As in .xml-notation, this information would be encoded using specific, predefined symbols and coding conventions. Within this code, those symbols and markings store absolute information, such as pitches, durations, dynamics, tempo markings, articulations, as they were encoded by the annotator. However, given the complex nature of music (cf. Chapter 1.3), any notation system can represent/contain only a finite number of EMPs, which, as a side note, further underlines the inevitable limitations of musical notation as a static representation of musical reality.

IMPs, on the other side, represent relational or contextual information. They aren't directly stated but are deduced by observing changes in one or more EMPs or IMPs across multiple consecutive musical events. For instance, in the context of the aforementioned .xml notation format, a musical interval would be considered an implicit musical parameter, as it is not explicitly notated but can be inferred from the difference in pitch between two compared notes.

As the number of performable calculations is, at least in theory, indefinite, any form of sheet music notation, whether digital or analogue, theoretically contains an equally infinite number of IMPs; of which, of course, not all are equally relevant for music analysis or related research.

Differentiating between EMPs and IMPs helps to better understand and handle complex musical information: EMPs, being directly indicated, form the basic layer of information that is relatively easy to access and process; IMPs, on the other hand, though not directly represented, bring additional levels of complexity which can be calculated and utilized when needed, and ignored when not. This can lead to better and more efficient musical parameter extraction methods, as well as rank and classify musical data representations according to the number of EMPs contained.

1.5 The Use of CL Tools for Music Information Retrieval

With this understanding of explicit and implicit musical parameters well established, we shall now turn to the question of how to navigate and make sense of this vast, sheer endless set of complex information. Luckily, the need to manage and extract patterns from a substantial and often ambiguous dataset is not unique to music: within the field of computational linguistics (CL) many of such explorative tools have already been developed and adopted to a variety of different use-cases. For instance, in the CL those patterns are usually found within repeating combinations of words, letters, or other linguistic elements, expressed as n-grams,

[3] https://www.midi.org/

[4] https://www.musicxml.com/

[5] https://music-encoding.org/

[6] https://tei-c.org/

[7] For a detailed discussion on the challenges and complexities of digitally encoding music notation compared to text encoding, particularly through the Music Encoding Initiative's (MEI) and Text Encoding Initiative's (TEI) respective formats, see (Teich Geertinger, 2021)

which are contiguous sequences of 'n' items from a given sample of text. Unsurprisingly, utilizing n-gram searches analogously in digital musicology research to identify recurring patterns in sequences of musical elements has already been well established, as exemplified in various studies (cf. section 1.1).

Those linguistic elements, seen as the smallest distinguishable units, are typically referred to as tokens. By applying the concept of tokens to music, these then newly created *musical tokens*, which could encompass a variety of explicit and implicit parameters such as individual pitches, durations, or as an example for relational information, horizontal (melodic) intervals, can, as this paper will show in its last section, facilitate the application of CL tools on musical source material, and therefore to a certain degree, close the gap between the computational linguistics and music information retrieval.

2 Data

The second section of this paper will focus on data collecting and preparation. In this process several tools where utilized to convert musical source material into textual data, as necessitated by the tool AntConc used for corpus analysis in sections 3 and 4 of this study (cf. Figure 2).

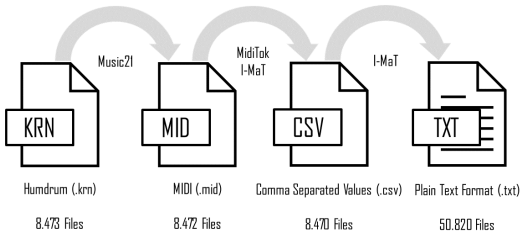


Figure 2: Musical Data Conversion and Custom Database Creation: From Kern Score to MIDI to CSV to Textual Data

The complete process can be divided into two sub-sections: *Corpus Creation* as well as *Tokenization and Database Creation*.

2.1 Corpus Creation

A selection of roughly 8.000 music files that belong to the Essen Folksong Database (Schaffrath, 1997) were chosen as the source material for the study at hand. With more than "20.000 songs and instrumental melodies, mostly from Germany, Poland and China, with minor collections from other (mostly European) countries" (Dahlig, n.d.), the Essen Folksong Database offers an extensive set of diverse monophonic musical pieces (cf. Figure 3). As of today, 8.473 pieces within The Essen Folksong Database are available as Humdrum data translations (Schaffrath and Huron, 1995; Sapp, 2005). Referenced from its associated website,[8] the **kern (.krn) based Humdrum file format is best described as a "text-based description for musical scores, and its primary purpose is for computational musical analysis using the Humdrum Toolkit." The Humdrum format is capable of containing metadata as well as basic music information for each individual musical event, such as pitch, duration, key signature, tempo, meter and others (cf. Figure 5). However, for compatibility with the chosen tool for tokenization, those files needed to be converted to a more universally accepted format: the aforementioned Musical Instrument Digital Interface (MIDI, .mid) file format for music representation. The conversion was accomplished by using the music21 Python framework (Cuthbert and Ariza, 2010), developed by the Massachusetts Institute of Technology (MIT), that offers a variety of tools for handling and manipulating music data.

2.2 Tokenization and Database Creation

After successfully converting nearly all of the available 8.473 .krn-files, with one exception, the resulting 8.472 .mid-files were prepared for further processing. This next step involved the extraction of certain explicit musical parameters for each successive musical event (also commonly referred to as musical *note*). This process, known as tokenization, involved, in the scope of this study, breaking down each musical piece and its entire melodic line into its fundamental components or, as termed earlier, *musical tokens*.

