Investigating the Harmonic Syntax of Rock Music: A Corpus Study

Matthew Ludwig

Schulich School of Music

McGill University, Montreal

December 2023

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Music Theory

Abstract:

In this thesis I will investigate the harmonic syntax of rock music by examining and comparing corpora and implementing various statistical models. I will begin with an introduction to harmonic syntax and discuss several models of harmonic syntax that music theorists have used. In particular, I will consider Christopher White and Ian Quinn's article "Chord Context and Harmonic Function in Tonal Music" and discuss the statistical model that they used and their results. Following this, I will introduce a corpus of rock music that was built by David Temperley and Trevor de Clercq in 2011 and examine their article "A Corpus Analysis of Rock Harmony". Then, I will expand De Clercq and Temperley's rock corpus and create a larger corpus called the Expanded Rock Corpus (ERC). Following this, I will replicate the basic statistical analyses that they conducted. After presenting patterns in the larger corpus with histograms, heat maps and dendrograms, I will implement two statistical models and attempt to model the harmonic syntax of the ERC. In particular, following the work of Tsushima et al. in "Generative statistical models with self-emergent grammar of chord sequences", I will implement Hidden Markov models and Probabilistic Context-Free Grammar models and attempt to capture the non-local and hierarchical harmonic dependencies in the ERC. Finally, I will discuss several difficulties and issues that I encountered while implementing these models on the ERC and conclude by discussing several directions for further research.

Resume:

Dans cette thèse, j'étudierai la syntaxe harmonique de la musique rock en examinant et en comparant des corpus et en mettant en œuvre divers modèles statistiques. Je commencerai par une introduction à la syntaxe harmonique et discuterai de plusieurs modèles de syntaxe harmonique utilisés par les théoriciens de la musique. En particulier, j'examinerai l'article de Christopher White et Ian Quinn intitulé "Chord Context and Harmonic Function in Tonal Music" et discuterai du modèle statistique qu'ils ont utilisé et des résultats qu'ils ont obtenus. Ensuite, je présenterai un corpus de musique rock constitué par David Temperley et Trevor de Clercq en 2011 et j'examinerai leur article "A Corpus Analysis of Rock Harmony". Ensuite, je développerai le corpus rock de De Clercq et Temperley et créerai un corpus plus large appelé Expanded Rock Corpus (ERC). Ensuite, je reproduirai les analyses statistiques de base qu'ils ont effectuées. Après avoir présenté des modèles dans le corpus élargi à l'aide d'histogrammes, de cartes thermiques et de dendrogrammes, je mettrai en œuvre deux modèles statistiques et tenterai de modéliser la syntaxe harmonique de l'ERC. En particulier, en suivant le travail de Tsushima et al. dans "Generative statistical models with self-emergent grammar of chord sequences", je mettrai en œuvre des modèles de Markov cachés et des modèles de grammaire contextuelle libre probabiliste et j'essaierai de capturer les dépendances harmoniques non locales et hiérarchiques dans l'ERC. Enfin, je discuterai de plusieurs difficultés et problèmes que j'ai rencontrés lors de la mise en œuvre de ces modèles sur l'ERC et je conclurai en discutant de plusieurs directions pour des recherches ultérieures.

Acknowledgements:

I would like to thank my supervisor, Jon Wild for helping me find and for allowing me to pursue a topic involving statistics and music theory. I would also like to acknowledge Eita Nakamura for his generosity in letting me use his code from his paper "Generative statistical models with self-emergent grammar of chord sequences" and for assisting me while setting up the model on the Expanded Rock Corpus data. I would also like to thank the Schulich School of Music for the generous support they provided through the Max Stern Fellowship without which the thesis would not have been possible. I would also like to thank my friends Philipp, Hamid and Linglan for the many insightful conversations surrounding statistics and machine learning which generated many ideas that took form in this thesis. Finally, I would like to thank my parents and brother for their unconditional support.

Table of Contents

1. Introduction to Harmonic Syntax and Models of Harmonic Syntax

In this chapter, I will introduce the concept of harmonic syntax and discuss several models that have been used to study the harmonic syntax of classical and popular music. I will also discuss the connection between syntax in language and the various statistical models that music theorists have used to study harmonic syntax.

1.1 Syntax in Music and Language

In music and language, the concept of syntax refers to the rules for arranging a sequence of items. These rules in both language and music vary across different languages, time periods and different genres of music and have been investigated by both linguists and music theorists. In "Musical Syntax I: Theoretical Perspectives", Rohrmeier and Pearce define musical syntax as "a formal characterization of the principles governing permissible sequential structure in music." [1] These principles can refer to many parameters of music including melody, harmony and rhythm. They also note how musical syntax "characterizes sequences of musical events generated from a *lexicon of building blocks* and a set of *rules* governing how the building blocks are combined."[2] In this thesis, I will focus on the musical feature of harmony and study the harmonic syntax of rock music.

In an effort to understand the harmonic syntax of music, music theorists have used a variety of models. In introducing some of the models that theorists have used to understand harmonic syntax, I will explain how some models are able to account for more of the complexity of the

[1] Martin Rohrmeier and Marcus Pearce, "Musical Syntax I: Theoretical Perspectives," in *Springer Handbook of Systematic Musicology*, ed. Rolf Baber (Berlin: Springer, 2018), 475.

[2] Ibid.

underlying harmonic syntax than others. In the remainder of this chapter, I will introduce several models that music theorists have used to study harmonic syntax.

1.2 Markov and n-gram Models

The first and simplest model that music theorists have used to study harmonic syntax is called an n-gram or Markov model. These models are widely used in linguistics, finance, ecology and many other disciplines. In modeling harmonic syntax, n-gram models give the probability that a chord will occur next in a sequence of chords given some number of chords. This probability ultimately gives us a rough model of how the harmonic syntax of a sequence of chords is behaving on a local level. In using n-gram or Markov models, the length of the context of the preceding items in the sequence (which is referred to as the order of the model) is often varied. As a general trend, increasing the order of the model, that is the number of preceding elements, often increases the accuracy that the model is able to correctly predict the next element in a sequence. A Markov model of order one, or a 1-gram model, only takes the previous element into account when making a prediction for the next element in the sequence and provides a very rough estimate of how the harmonic syntax is behaving. One drawback of Markov models is that they are not able to account for any of the nonlocal dependencies in the sequences they model as they only provide the probability of the next chord based on the sequence of some previous number of chords. In other words, they only use the transition probabilities between consecutive elements in a sequence to determine which chord comes next and don't consider the ways in which nonconsecutive elements in the sequence could be related to one another. One example of work done by music theorists that uses n-gram or Markov models to study harmonic syntax is

from Trevor de Clercq and David Temperley's article "A corpus analysis of rock harmony" and is discussed in the following sections.[3]

While n-gram or Markov models are simple models which are able to capture the occurrence of various surface level harmonic progressions in a corpus of music, they are unable to capture the nonlocal and hierarchical dependencies of harmonic progressions. Many theorists including Rohrmeier and Patel have noted that musical syntax, and in particular harmonic syntax, is similar to the syntax in language in that it is fundamentally a hierarchical phenomenon.[4][5] As a result, music theorists have implemented more complex models in order to account for the multi-level complexity of the underlying harmonic syntax that they are modeling. One example of a model that theorists have used to account for the nonlocal and hierarchical dependencies of harmonic syntax is a more sophisticated version of a Markov model called a Hidden Markov Model.

1.2 Hidden Markov models (HMMs):

Hidden Markov models (HMMs) are similar to Markov or n-grams models but use latent or hidden states in order to account for some of the nonlocal or hierarchical dependencies in the sequences that they model. In this thesis, the latent or hidden states that the HMMs use to model a sequence of chord progressions will correspond to unknown syntactic chord categories like "Tonic" or "Dominant". However, while music theorists traditionally assign specific chords to

[3] David Temperley and Trevor De Clercq, "A Corpus Analysis of Rock Harmony," *Popular Music,* vol. 30, no. 1 (2011): 64

[4] Martin Rohrmeier, "Towards a Generative Syntax of Tonal Harmony," *Journal of Mathematics and Music,* vol. 5, no. 1 (2011): 35

[5] Aniruddh D. Patel, "Syntax," in *Music, Language, and the Brain*, (Oxford University Press, 2007), 239-298.

syntactic categories like "Tonic" and "Dominant", using a sequence of chord progressions, HMMs will infer syntactic categories that best fit the sequences of chords from the data. For instance, after running a HMM on a sequence of chords from some genre of music, the HMM may find that the chords I, bVII and bII belong together in one syntactic category which could play the role of a traditional "Tonic" category in classical music. In addition to inferring syntactic categories from the sequence of chord progressions, the HMM will also determine the likelihood of moving between different syntactic categories or staying in the same category.

One example of a HMM that was used to model the harmonic syntax of popular music from the McGill Billboard corpus was given in "Generative Statistical Models with Self-Emergent Grammar of Chord Sequences". In the paper, Tsushima et al. implemented a HMM on the McGill Billboard Corpus and derived four hidden states – "Tonic", "Subdominant", "Dominant" and "Others", the output probabilities (the red bars in Figure 1) and the transition probabilities (the probabilities associated to the blue arrows in Figure 2). It is important to note that before running their HMM, Tsushima et al. first transposed the keys of all of works in the corpus to C major.

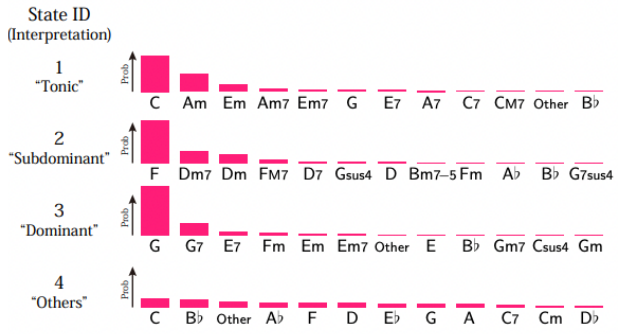

Figure 1: Composition of the Four Hidden States from Tsushima et al. [6]

[6] Tsushima et al, "Generative Statistical Models with Self-Emergent Grammar of Chords Sequences", *Journal of New Music Research,* vol. 47, no. 3 (2018): 17.

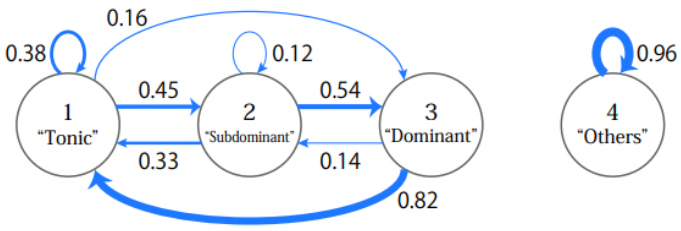

Figure 2: Diagram of the Transition Probabilities from Tsushima et al. [7]

In looking at the composition of the hidden states in Figure 1, it is clear how the state with a high percentage of I chords is "Tonic", IV chords is "Subdominant" and V chords is "Dominant". The fact that the HMM tells us that vi and iii chords function like "Tonic" chords in this corpus and that ii7 and IV7 chords function as "Subdominant" chords reflects how our harmonic system consists of chords built of stacked thirds which causes chords with roots a third apart to share some scale degrees and behave in a similar way. It is also interesting how 82% of the time, chords in the "Dominant" category move to chords in the "Tonic" but how chords in the "Tonic" category only move to chords in the "Dominant" category 16% of the time. This strong unidirectional tendency is an aspect of the harmonic syntax which I will return to when discussing the harmonic syntax of rock music in subsequent chapters. It is important to note that while these observations might be valid for the harmonic syntax of songs in the McGill Billboard corpus, they are not necessarily valid for the harmonic syntax of music from that time period in general. Since the McGill Billboard corpus contains harmonic information for 730 songs taken from the Billboard Hot 100 Singles charts from August 1958 through November 1991, the dataset represents a sample of songs which were popular during this period of time. As a result, the corpus does not necessarily provide a full picture of the popular music in this time period but

[7] Ibid.

rather an approximation which also does not provide equal representation of all artists from different backgrounds in this time period.