2.2.1 Tokenization: MidiTok

In this study, these *musical tokens* were created utilizing MidiTok (Fradet et al., 2021), an open-source Python package for MIDI file tokenization.[9] MidiTok offers a variety of ten different tokenizers, each of which uses a different pattern to combine several extracted explicit musical parameters (EMPs) into distinct musical tokens. The tokens

[8] http://kern.humdrum.org/help/tour/

[9] https://miditok.readthedocs.io/en/v2.1.7/index.html

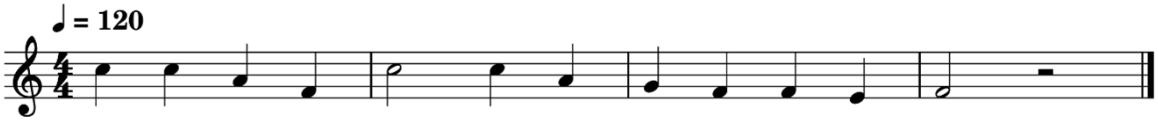


Figure 3: Score Representation of the Folk Song ”Nun schürz dich, Gretlein”; signature: deut0781

are stored in either one-dimensional lists (cf. Figure 6) or two-dimensional nested lists (cf. Figure 7), with the latter format proving useful for data transformation as it allows each musical token to be allocated to a separate array. Those arrays can easily be extracted and stored within a more universal table-like data structure for further processing using computational methods.

Tokenizer	Success Rate (%)	Structure
MIDILike	99.976	1D
MMM	99.976	1D
REMI	99.976	1D
Structured	99.976	1D
TSD	99.976	1D
CPWord	99.976	2D
MuMIDI	99.976	2D
OctupleMono	99.976	2D
Octuple	88.302	2D
REMIPlus	88.302	1D

Table 1: Success rates and structural dimensions of MidiTok tokenizers in processing 8472 .mid-files (sorted by Success Rate (%) in descending order).

Although each of MidiTok's ten tokenizers was applied on the data set equally, not all tokenizers performed with equal reliability. Specifically, the **Octuple** (Zeng et al., 2021) and **REMIPlus** (von Rütte et al., 2022) tokenizers failed to tokenize 991 of the in total 8472 .midi-files (11.697%) (cf. Table 1). Consequently, only the remaining eight tokenizers (1D: **MIDILike** (Oore et al., 2018), **MMM** (Ens and Pasquier, 2020)), **REMI** (Huang and Yang, 2020), **Structured** (Hadjeres and Crestel, 2021), **TSD** (Fradet et al., 2023); 2D: **CPWord** (Hsiao et al., 2021), **MuMIDI** (Ren et al., 2020) and **OctupleMono**) were considered for further study. In the end, **OctupleMono**[10] was chosen

1. for its *high reliability*, failing only two tokenizations out of 8472 files, and

2. its *two-dimensional structure*, which enables presorted parameters and efficient data handling. As shown in Figure 7, this structure is unique to OctupleMono as well as other two-dimensional tokenizers, offering a more organized approach compared to one-dimensional tokenizers due to their grouped parameters or tokens.

In a last data transformation process, the tokenized data of each of the 8470 successfully processed files were compiled into a single .csv file, with the first column being the individual file names, facilitating further data manipulation in subsequent steps. For a visual representation of this compiled data, refer to Table 4.

2.2.2 Data Refinement

The data refinement involved several steps. Initially, the 'Duration' values were converted into numeric values, making them easier to read, eventually calculate on and compare. Following this, a relational implicit musical parameter was calculated: The 'PitchDifferenceToNextPitch' parameter would later (cf. section 4) allow for a horizontal (melodic) interval search while also including its transpositions. Another refinement step involved the removal of redundant prefixes following the pattern "prefix_". This streamlined the data and reduced its overall size (cf. Table 5).

2.2.3 Data Extraction

The last phase of data preparation involved creating individualized data files for each piece and parameter. Essentially, for each music piece, every musical parameter tied to its filename was separated out.

These separated data sets were then saved as individual .txt files within a dedicated folder (cf. Figure 8). It is important to note that each of these .txt-files contained only a single type of extracted parameter (cf. Figure 4), making the data readable

[10]The OctupleMono tokenizer is constructed similarly to the Octuple tokenizer (Zeng et al., 2021). The difference is the exclusion of the 'Program token' in OctupleMono, making it specifically suitable for the tokenization of monophonic music files, which consist of a single track.

by the concordance and analysis tool AntConc used in sections 3 and 4.

```
0.0 -3.0 -4.0 7.0 0.0 -3.0 -2.0 -2.0 0.0 -1.0 1.0 nan
```

Figure 4: Extracted Relational Token "PitchDifferenceToNextPitch" of the folk song "Nun schürz dich, Gretlein"; signature: deut0781

With these steps, the necessary custom database was created and prepared for the next phase of the study, ensuring a solid and easily accessible data foundation for the subsequent corpus analysis.[11]

Note: All the data conversion and tokenization steps outlined in this section have been integrated into the Interactive Music Analysis Tool (I-MaT) python package (Eck, 2023). I-MaT, complete with its robust functionalities for corpus creation, database creation, tokenization, as well as music analysis is openly available for access and usage via GitHub.[12]

3 Methodology

In the following section, the methodological approach, used for demonstrating the applicability of well-established CL tools for answering questions emanating from the field of musicology will be presented.[13]

3.1 The Text Concordance and Analysis Tool AntConc

The tool of choice was the freeware corpus analysis toolkit AntConc (Anthony, 2004), which can be described as an easy-to-use tool originally developed for concordancing and text analysis. AntConc uses several tools that include pattern search, pattern distribution analysis and similar functions for analysing individual text files or conducting corpus studies. AntConc 4.2.0[14] offers these functions via the Keyword in Context (KWIC) tool, the Plot tool, and other tools such as Cluster, N-Gram, Collocate, Wordlist, and Keyword-List.

3.2 Utilizing AntConc for statistical analysis in the context of MIR

In a first step, AntConc's in-built Corpus Manager was used to create a custom database. For this task, the folder "PitchDifferenceToNextPitch" created earlier served as the source, containing all the necessary .txt-files to be included in the custom corpus. A critical aspect to consider during the corpus creation was to reconfigure the 'token_definition' setting: to ensure an accurate computation of types and tokens, this parameter was adjusted to include all numerical values and ".na" (value: [-0123456789.na]). This step was essential as the data in use primarily consists of numerical values but also possible "nan" [= no subsequent interval available] entries (cf. Chapter 4.2).

Utilizing AntConc for statistical analysis in the context of MIR demanded a conceptual re-framing: numerical values, which represent musical parameters, are treated as words. In theory, any desired number of numerical values, i.e., numerical representations of musical parameters, extracted from one and the same musical event (e.g., pitch, duration etc.) can be combined. A higher number would result in more complex, but also more exact representations of musical events.

For simplicity, in this study only one musical parameter was extracted from each musical event. These single-element word-sequences are separated by whitespace characters, imitating the structure of a sentence (cf. Figure 4). This unconventional approach is necessitated by the nature of the tool, which is designed primarily for text analysis. In short, for AntConc to correctly interpret the musical text data at hand, numbers had to be reinterpreted as numerical representations of musical events, as words ultimately forming sentences.

Though this approach might seem unusual, it allowed, as the final section will show, to nearly fully engage the capabilities of AntConc in a musicological context. This 'numerical-to-textual' approach made it possible, as the following examples show, to perform n-gram pattern searches, conduct cluster analyses on and plot pattern distributions on these newly created 'musical words' and 'sentences'.

[11] All the data (.mid-conversions, tokenizations, cleaned and enhanced databases (.csv-files), and extracted parameters (.txt-files) used in this study are freely available on figshare.com for further research and exploration: 'https://figshare.com/s/03179521a56c88bfac63'.

[12] https://github.com/sebastian-eck/I-MaT

[13] During the review process, the author has been made aware of a similar approach recently attempted (Norgaard and Römer, 2022), but as the work is not readily accessible, this study still offers unique contributions to digital musicology with its distinct use case and theoretical framework.

[14] https://www.laurenceanthony.net/software/antconc/.

4 Demonstration

In this last section, three distinct ways to utilize AntConc for music analysis will be demonstrated. Each of these demonstrations will focus on a particular aspect of music analysis:

1. **melodic pattern search** (N-Gram-Tool)
2. **end phrase pattern search** (Cluster-Tool)
3. **pattern distribution** (Plot-Tool)

These analyses will be performed on the custom database, created within AntConc earlier in section 2.2. Numerical representations of musical events will be treated as textual entities or 'musical words', allowing AntConc to be used as an user-friendly tool for conducting various music analyses.

Note: As the pattern search will be performed not on absolute pitch values, but on values extracted from the relational parameter "PitchDifferenceToNextPitch", patterns will be identified across melodic transpositions.

4.1 N-Gram-Tool

Identifying Common Melodic Sequences: One musicological question that can be addressed using the n-gram pattern search feature of AntConc is identifying the most common melodic sequences within a corpus of music. After interpreting sequences of relative pitch intervals, or 'musical words', as n-grams, the pattern search function revealed several sequences as the most prominent (cf. Table 2).

Type	Rank	Freq	Range
0.0 0.0 0.0 0.0 0.0	1	1210	441
-2.0 -2.0 -1.0 -2.0 -2.0	2	894	683
2.0 -2.0 -2.0 -1.0 -2.0	3	846	637
2.0 1.0 -1.0 -2.0 -2.0	4	555	436
-2.0 -1.0 -2.0 -2.0 0.0	5	543	455
-2.0 -1.0 -2.0 -2.0 2.0	6	519	413
2.0 2.0 -2.0 -2.0 -1.0	7	499	363
3.0 -2.0 -1.0 -2.0 -2.0	8	494	373
1.0 2.0 2.0 -2.0 -2.0	9	484	353
2.0 2.0 1.0 -1.0 -2.0	10	469	352

Table 2: Results (Excerpt) of a N-Gram-Search (n = 5) Performed on the Custom Database

These patterns can be grouped as:

1. **Repetitive Sequences:** {0.0 0.0 0.0 0.0 0.0}
2. **Descending Interval Patterns:** {-2.0 -2.0 -1.0 -2.0 -2.0} and {-2.0 -1.0 -2.0 -2.0 0.0}
3. **Combination of Ascending and Descending Intervals:** {2.0 -2.0 -2.0 -1.0 -2.0} and {2.0 1.0 -1.0 -2.0 -2.0}
4. **Alternating Interval Structures:** {2.0 2.0 -2.0 -2.0 -1.0} and {3.0 -2.0 -1.0 -2.0 -2.0}

Conclusions: "Regional and Cultural Influences" - The recurrence of certain patterns, such as repetitive sequences or descending interval patterns, might indicate patterns commonly used in melody construction. Given the diversity of the Essen Folksong Collection, a n-gram-search that groups results by geographic information (as indicated in the individual file names, e.g., "steier09...", cf. Table 9) could reveal specific patterns influenced by regional or cultural traditions, revealing patterns more prevalent in Germanic folk songs compared to those from other regions, in relation to their total number.

Note: Here, as well as in the following examples, melodic intervals are represented as numeric values, each whole number representing a semitone step. Negative values represent descending, positive values ascending intervals. Zero-values, such as in the most prominent interval-series (cf. Table 2), represent repetitions of the same pitch.

4.2 Cluster-Tool

Identifying Common End Phrase Patterns: By employing AntConc's pattern search capabilities, musicologists can identify common end phrase patterns in a corpus of music. This can be achieved by searching for recurring sequences at the end of musical phrases.