1.3 Probabilistic Context-free Grammars

Having introduced HMMs and given an example, I will now introduce another more sophisticated model that music theorists have used to study the concept of harmonic syntax called context-free grammars and their probabilistic counterparts called probabilistic context-free grammars. Since these models have even more explanatory power than Markov models and HMMs, they will be able to model the harmonic syntax of the chord sequences in the corpus in even more depth.

Probabilistic context-free grammars are closely related to formal grammars. A formal grammar consists of a set of rules describing how to form strings from an alphabet and can be thought of as a formalization and generalization of modern languages. Here, an alphabet refers to some set of symbols (which in the case of this thesis are chords) and strings refer to lists of symbols from the alphabet (which in the case of this thesis are harmonic progressions). In the appendix of their chapter "Musical Syntax I: Theoretical Perspectives", Rohrmeier and Pearce summarize the notion of a formal grammar as follows: "A formal grammar consists of a set of nonterminal symbols (variables), terminal symbols (elements of the surface), production rules, and a starting symbol to derive productions."[8] Consider the following example of a formal

[8] Martin Rohrmeier and Marcus Pearce, "MusicalSyntax I: Theoretical Perspectives," in *Springer Handbook of Systematic Musicology*, ed. Rolf Baber (Berlin: Springer, 2018), 483.

grammar which is a toy subset of the English language show below in Figure 3:

Terminals:

{generate, hate, great, green, ideas, linguists}

Nonterminals:

{S, NP, VP, N, V, Adj}

Production rules:

S → NP VP

NP → Adj NP

NP → N

VP → V NP

VP → V

N → ideas

N → linguists

V → generate

V → hate

Adj → great

Adj → green

and start symbol S.

An example derivation in this formal language is:

S → NP VP → Adj NP VP → Adj N VP → Adj N V NP → Adj N V Adj NP → Adj N V Adj Adj NP → Adj N V Adj Adj N → great N V Adj Adj N → great linguists V Adj Adj N → great linguists generate Adj Adj N → great linguists generate great Adj N → great linguists generate great green N → great linguists generate great green ideas.

Figure 3: An Example of a Formal Grammar

Here, the nonterminal symbols S, NP, VP, N, V and Adj stand for subject, noun phrase, verb phrase, noun, verb and adjective. Since the sentence “great linguists generate great green ideas” was derived from the terminal and nonterminal by applying the production rules, the sentence is syntactically correct in this formal grammar. On the other hand, the sentence “ideas ideas great hate” is syntactically incorrect since it cannot be derived from the terminals and nonterminals through the application of the production rules.

In order to understand probabilistic context-free grammars, it is instructive to mention context-free grammars. Context-free grammars are a special class of formal grammars in which words are generated with the application of recursive production rules. An example of a context-free language is the set of well-formed parentheses, { (), ()(), (()), (())(),…}.This context-free language has terminal symbols “(“ and “)”, start symbol “S” and is generated by the production rules:

S --> SS

S --> (S)

S --> ()

This formal grammar is an example of a context-free grammar because it contains production rules which expand the non-terminal symbol S into a string that also contains the non-terminal symbol S. If this formal grammar had no production rules that had the non-terminal S on the right-hand side of their production rules, then it would not be a context-free grammar. The fact that this formal grammar contains recursive production rules is ultimately the property that allows grammars of this type to capture the hierarchical relationship between the terminal elements in the grammar. In the PCFG used in this thesis, the terminal elements of the context-free grammar will be chords.

While a context-free grammar of a set of symbols contains a set of production rules, computational linguists have extended this idea to something called a probabilistic context-free grammar (PCFG) which assigns each production rule in the grammar a probability based on how often the production rule is used in the grammar describing the language. For a collection of sentences in a language or a sequence of chords, a probabilistic context-free grammar assigns probabilities to each production rule. Thus, the probabilities of the production rules are parameters of the model and are determined through an optimization algorithm on the dataset. In this thesis, a PCFG will provide a set of rewrite rules and their associated probabilities for the allowable ways in which a chord can be expanded or substituted for other chords. As a result of these rewrite rules, this model will be able to model the hierarchical relationships of the harmonic syntax of chord sequences in the corpus. To illustrate how the harmonic syntax governing a sequence of chords could be hierarchical in nature, consider the following analysis of the A section of the Jazz-standard "Afternoon in Paris" taken from "A Generalized Parsing Framework for Generative Models of Harmonic Syntax" by Daniel Harasim, Martin Rohrmeier and Timothy J. O'Donnell as reproduced in Figure 4 below:

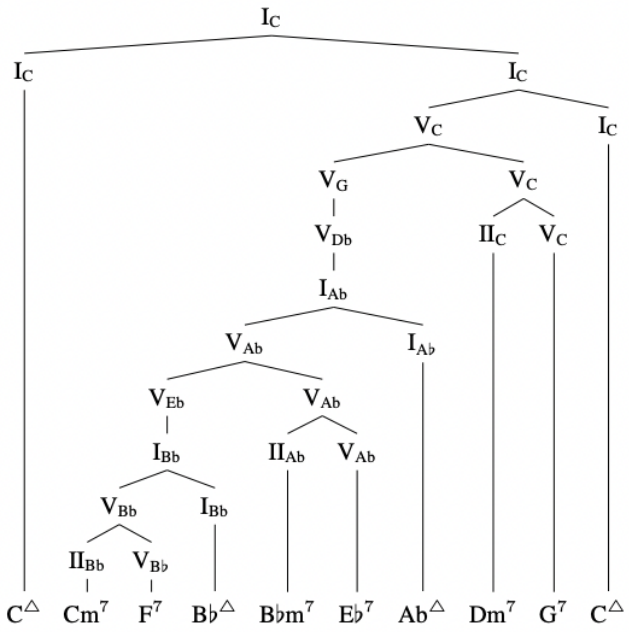

Figure 4: A Hierarchical Analysis of the A section of the Jazz-standard "Afternoon in Paris"[9]

Given this sequence of chords, the authors used a context-free grammar to model the hierarchical relationship of the harmonic syntax of this sequence of chords and to create a diagram called a parse tree which is shown in Figure 4. This parse tree models the hierarchical relationships in this sequence of chords. In Figure 4, the Subdominant, dominant, and tonic phrases are denoted by the scale degrees II, V, and I, respectively and the subscripts denote the key in which each of these progressions is in. For the sequences of chord progressions in the corpus, I will implement a PCFG which will consist of a collection of rewrite rules and the associated probabilities of each rewrite rule occurring (which is based on the sequence of chord progressions in the corpus). For instance, a few of the rewrite rules and their associated

[9] Daniel Harasim, Martin Rohrmeier and Timothy J. O'Donnell, "A Generalized Parsing Framework for Generative Models of Harmonic Syntax," *19th International Society of Music Information Retrieval Conference* (Paris, 2018): 152.

probabilities for a PCFG for the sequence of chords at the lowest level in Figure 4 could be of the form:

I_C	-->	I_C I_C	(10%)
I_C	-->	V_C I_C	(20%)
I_C	-->	C^{Δ}	(70%)
V_{Bb}	-->	II_{Bb} V_{Bb}	(20%)
V_{Bb}	-->	F^7	(80%)
V_{Ab}	-->	V_{Eb} V_{Ab}	(20%)
V_{Ab}	-->	II_{Ab} V_{Ab}	(20%)
V_{Ab}	-->	Eb^7	(60%)

Figure 5: Hypothetical Rewrite Rules and their Associated Probabilities for a PCFG Modeling the Chord Progression in Figure 3

In this PCFG, the rewrite rules for the I_C says that 10% of the time I_C can be rewritten as I_C followed by I_C, 20% it is rewritten as V_C followed by I_C and 70% of the time it is written as C^{Δ}. Similarly, in this probabilistic context-free grammar the rewrite rules for the V_{Bb} says that 20% of the time V_{Bb} can be rewritten as II_{Bb} followed by V_{Bb} and 80% it is rewritten as F^7. I will obtain these rewrite rules for a PCFG and their associated probabilities from the input sequence of chords by training the model. Ultimately, these rewrite rules and their associated probabilities will provide a detailed model of the harmonic syntax in the rock corpus which is able to account for some of the hierarchical dependencies in the harmonic syntax.

One connection between HMM and PCFGs is that the algorithm used to determine the production probabilities in the PCFG called the "inside-outside algorithm" is a generalization of the "forward-backward algorithm" which is used to compute the posterior marginals of all the hidden states given a sequence of observations/emissions in a Hidden Markov model.[10] For a more detailed discussion of the algorithms used to train the HMM and PCFG I refer the interested reader to the appendix.

[10] Daniel Jurafsky and James Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2023), 179.

2. Introduction to White and Quinn's Article and de Clercq and Temperley Rock Corpus

In this chapter I will introduce and discuss some recent work done by Christopher White and Ian Quinn in which they applied composite Hidden Markov Models to the Kostka-Payne, McGill Billboard and Bach Chorale corpora in order to question the generalizability of the traditional three-function model for harmonic syntax in classical music.

2.1 Kostka-Payne Corpus

The first corpus that White and Quinn study is the Kostka-Payne corpus which is a corpus of common-practice music excerpts. The data from the corpus comes from the workbook accompanying Stefan Kostka and Dorothy Payne's theory textbook Tonal Harmony, 3rd edition (McGraw-Hill, 1995). The corpus consists of 46 excerpts from various Beethoven string quartets, Bach chorales and Chopin Mazurkas. The analyses were done by the authors and are in conventional Roman numeral notation. For the Kostka-Payne corpus, White and Quinn first created a three-state composite HMM to see if the model would produce the traditional tonic, dominant and sub/predominant functions. After training the model, they produced the two models which are reproduced in Figure 6 below. Here, higher-transition probabilities are represented by thicker arrows and the composition of each hidden state or syntactic category is represented by the pie charts where the numbers represent roots of chords (which could be either major or minor).