Cluster	Rank	Freq	Range
-2.0 -1.0 -2.0 -2.0 nan	1	423	423
-2.0 -2.0 2.0 -2.0 nan	2	160	160
0.0 -2.0 0.0 -2.0 nan	3	157	157
-2.0 -2.0 -1.0 1.0 nan	4	132	132
2.0 2.0 -2.0 -2.0 nan	5	119	119
-2.0 2.0 -2.0 -2.0 nan	6	98	98
-2.0 -2.0 -1.0 -2.0 nan	7	95	95
-2.0 -2.0 -3.0 -2.0 nan	8	83	83
1.0 -1.0 -2.0 -2.0 nan	9	82	82
-1.0 -2.0 0.0 -2.0 nan	10	80	80

Table 3: Results (Excerpt) of a Cluster Search (n = 5; Search Term Position = On Right; Search Term = nan) Performed on the Custom Database

Based on the table results, we can group the common end phrase patterns as follows:

1. **Descending Endings:** The most frequent pattern, {-2.0 -1.0 -2.0 -2.0 nan}, can be interpreted as a part of a descending major scale, corresponding to, e.g., "G F E D C" in a major key. This pattern, as well as others like {-2.0 -2.0 2.0 -2.0 nan}, suggests a common use of descending intervals at the end of phrases. The prevalence of such patterns, particularly the descending major scale, might not sound surprising to many listeners due to its widespread use in traditional folk music. But more interestingly, it reflects an expected stylistic or structural preference in folk song compositions for simplicity, aligning with the perceived conventional or rather 'simple' nature of folk music melodies.

2. **Static and Minor Movements:** Patterns such as {0.0 -2.0 0.0 -2.0 nan} and {-2.0 -2.0 -1.0 1.0 nan} demonstrate either static (repeated pitches) or minor interval movements. These may reflect a simplicity and compactness in phrase endings commonly found in folk songs.

3. **Ascending and Mixed Intervals:** Patterns like {2.0 2.0 -2.0 -2.0 nan} and {1.0 -1.0 -2.0 -2.0 nan} include a mix of ascending and descending movements. This variety might represent a richer melodic closure in some folk songs.

Conclusions: The Cluster-Tool analysis provides insights into common phrase-ending techniques in the Essen Folksong Database. These patterns offer a glimpse into the melodic structures and stylistic tendencies in folk song compositions, particularly in how phrases are conventionally concluded. When considering geographic parameters (c.f. section 4.1), the variety and frequency of these patterns can also reflect regional or cultural influences in folk song traditions, contributing to the understanding of musicological characteristics within this genre.

Note: Here, the string "nan" indicates the presence of end-notes. This particular string acted as a placeholder when the relational token "PitchDifferenceToNextPitch" was calculated, but no subsequent pitch value could be identified. In short, "nan" is a marker for situations where the calculation could not continue due to the absence of a following pitch. Since this calculation was performed on sequential pitch values only, rests within the file will not be labeled. Furthermore, we can assume that the automatic inclusion of a "start" string at the beginning of each .txt-file would effortlessly allow for an analogue 'start pattern search'.

4.3 Plot-Tool

Visualizing Pattern Distributions Across a Musical Corpus: A third application could involve using AntConc's plotting function to visualize the distribution of specific melodic patterns across a musical corpus. For instance, a musicologist could search for a particular melodic sequence, interval pattern, or, assuming a suitable custom corpus has been created, a rhythmic motif. The results can then be displayed by using the plotting function to visually map out where and how frequently these patterns occur across different pieces (cf. Figure 9). Visualizing pattern distributions seems particularly useful within polyphonic music, such as imitative fugues, or isorhythmic motets.

5 Conclusion

This study has successfully demonstrated the practical applicability of well-established computational linguistics (CL) tools, such as AntConc, in the the field of digital musicology. The successful reinterpretation of tokenized music data as 'musical words' outlined in sections 2 and 3 has not only indicated the existence of certain inherent structural/formal similarities between language and music; it also unveiled the potential benefits of developing new analytical tools and methods that can be used both within the fields of linguistics and musicology.

A notable area for future code development lies in the optimization of the tokenization and parameter extraction processes. Integrating more versatile tools like music21 into the tokenization process could address the current limitations encountered with MidiTok, particularly its restriction to the rather unreliable .mid-file format. Utilizing music21's comprehensive capabilities would enable the processing of a broader spectrum of music file formats, enhancing the methodology's versatility as well as its robustness by further reducing its dependency on third-party python packages.

Lastly, AntConc was mainly used for exemplary reasons within this study. The incorporation of advanced Natural Language Processing (NLP) packages, such as NLTK or SpaCy, directly into the aforementioned Interactive Music Analysis Tool

(I-MaT), as outlined towards the end of section 2.2, presents a significant opportunity. This integration would enable more sophisticated analysis capabilities, allowing for the calculation, exportation, and visualization of results within a unified platform. Such an approach could make the analysis process more user-friendly and accessible, particularly for scholars and students who are new to the field of digital musicology. It would further lower the barriers to entry in this interdisciplinary field, enhancing the appeal and reach of digital musicological studies.

We can assume that this research not only makes a step towards bridging the gap between computational linguistics and musicology but also lays the groundwork for a more integrated and holistic approach to the analysis of music and language.

References

Helen Aguera et al. 1987. The preparation of text encoding guidelines. Closing Statement of the Vassar Planning Conference.

Laurence Anthony. 2004. Antconc: A learner and classroom friendly, multi-platform corpus analysis toolkit. *proceedings of IWLeL*, pages 7–13.

Cors Brinkman, Manolis Fragkiadakis, and Xander Bos. 2016. Online music recognition: the echoprint system.

Nicholas Cook. 2005. Towards the complete musicologist. In *Proceedings of the Sixth International Conference on Music Information Retrieval (ISMIR 2005)*.

Michael Scott Cuthbert and Christopher Ariza. 2010. music21: A toolkit for computer-aided musicology and symbolic music data. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 637–642, Utrecht, Netherlands. Version: Author's final manuscript.

Ewa Dahlig. n.d. Essen associative code and folksong database. http://www.esac-data.org/. Accessed on 2023/07/28.

Sebastian Oliver Eck. 2023. Interactive music analysis tool (i-mat). In *Proceedings of the JADH2023 conference*, pages 20–23. Japanese Association for Digital Humanities.

Jeff Ens and Philippe Pasquier. 2020. Mmm : Exploring conditional multi-track music generation with the transformer.

Michael Fell and Caroline Sporleder. 2014. Lyrics-based analysis and classification of music. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical papers*, pages 620–631.

Nathan Fradet, Jean-Pierre Briot, Fabien Chhel, Amal El Fallah Seghrouchni, and Nicolas Gutowski. 2021. MidiTok: A python package for MIDI file tokenization. In *Extended Abstracts for the Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference (ISMIR 2021)*.

Nathan Fradet, Jean-Pierre Briot, Fabien Chhel, Amal El Fallah Seghrouchni, and Nicolas Gutowski. 2023. Byte pair encoding for symbolic music.

Michael Good. 2001. Musicxml: An internet-friendly format for sheet music. In *Proceedings of the XML 2001 Conference*.

Mark Granroth-Wilding and Mark Steedman. 2014. A robust parser-interpreter for jazz chord sequences. *Journal of New Music Research*, 43(4):355–374.

Mark Thomas Granroth-Wilding. 2013. Harmonic analysis of music using combinatory categorial grammar.

Gaëtan Hadjeres and Léopold Crestel. 2021. The piano inpainting application.

Andrew Hankinson, Perry Roland, and Ichiro Fujinaga. 2011. The music encoding initiative as a document-encoding framework. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, pages 293–298.

Ruben Hillewaere, Bernard Manderick, and Darrell Conklin. 2009a. Global feature versus event models for folk song classification. In *ISMIR*, volume 2009, page 10th.

Ruben Hillewaere, Bernard Manderick, and Darrell Conklin. 2009b. Melodic models for polyphonic music classification. In *Second International Workshop on Machine Learning and Music*.

Ruben Hillewaere, Bernard Manderick, and Darrell Conklin. 2010. String quartet classification with monophonic models. In *ISMIR*, pages 537–542.

Ruben Hillewaere, Bernard Manderick, and Darrell Conklin. 2014. Alignment methods for folk tune classification. In *Data analysis, machine learning and knowledge discovery*, pages 369–377. Springer.

María Hontanilla, Carlos Pérez-Sancho, and Jose M Inesta. 2013. Modeling musical style with language models for composer recognition. In *Pattern Recognition and Image Analysis: 6th Iberian Conference, IbPRIA 2013, Funchal, Madeira, Portugal, June 5-7, 2013. Proceedings 6*, pages 740–748. Springer.

Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. 2021. Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):178–186.

Yu-Siang Huang and Yi-Hsuan Yang. 2020. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, page 1180–1188, New York, NY, USA. Association for Computing Machinery.

David Huron. 1999. The new empiricism: Systematic musicology in a postmodern age. *Berkeley, University of California*, page 2.

David Huron et al. 1996. The melodic arch in western folksongs. *Computing in Musicology*, 10:3–23.

Maximos A Kaliakatsos-Papakostas, Michael G Epitropakis, and Michael N Vrahatis. 2010. Musical composer identification through probabilistic and feedforward neural networks. In *European Conference on the Applications of Evolutionary Computation*, pages 411–420. Springer.

Allison Lahnala, Gauri Kambhatla, Jiajun Peng, Matthew Whitehead, Gillian Minnehan, Eric Guldan, Jonathan K Kummerfeld, Anıl Çamcı, and Rada Mihalcea. 2021. Chord embeddings: Analyzing what they capture and their role for next chord prediction and artist attribute prediction. In *Artificial Intelligence in Music, Sound, Art and Design: 10th International Conference, EvoMUSART 2021, Held as Part of EvoStar 2021, Virtual Event, April 7–9, 2021, Proceedings 10*, pages 171–186. Springer.

Gareth Loy. 1985. Musicians make a standard: The midi phenomenon. *Computer Music Journal*, 9(4):8–26.

Sephora Madjiheurem, Lizhen Qu, and Christian Walder. 2016. Chord2vec: Learning musical chord embeddings. In *Proceedings of the constructive machine learning workshop at 30th conference on neural information processing systems (NIPS2016), Barcelona, Spain*.

Jose P. G. Mahedero, Álvaro Martínez, Pedro Cano, Markus Koppenberger, and Fabien Gouyon. 2005. Natural language processing of lyrics. In *Proceedings of the 13th annual ACM international conference on Multimedia (MULTIMEDIA '05)*, pages 475–478, New York, NY, USA. Association for Computing Machinery.

Rudolf Mayer and Andreas Rauber. 2011. Musical genre classification by ensembles of audio and lyrics features. In *Proceedings of international conference on music information retrieval*, pages 675–680.

James R Meehan. 1979. An artificial intelligence approach to tonal music theory. In *Proceedings of the 1979 annual conference*, pages 116–120.

Martin Norgaard and Ute Römer. 2022. Patterns in music: How linguistic corpus analysis tools can be used to illuminate central aspects of jazz improvisation. *Jazz Education in Research and Practice*, 3(1):3–26.

Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. 2018. This time with feeling: Learning expressive musical performance. *Neural Computing and Applications*, 32:955–967.

Yi Ren, Jinzheng He, Xu Tan, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2020. Popmag: Pop music accompaniment generation. In *Proceedings of the 28th ACM International Conference on Multimedia*, page 1198–1206. Association for Computing Machinery.

Perry Roland. 2000. Xml4mir: Extensible markup language for music information retrieval. In *Proceedings of the 1st International Society for Music Information Retrieval Conference (ISMIR 2001)*.

Dimitri von Rütte, Luca Biggio, Yannic Kilcher, and Thomas Hofmann. 2022. Figaro: Generating symbolic music with fine-grained artistic control.

Craig Stuart Sapp. 2005. Online database of scores in the humdrum file format. In *Proceedings of the ISMIR*, pages 664–665.

Helmut Schaffrath. 1997. The essen associative code: a code for folksong analysis. In Eleanor Selfridge-Field, editor, *Beyond MIDI: the handbook of musical codes*, pages 343–361. MIT Press, Cambridge, Massachusetts.

Helmut Schaffrath and David Huron. 1995. The essen folksong collection in kern format. `http://kern.ccarh.org/browse?l=essen`. Accessed on 2023/07/28.