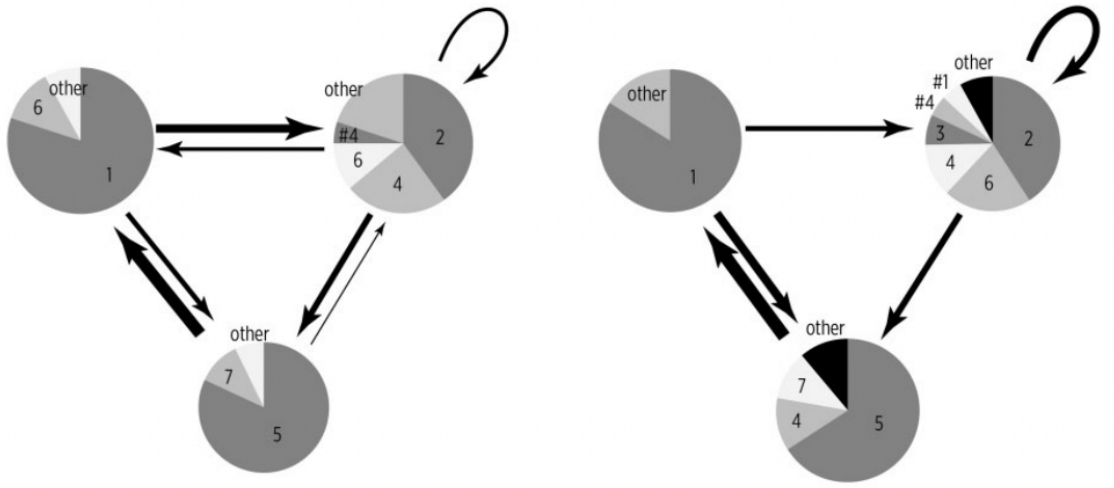

Figure 6: White and Quinn's Example 6: Composite HMMs with Three Hidden States for the Kostka-Payne Corpus[11]

In the model on the left, it is apparent that moving clockwise from the top left diagram, the chord categories could correspond to tonic, predominant and dominant functions. White and Quinn also note how none of the arrows are particularly thick which suggests that no motion between syntactic categories is entirely unidirectional. In the model on the right, White and Quinn note how there are some unidirectional arrows and how the lexical probabilities are more uniformly distributed. In analyzing different three-state models for the Kostka-Payne corpus, White and Quinn created a composite HMM based on 300 different HMMs, in which they used a different expectation-maximization algorithm to train their composite model known as k-medoids. For more details see their "Chord Context and Harmonic Function in Tonal Music".[12] Ultimately, they found that their composite three hidden state model was incoherent and found a coherent composite four hidden state model which is reproduced in Figure 7 below.

[11] Christopher White and Ian Quinn, "Chord Context and Harmonic Function in Tonal Music" *Journal of New Music Research*, vol. 40, no. 2, (2018): 321.

[12] Ibid, 355A.

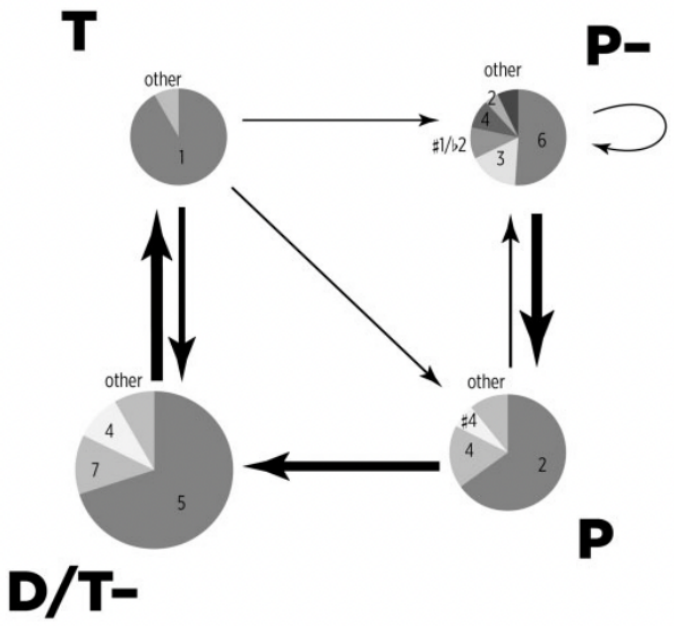

Figure 7: White and Quinn's Example 7: Composite HMMs with Three Hidden States for the Kostka-Payne Corpus[13]

Here, the states from the upper left moving clockwise are Tonic, Pre-predominant, Pre-dominant and Dominant/Pretonic which are denoted as T, P^-, P and D/T^-. White and Quinn go on to observe how their four-state model reflects Kostka and Payne's notion of the "three common functions" of a IV chord: proceeding to I, proceeding to V, or proceeding to ii, which will in turn proceed to V.[14] They also note how their four-state model is very similar to a model of harmonic function created by Allen Irvine McHose in his 1947 textbook "The Contrapuntal-Hrmonic Technique of the 18th Century" that was also based on a corpus which was a novel concept in its time. McHose's Four-Chord Classification is reproduced below.

[13] Ibid, 322.

[14] Ibid, 320.

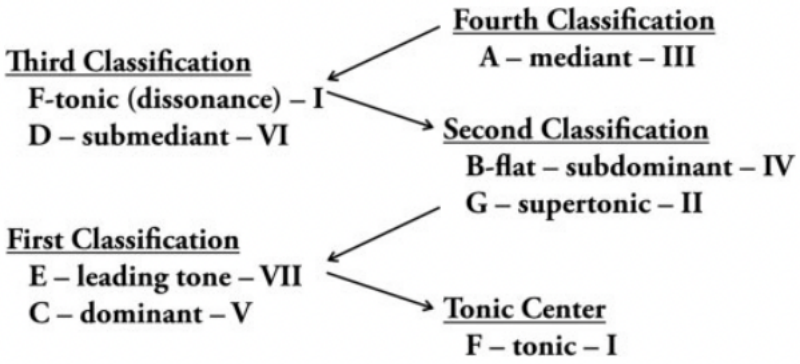

Figure 8: Model of Harmonic Function Created by Allen Irvine McHose[15]

Despite their four state model's similarities with other models of harmonic function, they note how their model is peculiar in the way in which it conflates dominant and subdominant functions and derives a pretonic syntactic category from the Kostka-Payne corpus which suggests how dominant and subdominant functions tend to occur in the same contexts in the corpus.

2.2 McGill Billboard Corpus

Following their analysis of the Kostka-Payne corpus, White and Quinn constructed several models for the McGill Billboard Corpus which contains harmonic progressions for 730 songs taken from the Billboard Hot 100 Singles charts from August 1958 through November 1991. In particular, they found that their best composite HMM was an eight-state HMM which had the states T, T^+, S, S^+, with two peripheral pairs, P/Q and X/W. Here S and T are subdominant and tonic categories and T^+ and S^+ are post-tonic and post-subdominant categories which contain chords that follow chords in the tonic and subdominant categories. In discussing

[15] Allen Irvine McHose, *The Contrapuntal Harmonic Technique of the 18th Century* (F.S. Crofts & Company, 1947), 221.

their model, they note how the main circuit of the T, T^+, S, S^+ of states or syntactic categories account for the majority of the chord transitions (67.2%) while the X/W accounts for 22.7%, P/Q accounts for 2.3%, and the remaining transitions are accounted for by improbable (but possible) moves between these three different circuits. A transition diagram of their eight-state HMM of the McGill Billboard Corpus is reproduced in Figure 9 below:

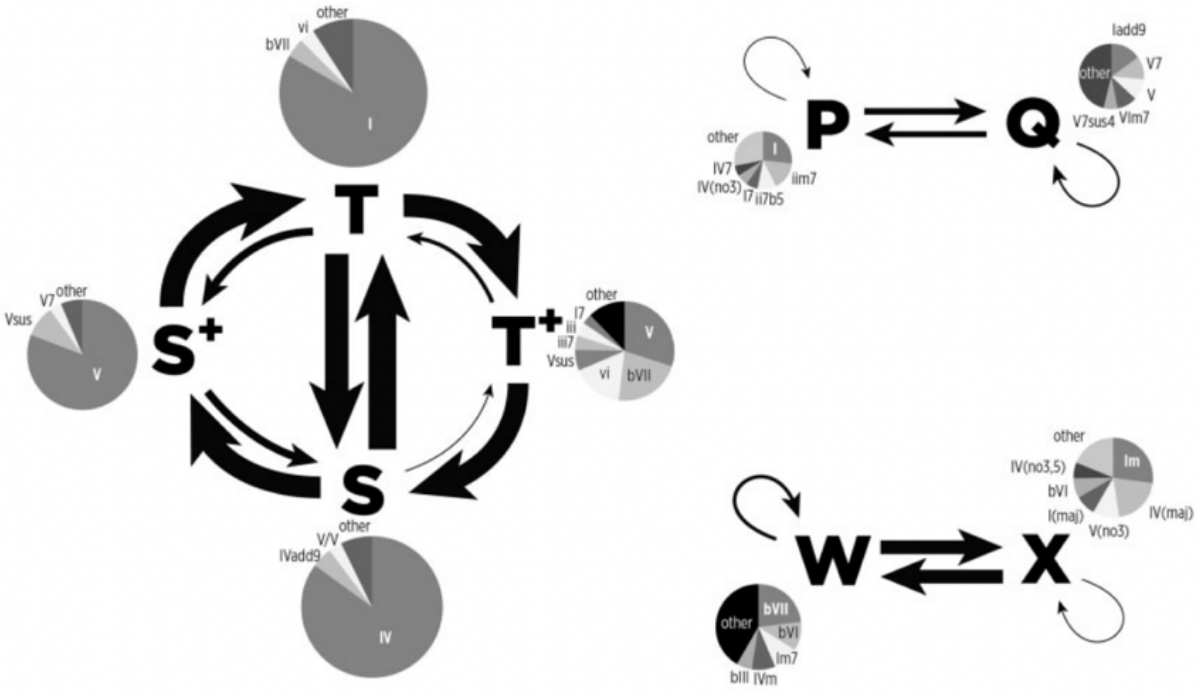

Figure 9: A Reproduction of Example 10 from White and Quinn's Article[16]

In discussing their model, they note that the main circuit of states, T, T^+, S, S^+, contains two primary poles, T and S, which are very similar to the traditional tonic and subdominant functions, and in which the most frequently represented chord in T is I and how S is most frequently represented by IV chords. They note how T^+ manifests itself as a number of chords

[16] Ibid, 324.

such as V, bVII, vi, etc… and how S^+ manifests itself tends to manifest itself as chords that are closely related to V.

2.3 Bach Chorale Corpus

Finally, White and Quinn also used HMMs to model the harmonic syntax of the Bach Chorale corpus which consists of the 370 Bach chorales. Instead of providing their model with roman numerals, they provided it with 35,139 salami slices which were taken from chorales in the corpus which were later mapped onto chords. From this data, they found that their best composite HMMs had 3 and 13 hidden states. They found that the three-state model closely resembled the traditional I-IV-V harmonic function model. The transition diagram for their three-state model is reproduced in Figure 10 below:

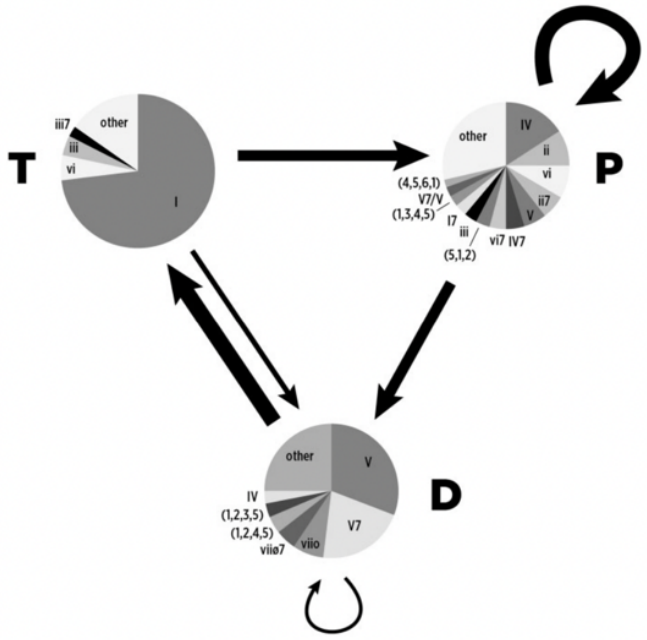

Figure 10: White and Quinn's Composite HMM with Three Hidden States for the Bach Chorale Corpus[17]

[17] Ibid, 328.