Bo Shao, Tao Li, and Mitsunori Ogihara. 2008. Quantify music artist similarity based on style and mood. In *Proceedings of the 10th ACM workshop on web information and data management*, pages 119–124.

Jonatan Taminau, Ruben Hillewaere, Stijn Meganck, Darrell Conklin, Ann Nowé, and Bernard Manderick. 2009. Descriptive subgroup mining of folk music. In *2nd International Workshop on Machine Learning and Music (MML 2009), Bled, Slovenia*.

Axel Teich Geertinger. 2021. Digital Encoding of Music Notation with MEI. In Margrethe Støkken Bue and Annika Rockenberger, editors, *Notated Music in the Digital Sphere. Possibilities and Limitations*, volume 15 of *Nota bene – Studies from the National Library of Norway*, page 35–56. National Library of Norway, Oslo.

Peter Van Kranenburg and Eric Backer. 2005. Musical style recognition—a quantitative approach. In *Handbook of pattern recognition and computer vision*, pages 583–600. World Scientific.

Anja Volk, Frans Wiering, and Peter Van Kranenburg. 2011. Unfolding the potential of computational musicology. In *Proceedings of the 13th International Conference on Informatics and Semiotics in Organisations*, pages 137–144. Fryske Akademy.

Jacek Wołkowicz, Zbigniew Kulka, and Vlado Kešelj. 2008. N-gram-based approach to composer recognition. *Archives of Acoustics*, 33(1):43–55.

Mingliang Zeng, Xu Tan, Rui Wang, Zeqian Ju, Tao Qin, and Tie-Yan Liu. 2021. MusicBERT: Symbolic music understanding with large-scale pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 791–800, Online. Association for Computational Linguistics.

Appendices

filename	Pitch	Velocity	Duration	Position	Bar
deut0781	Pitch_72	Velocity_91	Duration_1.0.16	Position_0	Bar_0
deut0781	Pitch_72	Velocity_91	Duration_1.0.16	Position_16	Bar_0
deut0781	Pitch_69	Velocity_91	Duration_1.0.16	Position_32	Bar_0
deut0781	Pitch_65	Velocity_91	Duration_1.0.16	Position_48	Bar_0
deut0781	Pitch_72	Velocity_91	Duration_2.0.16	Position_0	Bar_1
deut0781	Pitch_72	Velocity_91	Duration_1.0.16	Position_32	Bar_1
deut0781	Pitch_69	Velocity_91	Duration_1.0.16	Position_48	Bar_1
deut0781	Pitch_67	Velocity_91	Duration_1.0.16	Position_0	Bar_2
deut0781	Pitch_65	Velocity_91	Duration_1.0.16	Position_16	Bar_2
deut0781	Pitch_65	Velocity_91	Duration_1.0.16	Position_32	Bar_2
deut0781	Pitch_64	Velocity_91	Duration_1.0.16	Position_48	Bar_2
deut0781	Pitch_65	Velocity_91	Duration_2.0.16	Position_0	Bar_3

Table 4: Database (OctupleMono) Representation of the Folk Song ”Nun schürz dich, Gretlein”; Signature: deut0781

filename	Pitch	Velocity	Duration	Position	Bar	PitchDifferenceToNextPitch
deut0781	72	91	1	0	0	0
deut0781	72	91	1	16	0	-3
deut0781	69	91	1	32	0	-4
deut0781	65	91	1	48	0	7
deut0781	72	91	2	0	1	0
deut0781	72	91	1	32	1	-3
deut0781	69	91	1	48	1	-2
deut0781	67	91	1	0	2	-2
deut0781	65	91	1	16	2	0
deut0781	65	91	1	32	2	-1
deut0781	64	91	1	48	2	1
deut0781	65	91	2	0	3	nan

Table 5: Refined Database (OctupleMono) Representation of the Folk Song ”Nun schürz dich, Gretlein”; Signature: deut0781

```
!!!OTL: SCHUERZ DICH GRETLE
!!!ARE: Europa, Mitteleuropa, Deutschland
!!!SCT: E0113B
!!!YEM: Copyright 1995, estate of Helmut Schaffrath.
**kern
*ICvox
*Ivox
*M4/4
*k[b-]
*F:
=1
{4cc
4cc
4a
4f
=2
2cc
4cc}
{4a
=3
4g
4f
4f
4e
=4
2f
2r}
==
!!!AGN: erzaehlendes Volks - Lied, Schalk -, Schelmen - Lied, betrogene Liebe, Ballade ?
!!!ONB: ESAC (Essen Associative Code) Database: ERK1
!!!AMT: simple quadruple
!!!AIN: vox
!!!EED: Helmut Schaffrath
!!!EEV: 1.0
*-
```

Figure 5: Humdrum Representation of the Folk Song ”Nun schürz dich, Gretlein”; Signature: deut0781

```
[TokSequence(tokens=[
'Pitch_72', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_72', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_69', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_65', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_72', 'Velocity_91', 'Duration_2.0.8', 'TimeShift_2.0.8',
'Pitch_72', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_69', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_67', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_65', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_65', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_64', 'Velocity_91', 'Duration_1.0.8', 'TimeShift_1.0.8',
'Pitch_65', 'Velocity_91', 'Duration_2.0.8'],
ids=[55, 114, 131, 196, 55, 114, 131, 196, 52, 114, 131, 196, 48, 114,
131, 196, 55, 114, 139, 204, 55, 114, 131, 196, 52, 114, 131, 196, 50,
114, 131, 196, 48, 114, 131, 196, 48, 114, 131, 196, 47, 114, 131, 196,
48, 114, 139],
bytes=None, events=[Event(type=Pitch, value=72, time=0, desc=72), […]],
ids_bpe_encoded=False, _ids_no_bpe=None)]
```

Figure 6: “Structured” Token Representation (One-Dimensional) of the Folk Song ”Nun schürz dich, Gretlein”; Signature: deut0781

```
[TokSequence(tokens=
[['Pitch_72', 'Velocity_91', 'Duration_1.0.16', 'Position_0', 'Bar_0'],
['Pitch_72', 'Velocity_91', 'Duration_1.0.16', 'Position_16', 'Bar_0'],
['Pitch_69', 'Velocity_91', 'Duration_1.0.16', 'Position_32', 'Bar_0'],
['Pitch_65', 'Velocity_91', 'Duration_1.0.16', 'Position_48', 'Bar_0'],
['Pitch_72', 'Velocity_91', 'Duration_2.0.16', 'Position_0', 'Bar_1'],
['Pitch_72', 'Velocity_91', 'Duration_1.0.16', 'Position_32', 'Bar_1'],
['Pitch_69', 'Velocity_91', 'Duration_1.0.16', 'Position_48', 'Bar_1'],
['Pitch_67', 'Velocity_91', 'Duration_1.0.16', 'Position_0', 'Bar_2'],
['Pitch_65', 'Velocity_91', 'Duration_1.0.16', 'Position_16', 'Bar_2'],
['Pitch_65', 'Velocity_91', 'Duration_1.0.16', 'Position_32', 'Bar_2'],
['Pitch_64', 'Velocity_91', 'Duration_1.0.16', 'Position_48', 'Bar_2'],
['Pitch_65', 'Velocity_91', 'Duration_2.0.16', 'Position_0', 'Bar_3']],
ids=[[55, 26, 19, 4, 4], [55, 26, 19, 20, 4], [52, 26, 19, 36, 4], [48,
26, 19, 52, 4], [55, 26, 35, 4, 5], [55, 26, 19, 36, 5], [52, 26, 19, 52,
5], [50, 26, 19, 4, 6], [48, 26, 19, 20, 6], [48, 26, 19, 36, 6], [47, 26,
19, 52, 6], [48, 26, 35, 4, 7]],
bytes=None, events=None, ids_bpe_encoded=False, _ids_no_bpe=None)]
```

Figure 7: “OctupleMono” Token Representation (Two-Dimensional) of the Folk Song ”Nun schürz dich, Gretlein”; Signature: deut0781

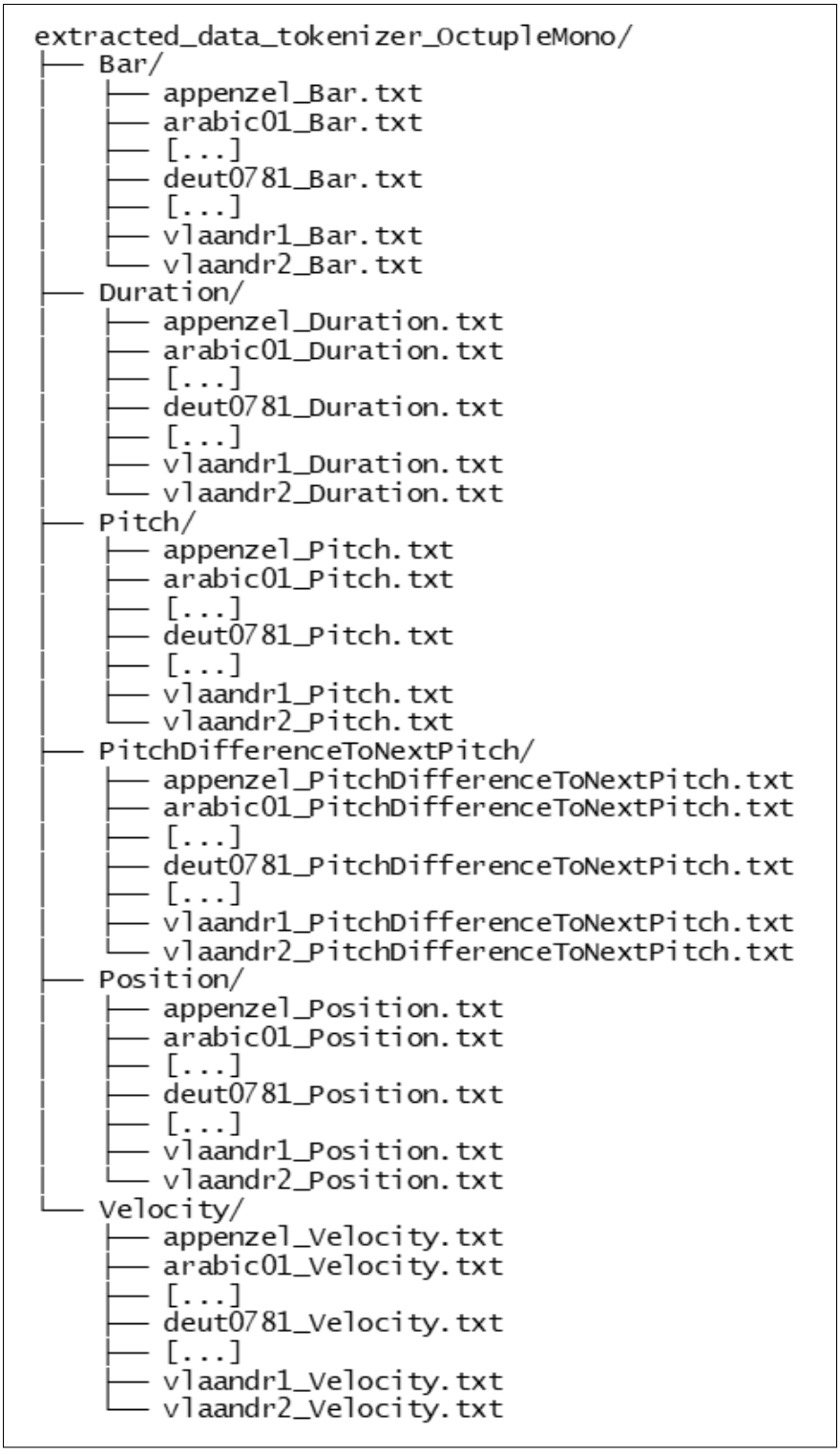
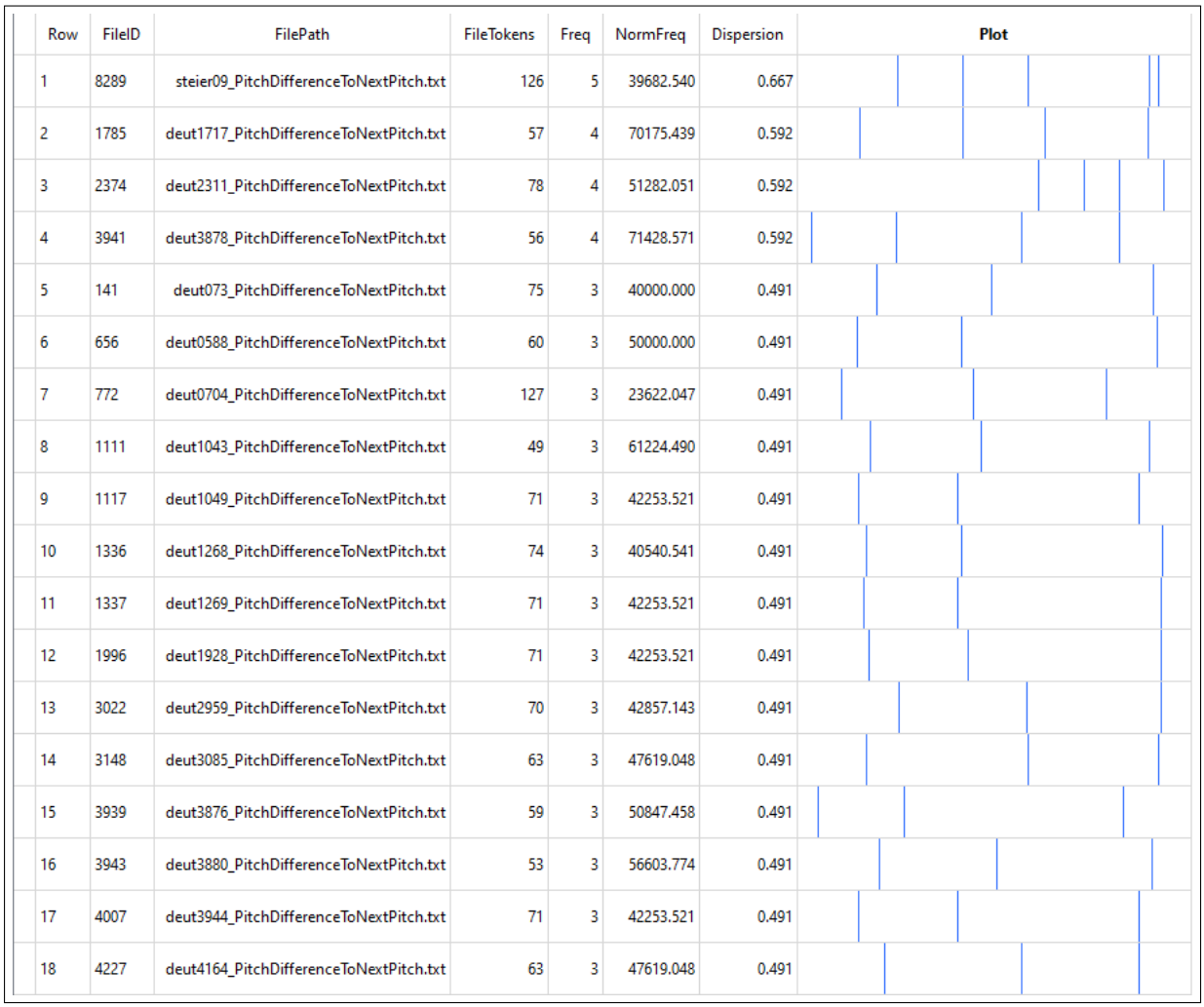