White and Quinn then introduced their thirteen-state composite HMM. In discussing this model, they note how the advantage of the 13-state model is that it can capture the traditional I-IV-V tonal relationships while also adding some new pathways and detours which are unique to the harmonic syntax of the Bach Chorale Corpus. In addition to the traditional tonic, subdominant and dominant functional categories, their thirteen-state model adds a T^x (tonic expansion) category which contains mostly I triads along with several vi and iii chords. They note how this category allows the model to capture various passing and neighbor motions in the corpus. They also include a category denoted D^+ (late dominants) which is comprised of most V^7 chords as well as categories denoted T^+ (late tonics) which contains chords such as I4/2 and p (weak predominant) and p^x (predominant expansion).[18] In this context, they define late tonics as a passing function that progresses from tonic to weak predominant and late dominants as a passing function between dominant and tonic. The remaining functional categories are summarized in their example 23 on pg. 330 which is reproduced in Figure 11 below along with the transition diagram of their HMM model:

[18] Ibid, 329.

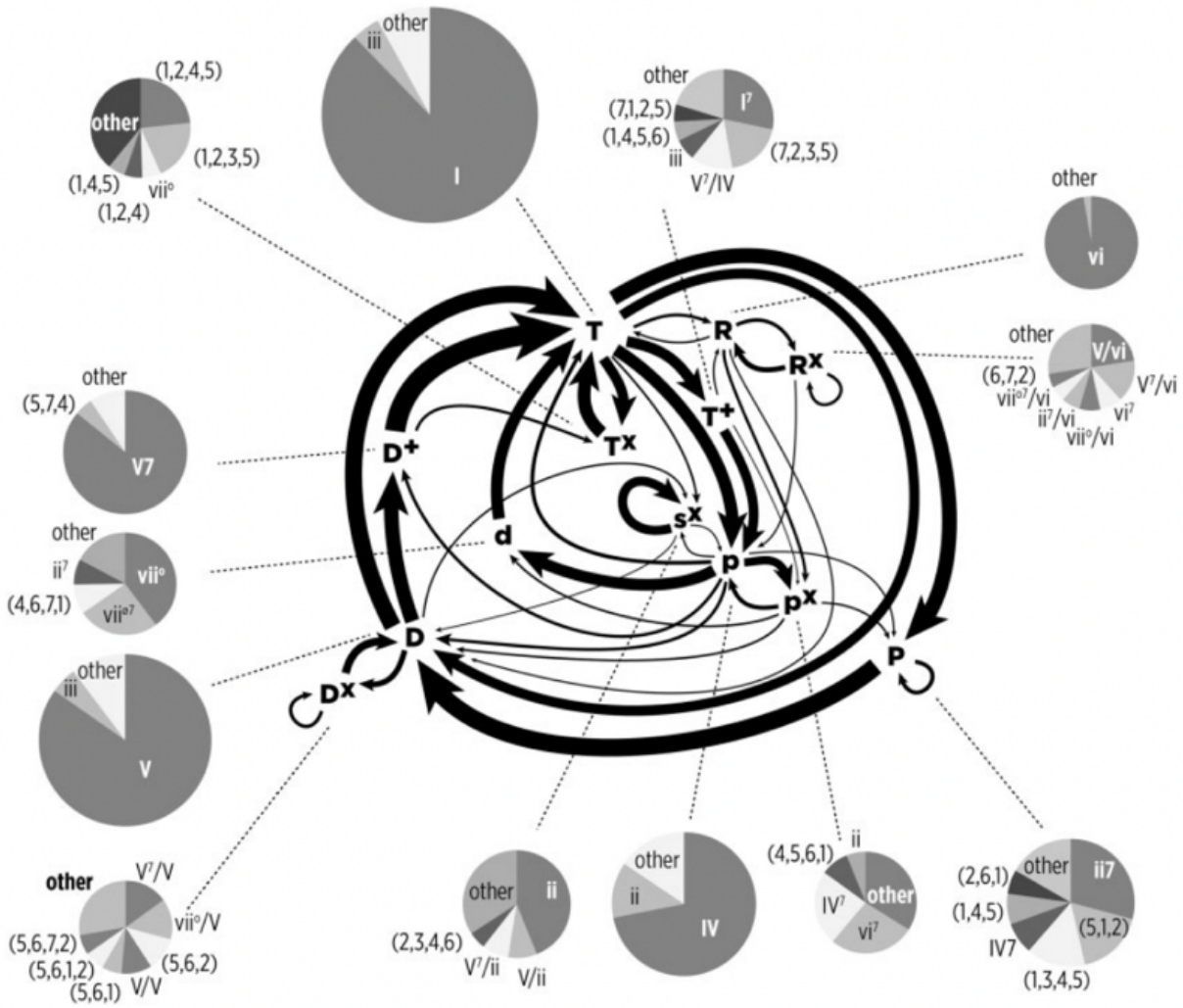

Figure 11: White and Quinn's Composite HMM with Thirteen Hidden States for the Bach Chorale Corpus[19]

[19] Ibid, 330.

function	*characteristic chords*	*behavior*	*corpus probability*
T	I, iii, vi	Comes from D-functions, goes many places. Expanded by T^x.	19.0%
D	V, iii	Comes from P-functions, T, and R, goes to D^+ and T. Expanded by D^x.	18.4%
D^+	V^7, all its trichord subsets	Comes from p and D. Goes to T.	13.2%
P	ii^7, $\{\hat{1}, \hat{2}, \hat{5}\}$, $I^{add\ 4th}$	Comes from other P-functions and T. Goes to D. Expanded by itself.	9.0%
s^x	ii, $V^{(7)}$/ii	Comes from D and T. Goes to p. Expanded by itself.	8.4%
D^x	V^7/V, all its trichord subsets	Comes from and goes to D. Expanded by itself.	7.8%
p	IV, ii	Comes from T-functions and s^x. Goes to D-functions. Expanded by P^x.	6.0%
R^x	V^7/vi, all its trichord subsets	Comes from and goes to R. Expanded by itself.	5.9%
T^x	$\{\hat{1}, \hat{2}, \hat{4}, \hat{5}\}$, $\{\hat{1}, \hat{2}, \hat{3}, \hat{5}\}$, $\{\hat{1}, \hat{2}, \hat{4}\}$, vii	Comes from and goes to T.	4.9%
d	vii°, $vii^{\circ 7}$	Comes from p and p^x, goes to T.	3.6%
p^x	vi^7, IV^7	Comes from and goes to p, but also can come from T^+ and R and go to P or D-functions.	2.0%
R	vi	Comes from and goes to T and R, but can also go to P^x and D.	1.3%
T^+	I^7, $\{\hat{2}, \hat{3}, \hat{5}, \hat{7}\}$, I^{b7}	Comes from T, and goes to p or p^x.	0.7%

EXAMPLE 23. *A summary of the thirteen functions*

Figure 12: A Summary of the 13 Hidden States (Quinn and White's Example 23)[20]

[20] Ibid, pg. 331.

2.4 White and Quinn's Discussion of Rock Syntax

Having introduced and briefly summarized White and Quinn's construction of several composite HMMs for the Kostka-Payne, McGill Billboard and Bach Chorale corpora, I will now turn our attention to rock music and summarize White and Quinn's discussion of rock syntax. While discussing the difference between the harmonic syntax of the McGill Billboard corpus and the syntax of other genres of music White and Quinn noted:

> "Temperley and DeClercq, in a corpus study of rock harmony, emphasize many aspects of this difference: rock harmony does not have strong unidirectional tendencies (e.g., V progresses to IV as much as IV progresses to V), and, in many cases, IV (rather than V) functions as the primary nontonic triad. On the other hand, several analysts have attempted to theorize pop/rock harmonic syntax as an extension of common-practice norms. Nicole Biamonte and Chris Doll, for instance, argue for including modal harmonies into functional models, with bVII functioning as dominant (Doll's 'rogue dominant') or as IV/IV (Biamonte's 'Double Plagal' progression). Going even further, Drew Nobile entirely dissociates traditional harmonic functions from the scale-degree content of chords. In Nobile's formalization, almost any chord can function as a tonic, dominant, or predominant: 'a chord's function is given more by formal considerations—i.e., what role it plays within the form—than by its internal structure or any specific voice-leading motion.' Nobile allows for predominant V chords, dominant IV chords, and so on."[21]

From this, it is clear that there are several notions which theorists have posited which characterize rock music such as the lack of strong unidirectional tendencies, IV rather than V

[21] (White and Quinn *Chord context and harmonic function in tonal music, pg. 322*)

functions as primary nontonic triad and bVII functioning as a dominant. We will now look more closely at De Clercq and Temperley's article and examine how they arrived at their conclusions.

3. Introduction to De Clercq and Temperley's Rock Corpus

In this chapter I will provide an overview of the article by De Clercq and Temperley. In "A corpus analysis of rock harmony" they built a corpus of 100 rock songs that were selected from the Rolling Stone magazine's list of the '500 Greatest Songs of All Time".[22] Their corpus consists of harmonic analyses for all of the songs done by both authors.

3.1 Discussion of de Clercq and Temperley's Corpus

In constructing the corpus, De Clercq and Temperley use a subset of the *Rolling Stone* magazine's list of the "500 Greatest Songs of All Time" which they note is one of the few lists that contains general 'rock' music without stylistic modifiers like 'hard rock'. However, it is important to note how the corpus includes blues, country, R&B and hip hop as well as rock and that the majority of the songs in the corpus are from the late 1960s and 1970s and are from artists who are both white and male. The list that De Clercq and Temperley chose to pick songs from was a compilation of the top 50 rock songs chosen by 172 'rock stars and leading authorities.[23] However, it is important to note the lack of transparency in the use of unnamed and hand-picked industry insiders to determine the composition of the corpus is in itself methodologically problematic. In addition, while the corpus that De Clercq and Temperley constructed contains songs that were perceived by the unnamed industry insiders to be the popular or the greatest rock songs, there is no objective measure for the popularity of the songs in the corpus. Therefore, White and Quinn's conclusions about the harmonic syntax of rock and are not necessarily representative of the harmonic syntax of the genre as a whole.

---

[22] David Temperley and Trevor De Clercq, "A Corpus Analysis of Rock Harmony," *Popular Music,* vol. 30, no. 1 (2011): 47.

[23] Ibid, 51.

After determining the list of songs to include in the corpus, De Clercq and Temperley both independently conducted harmonic analyses and constructed a dataset that contains both of their harmonic analyses in order to reduce the effect of idiosyncrasies. After doing this, they compared harmonic analyses and found that their harmonic analyses agreed completely on 39 of the 100 songs and that all of their harmonic analyses agreed on the relative root of each song. They also found that one song, Public Enemy's 'Bring the Noise" did not contain any triadic harmony, so they decided to remove it from the list of 100 songs. Each of the harmonic analyses in their corpus of 99 songs identifies the key center and contained a sequence of roman numerals. The text file of Temperley's analysis of Nirvana's "Smells like Teen Spirit" is shown below:

```
[F] i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV |
bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI |
i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII
bVI | i IV | bIII bVI | i IV | bIII bVI | i bII | i64 IV | i bII | i64 IV | i IV | bIII bVI | i IV | bIII bVI | i IV |
bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI |
i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII
bVI | i bII | i64 IV | i bII | i64 IV | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV |
bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI |
i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII
bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV
| bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i IV | bIII bVI | i |
```

Figure 13: Example of the Format of a Song in De Clercq and Temperley's Corpus[24]

3.2 Discussion of de Clercq and Temperley's Analysis of their Rock Corpus

In analyzing the harmonies in the Rock corpus, De Clercq and Temperley first examined the overall distribution of chromatic relative roots. In order to do this, they found the sheer number of occurrences of each root and found the proportion of that number to the total number of roots. From this, they found that major and minor chords built on the chromatic roots I, IV, V, bVII and then VI were most frequently used. Next, they considered chord transitions within a

[24] "Harmonic Analyses," A Corpus Study of Rock Music, accessed November 25, 2023, http://rockcorpus.midside.com/harmonic_analyses.html.

single key (they did not record transitions from one key to another). From this, they found that the most frequent chords to precede the tonic were IV, V and then bVII and found that the 'pre-tonic' distribution (chord approaching I) and the 'post-tonic' distribution (chords approached from I) as well as the overall distribution of chords roots (excluding the tonic) were similar. As a result, they wrote, "In light of this data, one might conclude that rock is not governed by rules of 'progression' at all; rather, there is simply an overall hierarchy of preference for certain harmonies over others, regardless of context."[25] For reference, I have reproduced their table of chord transition counts in Figure 14 below:

Ant \ Cons	I	bII	II	bIII	III	IV	#IV	V	bVI	VI	bVII	VII
I	0	25	132	94	44	1052	2	710	104	302	470	16
bII	31	0	0	0	2	0	0	0	0	0	0	12
II	120	1	0	2	20	58	0	97	0	24	10	0
bIII	50	6	6	0	0	64	2	2	67	0	41	0
III	16	0	39	0	0	46	0	6	0	60	3	4
IV	1,162	14	30	98	45	0	4	514	57	72	90	4
#IV	7	0	0	6	0	10	0	0	0	0	0	0
V	788	0	36	6	17	392	4	0	6	191	48	0
bVI	208	0	1	20	0	22	6	22	0	10	78	0
VI	144	0	87	0	32	260	0	124	21	0	3	0
bVII	386	0	0	11	2	188	2	26	114	6	0	0
VII	18	0	0	0	12	0	4	0	0	3	0	0

Figure 14: De Clercq and Temperley's Table of Chord Transitions in the Rock corpus[26]

Following this, they considered trigrams leading up to the tonic and found that the trigram IV-V-I was the most frequently occurring trigram, which occurred 352 times, and was followed by V-

[25] David Temperley and Trevor De Clercq, "A Corpus Analysis of Rock Harmony," *Popular Music,* vol. 30, no. 1 (2011): 61.

[26] Ibid.

IV-I which occurred 292 times. Their table of the most frequent trigram occurrences is also reproduced in Figure 15 below:

Trigram	Instances
IV V I	352
V IV I	292
bVII IV I	146
VI IV I	126
bVII bVI I	103
bIII bVI I	66
II V I	63
bVI bVII I	60
V VI I	42
IV bVII I	39

Figure 15: De Clercq and Temperley's Table of Trigrams[27]

They also considered the co-occurrence of chords in the corpus. In order to measure this, they created a 99-dimensional chord vector for each of the 12 relative roots with zeros and ones for whether a particular chord is present each of 99 songs in the corpus. They then measured the correlations between each root with the other 11 and produced a 12x12 correlation matrix and highlighted values that were .35 or above which they deemed to be significant. From this they found that the roots bVII, bIII and bVI tended to co-occur as the three pairs were correlated. In finding this, they suggested that these co-occurrences offered some evidence as to some kind of modal organization that could be thought of as similar to the major and natural minor modes in the traditional common practice.

In summary, De Clercq and Temperley found a greater frequency of IV and V as frequent pre-tonic chords from their analysis of chord-to-chord transitions and a high frequency of VI and bVII chords in the overall frequency of chords in the corpus. They also found that the most

[27] Ibid, 63.

frequently occurring asymmetrical root motions characteristic of the common-practice period were noticeably absent in rock. They also observed that the harmonies in the corpus were overwhelmingly in root position, major triads were more frequently occurring than minor triads, and that root motion by ascending or descending fourth was most common. In analyzing trends across the decades, they also found that the harmony of the songs in the 1950s was mostly confined to the use of I, IV and V whereas the distribution of chords used between the 1960's and 2000 was broader.

3.3 Suggestions for further directions

In the final sections of their article, De Clerc and Temperley suggested several possible directions for further research. The first direction they suggested was to gather more data from Rolling Stone list and conduct more harmonic analyses. They noted that doing this would help to determine larger-level harmonic patterns or more hierarchical relationships in the corpus which would involve constructing a new model. They also note that other scholars have suggested that "rock music reflects a strong preference for the placement of tonic harmony in metrically strong positions and that this is an important cue for tonal orientation" and that the corpus could be used to analyze this because it includes metric structure as well as the chords. [28] [29]

In the final section of their article, De Clercq and Temperley also note that one of the possible criticisms of their project is that they treated the harmonic structure of the songs in the corpus as "flat" or viewed the harmonic structure of the works only as a one-level sequence of

[28] Ibid, 68.

[29] Walter Everett, "Making Sense of Rock's Tonal Systems," *Music Theory Online,* vol. 10, no. 4 (December 2004): 10.

harmonies.[30] In discussing this, they note how many other authors such as (Brown 1997; Burns 2008; Everett 2008) analyze rock in a hierarchical (e.g. Schenkerian) manner and distinguish structural from elaborative harmonies.[31] While they justify their approach by noting that their "single-level" harmonic analysis is relatively immune to differences in opinion (which could arise from the high degree of subjectivity that could arise in reductive analysis) they note that a hierarchical approach to understanding the harmony of the songs in the corpus could offer new insights. My implementation of self-emergent midden Markov models and probabilistic context-free grammar models which are discussed in Chapter 5 will attempt to account for the hierarchical nature of the harmony in the Expanded Rock Corpus.

[30]David Temperley and Trevor De Clercq, "A Corpus Analysis of Rock Harmony," *Popular Music,* vol. 30, no. 1 (2011): 68.

[31] Ibid.

4. The Expanded Rock Corpus and Patterns in the Data

In this chapter, I will discuss how I expanded De Clercq and Temperley's rock corpus. The rock corpus that De Clercq and Temperley studied in their 2011 article consisted of the top 20 songs in each of the five decades from the Rolling Stone magazine's list of "500 Greatest Songs of All Time". Since the publication of their article, De Clercq and Temperley added the next highest ranked songs from the RS 500 list and created a corpus of harmonic analyses of 200 songs.[32] After determining which songs of the RS 500 De Clercq and Temperley did not include in their corpus of 200 songs, I produced harmonic analyses for an additional 239 songs which together with De Clercq and Temperley's harmonic analyses of 200 songs make up the 439 songs of the Expanded Rock Corpus. These additional 239 songs that I added were taken from *Rolling Stone's* "500 Greatest Songs of All Time" based on De Clercq and Temperley's 2021 revised list.[33] Since the Expanded Rock Corpus has more than four times as many songs as the original corpus which De Clercq and Temperley studied in their 2011 article, I will replicate the analyses that they performed and compare my findings from the Expanded Rock Corpus with the results from the 2011 article.

4.1 Harmonic Analyses of the Songs in the Expanded Rock Corpus

While De Clercq and Temperley conducted harmonic analyses by ear and recorded metrical data associated with the harmonies, due to time constraints I used pre-existing guitar tabs from "https://www.ultimate-guitar.com/" which provided a list of chords for each song which I converted into harmonic analysis by hand. The main drawback of this method is that

[32] David Temperley and Trevor De Clercq, ""Statistical Analysis of Harmony and Melody in Rock Music," *Journal of New Music Research,* vol. 42, no. 3 (2013): 187

[33] "The Corpus," A Corpus Study of Rock Music, accessed June 3, 2023, http://rockcorpus.midside.com/.

there are often multiple existing guitar tabs for each song, some of which were found to be inaccurate. To mitigate this issue, I used the official guitar tabs from a premium subscription to "Ultimate Guitar Com."[34] These guitar tabs were created by the UG team which was formed in 2016 and consists of more than 30 people each of which has over ten years of guitar experience and half of which have music degrees. After using one of the UG team's official tabs I also verified that the tab was an accurate or close approximation of the chords of the song. In the instances in which the premium version of the guitar tab was not available I used the guitar tab with the highest user rating. While De Clercq and Temperley recorded the metrical placement of harmonies, I did not and also only recorded chord changes meaning that my data does not include repeated chords. While there are likely errors in the corpus as a result of inaccurate guitar tabs or human error, I decided that in creating a large enough corpus these errors would have a minimal effect on underlying harmonic trends in the corpus.

4.2 Patterns in the Expanded Rock Corpus

Now, I will replicate the basic statistical analysis that De Clercq and Temperley performed on their rock corpus but on the Expanded Rock Corpus and analyze the distribution of chromatic relative roots, chord transitions, trigrams, the proportions of chromatic roots in each decade, and other patterns. After loading the corpus data, which was compiled in an excel spreadsheet, into the programming language Python, I calculated the frequency of the top 20 chords in the corpus, which is summarized in the histogram below.

[34] "Tabs," Ultimate Guitar Com, accessed June 3, 2023, https://www.ultimate-guitar.com/explore

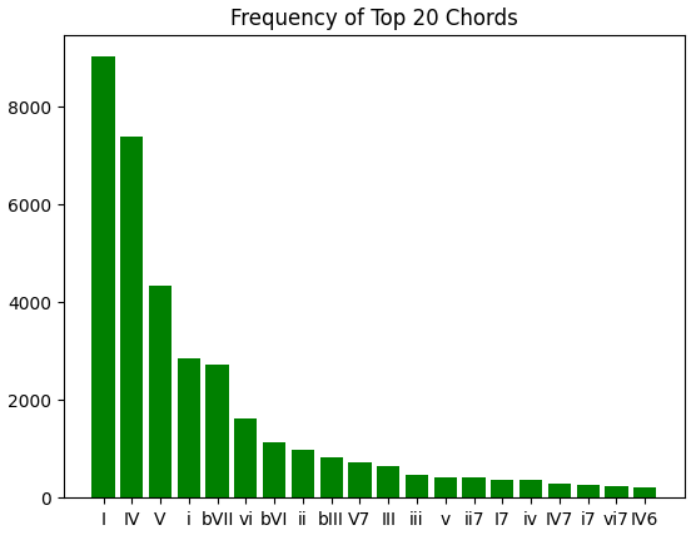

Figure 16: Frequency of the Top 20 Chords in the Expanded Rock Corpus

It is interesting to note how the IV chord occurs a little under twice as frequently as the V chord. This confirms De Clercq and Temperley's previous observation that the IV chord is prevalent in rock harmony. Following the I, IV, V and i chords, it is interesting to note that the next prevalent chords in the corpus are bVII, vi, bVI, ii and bIII. The prevalence of bVII, bVI and vi also confirms De Clercq and Temperley's previous observations and illustrate how the chords that make up rock harmony are clearly different from those of common-practice harmony.

After examining the most frequently occurring chords in the corpus, I decided to follow De Clercq and Temperley's investigation of the frequency of the chromatic roots in the corpus. Like De Clercq and Temperley, I defined the chromatic relative root of chords in the corpus as any chord built on that root. The frequency of the relative chromatic roots in the RS 5x20 corpus and the Expanded Rock Corpus is shown in the histograms below.