Figure 8: Organized Folder Structure of Extracted Data, Segregated by Parameter and Musical Piece



Row	FileID	FilePath	FileTokens	Freq	NormFreq	Dispersion	Plot
1	8289	steier09_PitchDifferenceToNextPitch.txt	126	5	39682.540	0.667	
2	1785	deut1717_PitchDifferenceToNextPitch.txt	57	4	70175.439	0.592	
3	2374	deut2311_PitchDifferenceToNextPitch.txt	78	4	51282.051	0.592	
4	3941	deut3878_PitchDifferenceToNextPitch.txt	56	4	71428.571	0.592	
5	141	deut073_PitchDifferenceToNextPitch.txt	75	3	40000.000	0.491	
6	656	deut0588_PitchDifferenceToNextPitch.txt	60	3	50000.000	0.491	
7	772	deut0704_PitchDifferenceToNextPitch.txt	127	3	23622.047	0.491	
8	1111	deut1043_PitchDifferenceToNextPitch.txt	49	3	61224.490	0.491	
9	1117	deut1049_PitchDifferenceToNextPitch.txt	71	3	42253.521	0.491	
10	1336	deut1268_PitchDifferenceToNextPitch.txt	74	3	40540.541	0.491	
11	1337	deut1269_PitchDifferenceToNextPitch.txt	71	3	42253.521	0.491	
12	1996	deut1928_PitchDifferenceToNextPitch.txt	71	3	42253.521	0.491	
13	3022	deut2959_PitchDifferenceToNextPitch.txt	70	3	42857.143	0.491	
14	3148	deut3085_PitchDifferenceToNextPitch.txt	63	3	47619.048	0.491	
15	3939	deut3876_PitchDifferenceToNextPitch.txt	59	3	50847.458	0.491	
16	3943	deut3880_PitchDifferenceToNextPitch.txt	53	3	56603.774	0.491	
17	4007	deut3944_PitchDifferenceToNextPitch.txt	71	3	42253.521	0.491	
18	4227	deut4164_PitchDifferenceToNextPitch.txt	63	3	47619.048	0.491	

Figure 9: Results (Excerpt) Created by Using the Plot-Tool (Search Term = -2.0 -2.0 -1.0 -2.0 -2.0) on the Custom Database