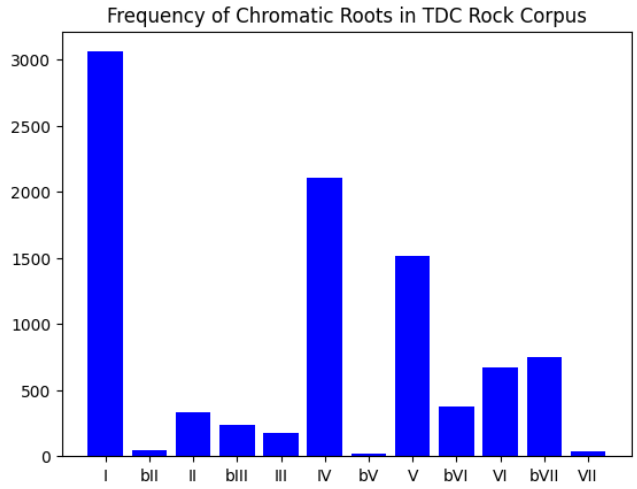

Figure 17: Frequency of Chords on the Chromatic Roots in De Clercq and Temperley's Rock Corpus

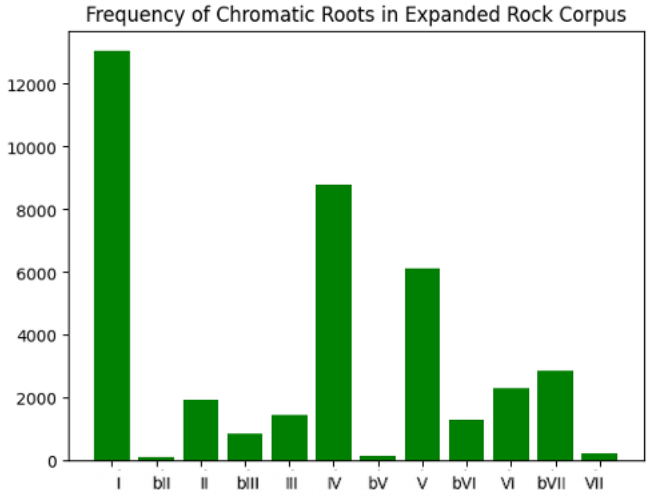

Figure 18: Frequency of Chords on the Chromatic Roots in the Expanded Rock Corpus

Here, the values on the y-axis represent the frequency of the chord in the corpus and the labels on the x-axis in both figures represent the set of major and minor chords built on each of the twelve chromatic scale degrees. Temperley and De Clercq also observed that the most frequently occurring chromatic roots are I, IV, V, bVII and VI and the analysis of the distribution of the twelve chromatic roots in the larger corpus above closely resembles their observations.

Next, I found the top ten most frequency occurring chords in the corpus and tracked their frequency across the decades. These frequencies are summarized in the bar plots below:

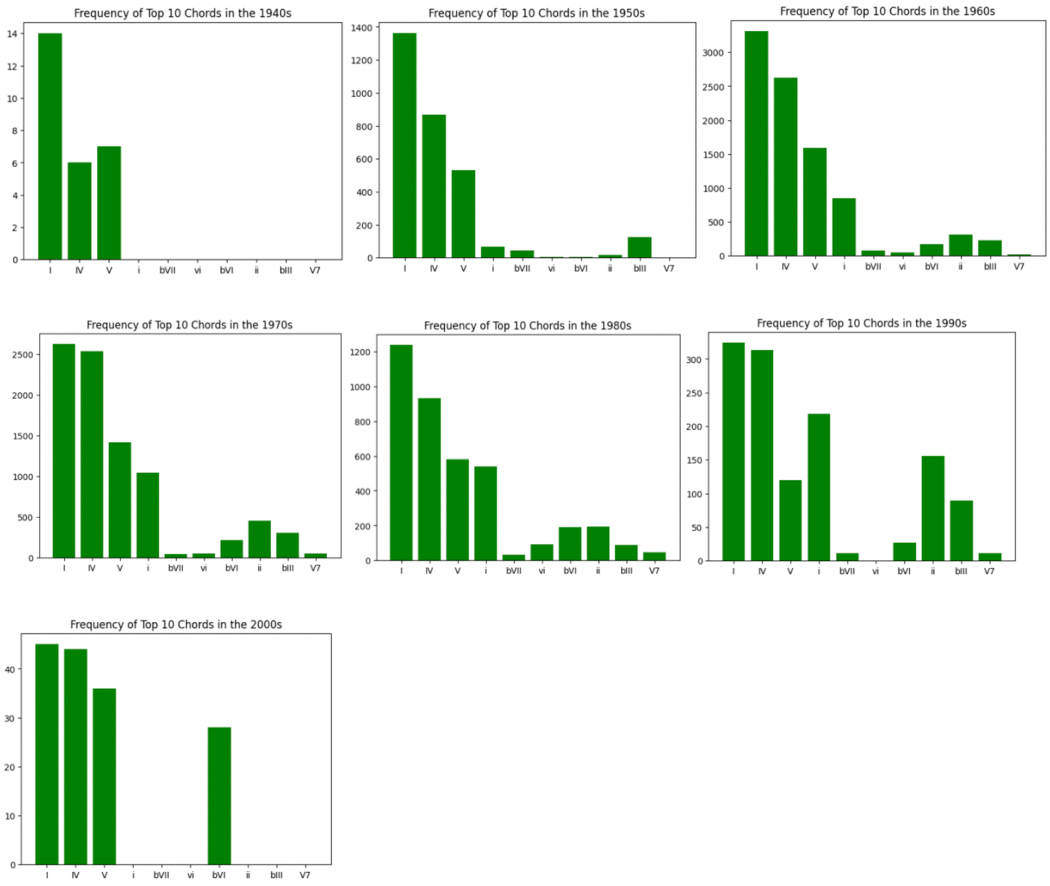

Figure 19: Frequency of the Top Ten Chords in each Decade

Here, the order of chords on the x-axis begins with the most frequently occurring chord in the corpus and ends with the 10th most frequently occurring chord in the corpus. Since there is not an equal number of songs in each decade (and only a few songs from the 1940s and 2000s) the frequency histograms are not exactly representative of the frequency of the top ten chords in the corpus in songs from each decade. In an effort to mitigate the fact that there are a different number of songs in each decade, I also calculated the percentage that each of these top ten chords made up of all of the chords in a particular decade. These histograms are shown below:

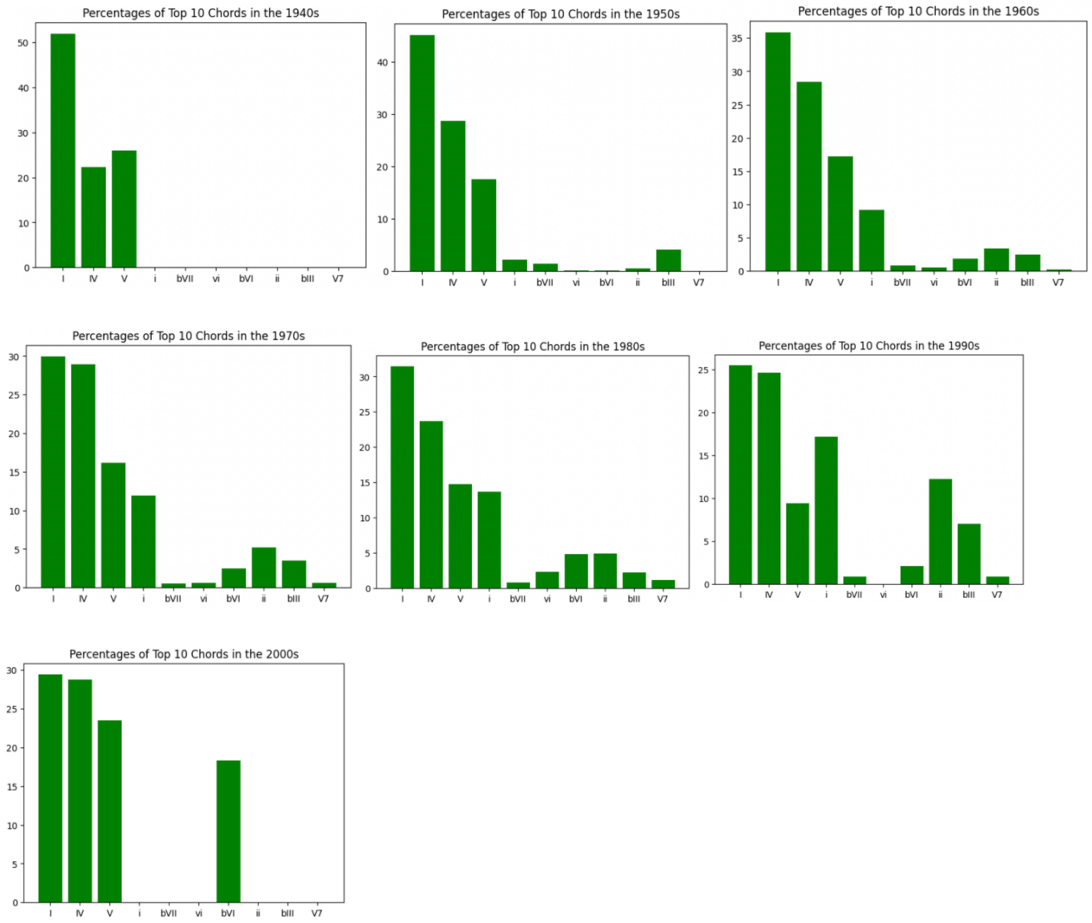

Figure 20: Percentages of the Top Ten Chords in each Decade

Comparing the two sets of histograms, it is clear that the I and IV chords make up the majority of the chords in all of the decades. This might be due to the fact that a large number of works in the corpus utilized guitars which are tuned in fourths or because of the strong influence of blues music on rock music. One trend that these histograms illustrate is that between the 1950s and 1990s, more of the top ten chords outside of I, IV, V and i are used suggesting that there is more harmonic diversity in the songs from these decades. While there are less songs in the corpus from the 1940s and 1950s compared with the 1980s and 1990s, the increase in harmonic diversity could also be attributed to the genre's predilection for modal mixture and subversion.[35]

[35] Chris McDonald, "Exploring Modal Subversions in Alternative Music." Popular Music 19, no. 3 (2000): 355.

Following this, I investigated chord transitions between major triads based on each chromatic root and between classes of chords built on each of the twelve chromatic roots. The results for the Expanded Rock Corpus are summarized in the heat map below.

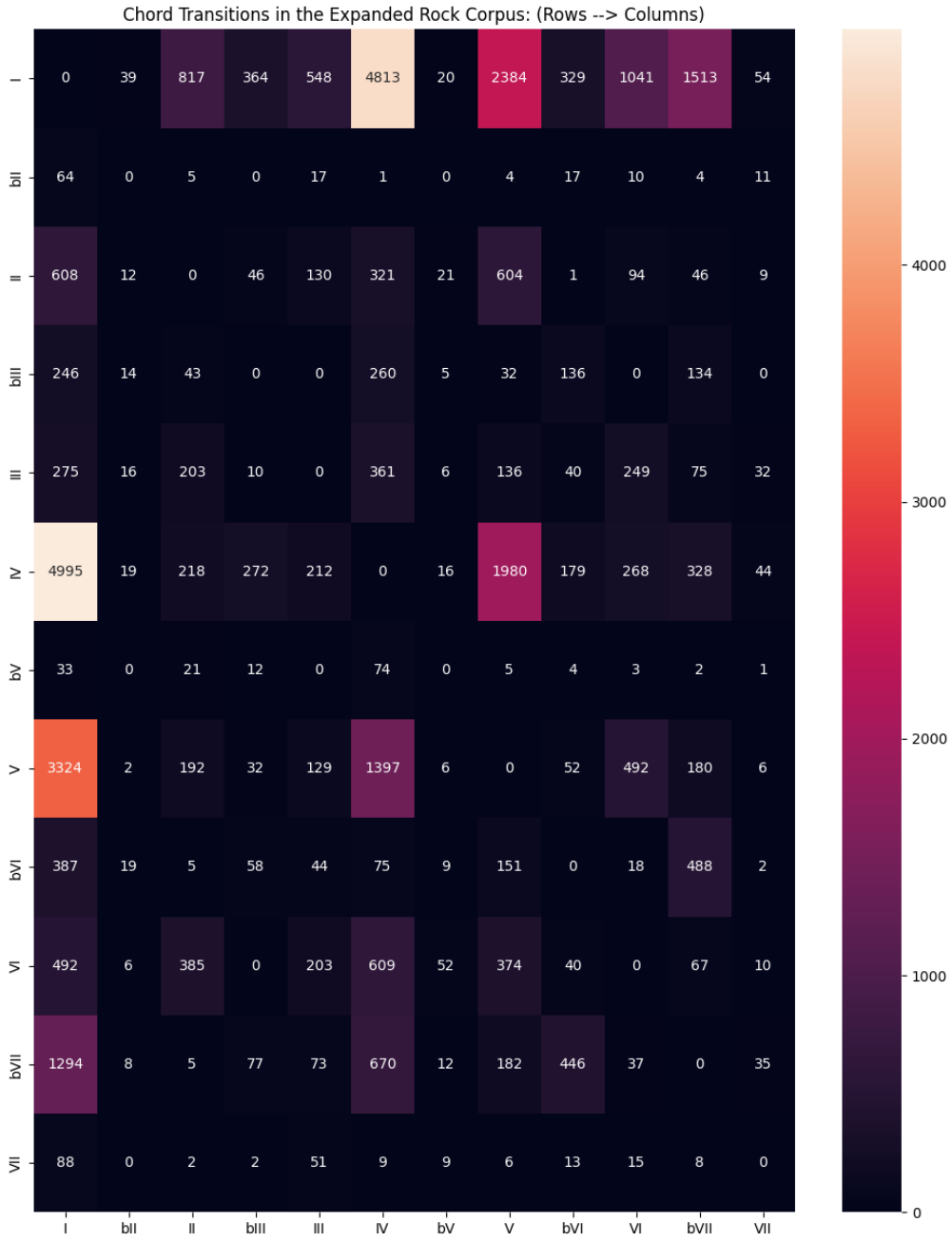

Figure 21: A Heatmap for the Frequency of Bigrams of Chromatic Roots in the Expanded Rock Corpus

This heat map of bigrams of chromatic roots illustrates how the bigrams I - IV and IV - I are the most common and occur 4813 and 4995 times respectively in the Expanded Rock Corpus. The next most frequently occurring bigrams are V - I , I - V, IV - V and V - IV which occur 3324, 2384, 1980 and 1397 times respectively. These observations suggest that the majority of bigrams between classes of chords on chromatic roots follow the harmonic syntax of common-practice tonal music. However, observing the set of the next most frequently occurring chromatic bigrams suggests that the syntax of rock music differs from the harmonic syntax of common-practice tonal music. For instance, the some of the next most frequently occurring chromatic bigrams include I - bVII, bVII - I, bVII - IV, which occur 1513, 1294 and 670 times respectively. The fact that these are some of the next most frequently occurring bigrams following suggests how bVII often serves as a modal substitute of V.

Similar to how I first calculated the empirical distribution of top ten chords in the corpus before converting it to percentages, I then found the top chromatic bigrams and calculated the percentage that they make up in the total number of bigrams in the corpus. To illustrate these findings, I will use a histogram with both positive and negative values. The positive values in the histogram correspond to the percentages that the chromatic bigram occur which listed on the X-axis in Figure 22 below. The negative values in the histogram, or the values that are shown by the red bars, represent the percentage of the total number of chromatic bigrams that the bigram on the x-axis occurs in the opposite direction of those on the x-axis. For instance, the chromatic bigrams I - IV and IV - I each make up roughly 15% of the total number of chromatic bigrams in the corpus. The histogram of the most frequent bigrams and the reverse bigrams are shown in the histogram below.

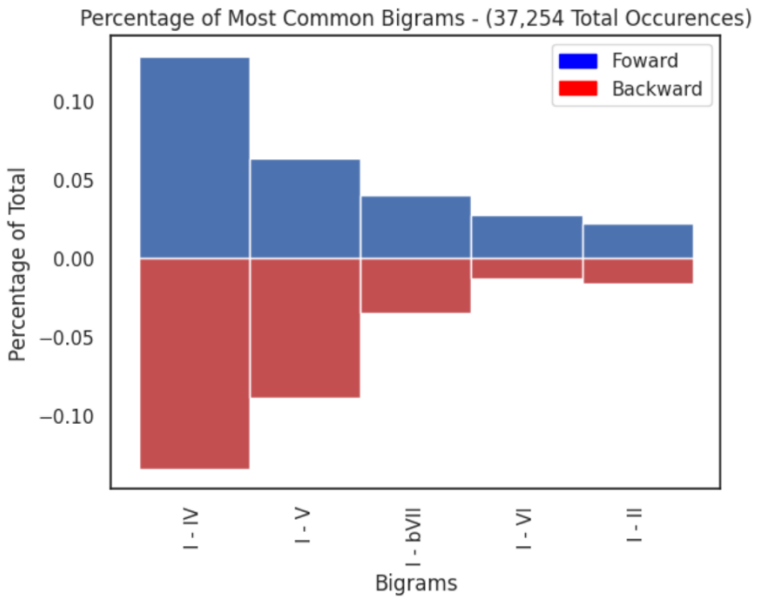

Figure 22: Tornado Diagram of the Top 5 Chromatic Chord Bigrams

Another observation that can be drawn from this histogram is that for each of these five most frequently occurring bigrams, the inverse bigram occurs roughly the same percentage of times which supports the conception that the harmonic syntax of rock music has strong bidirectional tendencies. Comparing my results for the chord transitions between classes of chords on the twelve chromatic roots with De Clercq and Temperley's on the RS 5x20 corpus, I found that the results qwew similar. I have again reproduced a copy of table 3 from De Clercq and Temperley's article below for comparison:

Ant \ Cons	I	bII	II	bIII	III	IV	#IV	V	bVI	VI	bVII	VII
I	0	25	132	94	44	1052	2	710	104	302	470	16
bII	31	0	0	0	2	0	0	0	0	0	0	12
II	120	1	0	2	20	58	0	97	0	24	10	0
bIII	50	6	6	0	0	64	2	2	67	0	41	0
III	16	0	39	0	0	46	0	6	0	60	3	4
IV	1,162	14	30	98	45	0	4	514	57	72	90	4
#IV	7	0	0	6	0	10	0	0	0	0	0	0
V	788	0	36	6	17	392	4	0	6	191	48	0
bVI	208	0	1	20	0	22	6	22	0	10	78	0
VI	144	0	87	0	32	260	0	124	21	0	3	0
bVII	386	0	0	11	2	188	2	26	114	6	0	0
VII	18	0	0	0	12	0	4	0	0	3	0	0

Figure 23: De Clercq and Temperley's table of chord transitions in the Rock corpus[36]

For instance, in De Clercq and Temperley's table, the values for the bigrams I-IV and IV-I are 1052 and 1162, I-V and V-I are 710 and 788, I-bVII and bVII-I are 470 and 386 and the values for the bigrams I-II and II-I are 132 and 120. Since the two values in each of the bigrams are roughly similar, my results parallel De Clercq and Temperley's. After investigating chromatic bigrams in the corpus, I began to investigate the larger harmonic patterns. The top twenty trigrams of between the chord classes on the chromatic roots are shown in Figure 24 below:

[36] David Temperley and Trevor De Clercq, "A Corpus Analysis of Rock Harmony," *Popular Music,* vol. 30, no. 1 (2011): 61.

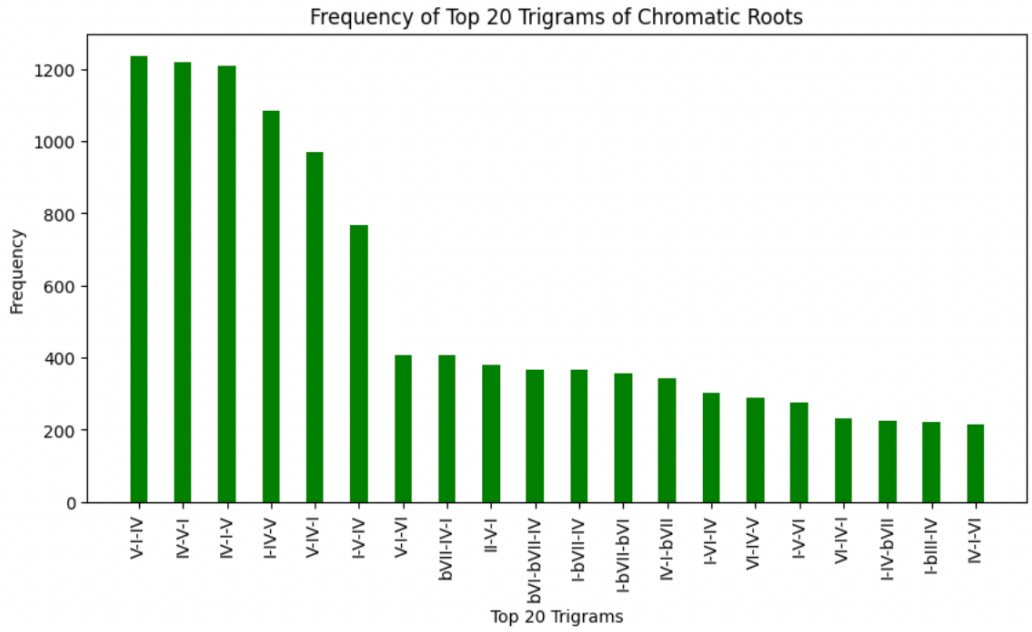

Figure 24: Frequency of the Top Twenty Trigrams in the Expanded Rock Corpus

This histogram of the top twenty trigrams in the corpus illustrates how the most frequently occurring trigrams are between the chromatic roots I, IV and V. The fact that the trigrams bVII - IV - I, II - IV - I and bVI - bVII - I all occur with similar frequencies around 400 and all end on the chromatic root suggest that these are the typical cadential formulations in the music in the Expanded Rock Corpus.

Finally, following De Clercq and Temperley's work I created and investigated chord vectors. Given the 439 songs in the corpus, I created a 439-dimensional vector for each major chord on the twelve chromatic roots and used roman numeral notation to denote each of these twelve vectors. I then put a '1' or '0' in each position of each vector if the chord appeared in the song (thus ignoring the number of times the chord occurs in the song). Following this, I calculated the correlation between each of the vectors. Since the correlation between two vectors indicates a degree of similarity between them, the correlation between two chords representing

two major triads on each relative root indicates the degree to which the chords are likely to occur in the same songs. Correlation values are between -1.0 and 1.0, where the correlation is positive if two chords co-occur and negative if they don't. Any correlation above .35 suggests that there is some correlation between the two chords. The correlations between the chord vectors are summarized in the heatmap below:

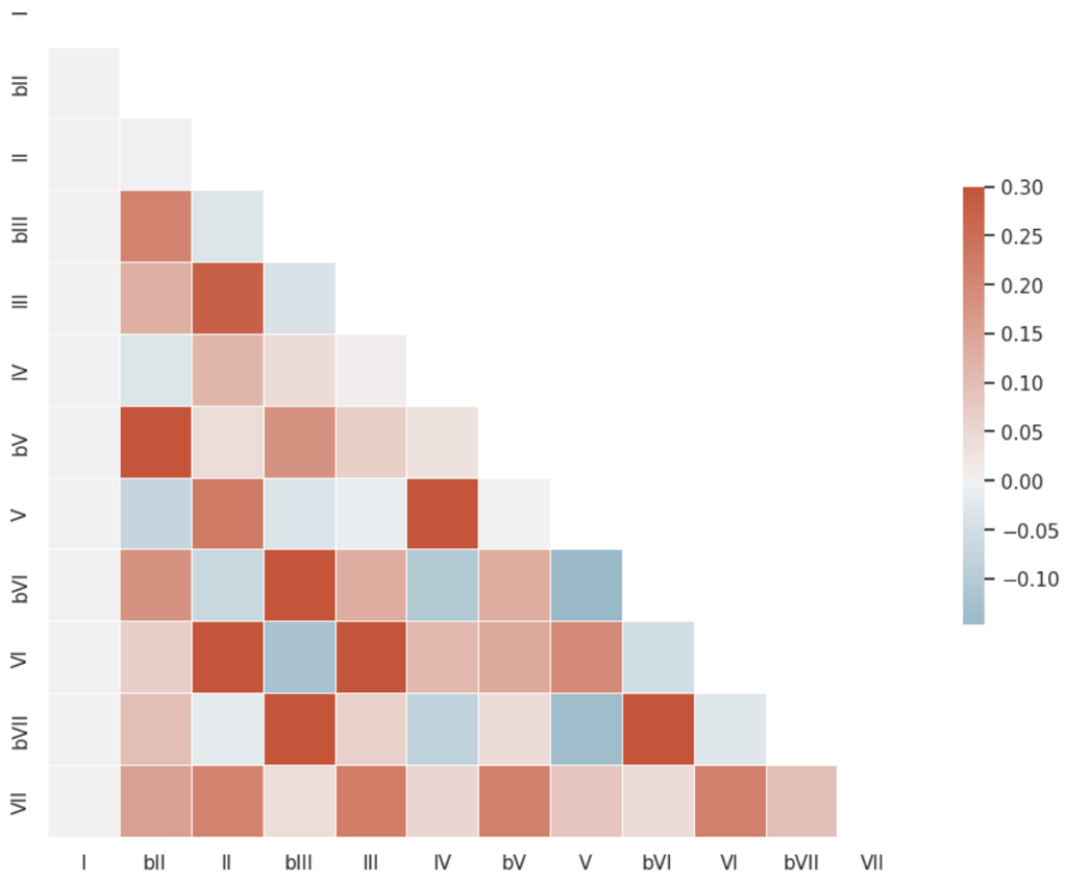

Figure 25: Heat Map of the Correlation Between the Chromatic Root Vectors

These results suggest that the following pairs of vectors tend to co-occur:

(bII, bV), (II, IV), (bIII, bVII), (bIII, bVI), (III, IV), (IV, V) and (bVI, bVII)

One observation about these pairs of vectors is that most of the intervals between their roots are fourths and seconds which might suggest that chords a fourth or second apart tend to occur

together. In order to better determine if certain vectors occurred together, I also produced a heat map with a dendrogram for the vectors, which is shown in Figure 26 below:

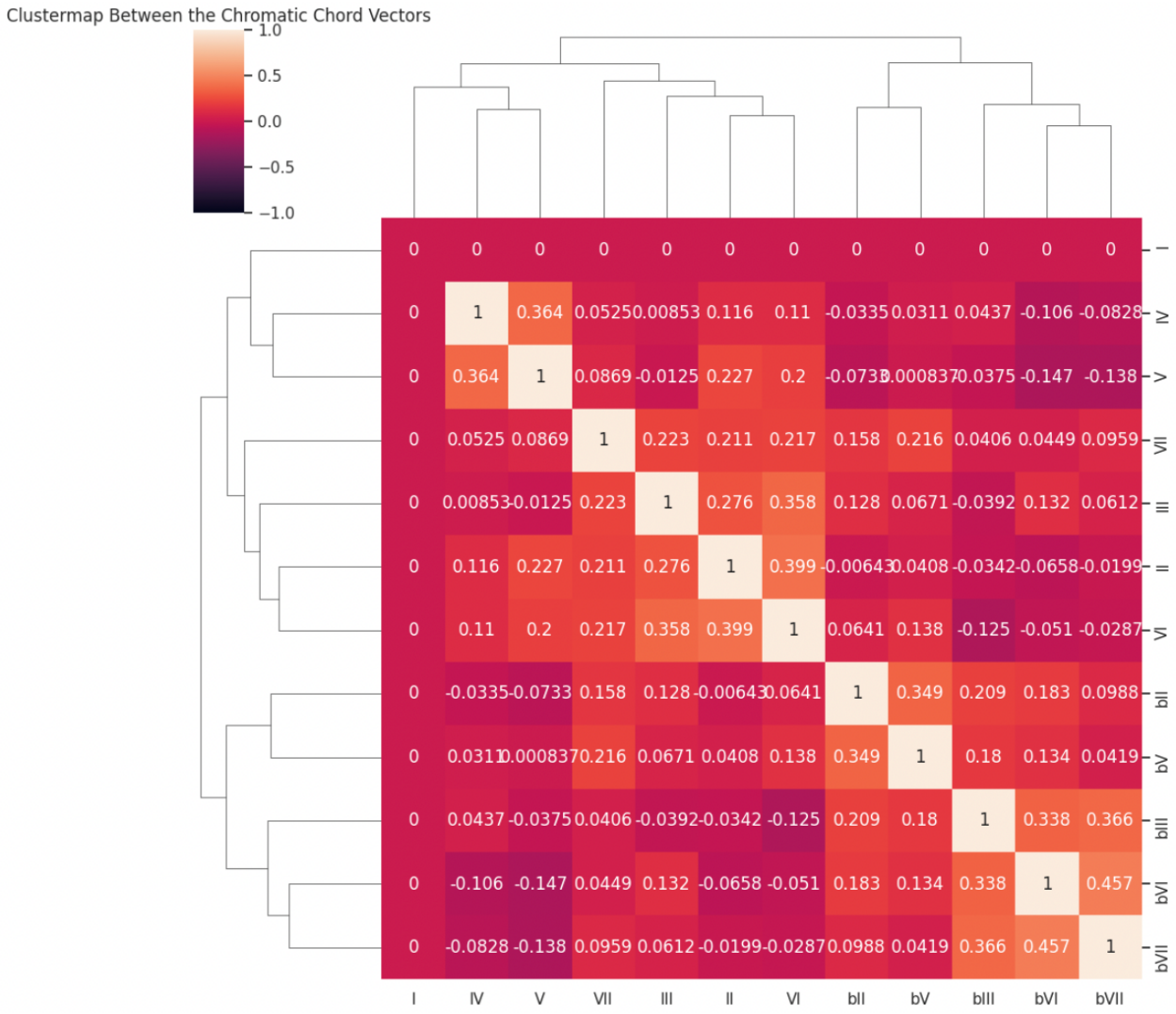

Figure 26: Heat Map with a Dendrogram of the Correlation Between the Chromatic Root Vectors

This heat map and dendrogram clearly suggests that the chromatic roots are roughly clustered into the following two groups: [I, IV, V, VII, III, II, VI] and [bII, bV, bIII, bVI, bVII]. This broad grouping confirms De Clercq and Temperley's observation that the chords based on the roots bVII, bIII and bVI tend to occur together and that the chords based on the roots II, VI and

III tend to occur together. The rough grouping from the heat map and dendrogram both confirms and expands upon De Clercq and Temperley's observation. The finer groupings of the dendrogram are as follows (where the brackets correspond to subsequent subgroupings):

[[I, [IV, V]], [VII, [III, [II, VI]]]]

[[bII, bV], [bIII, [bVI, bVII]]]

In addition to measuring the correlation between chord vectors and producing a dendrogram to capture the correlation between chord vectors, I also implemented K-means clustering on rows of the chord vector matrix; that is, on the set of 439 vectors for the songs which contain information about whether each of the twelve chromatic roots is present in the song in order to determine if these vectors for the songs could be clustered into groups. Using a "within-cluster sum of square distances" which measures the distances of each data point in all clusters to their respective centroids, I determined that the optimal number of clusters to use to try and cluster the song vectors was three. After running K-means clustering with K = 3, I then used principal component analysis (PCA) with two principal components in order to reduce the dimension of the data to try and visualize the results. The two axes of the plot represent the two principal components which are comprised of the different weighted combinations of the axes of the original data. The resulting scatter plot with K clusters and the composition of the two principal components are shown in Figure 27 below:

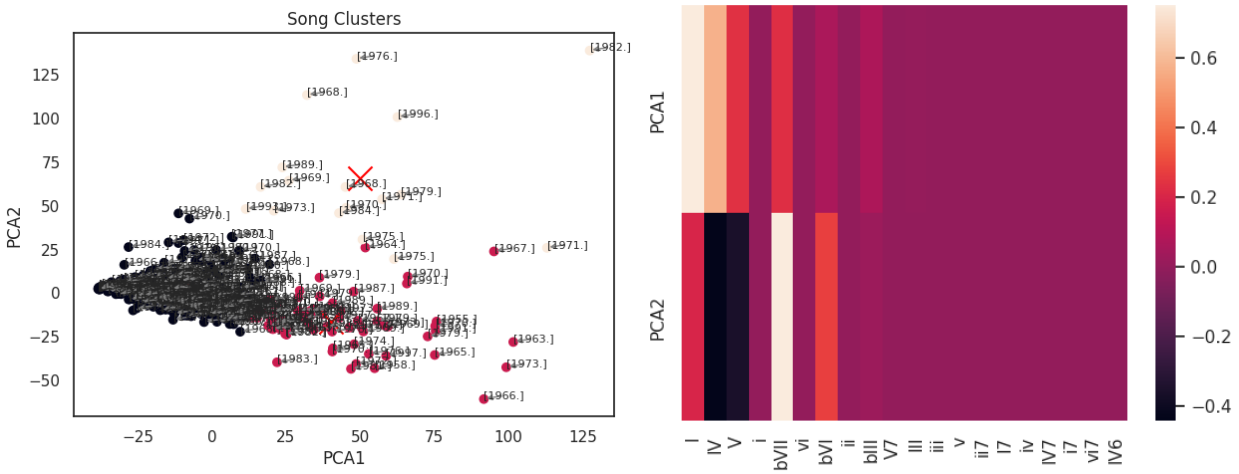

Figure 27: Principal Component Analysis and K-means Clustering on the Chord Vectors of the Songs in the Expanded Rock Corpus

Here, each dot in the left portion of Figure 27 represents a song which is labeled with its date and the X represents the center of each cluster. While the songs do not appear to group into clearly defined clusters, the composition of the two principal components (the high percentage of bVII and bVI in PCA2 and higher percentage of I, IV, and V in PCA1 suggest that these chords are important in determining the differences between groups of songs in the corpus which is reflective of how some progressions in the corpus are based on bVII and bVI while others are based on I, IV and V.

5. Hidden Markov Models and Probabilistic Context-Free Grammar Models on the Expanded Rock Corpus

Having introduced the Expanded Rock Corpus and discussed some of the patterns in the data, I will now discuss the implementation of Hidden Markov Models and Probabilistic Context-Free Grammar models.

5.1 Setting up the HMMs and PCFGs

Following the work of Hiroaki Tsushima et al. in "Generative Statistical Models with Self-Emergent Grammar of Chord Sequences", I implemented Hidden Markov Models and Probabilistic Context-Free Grammar models that were coded in C++. Since the models for the self-emergent HMM and PCFG from Tsushima et al.'s work were not compatible with the format of the data from the De Clercq and Temperley Rock corpus, I first implemented a find-and-replace algorithm in Python to convert the harmonic analyses in the rock corpus to pitch-class numbers that resulted after the chords in each of the analyses had been transposed to C major. An example of a harmonic analysis in this new format is shown below:

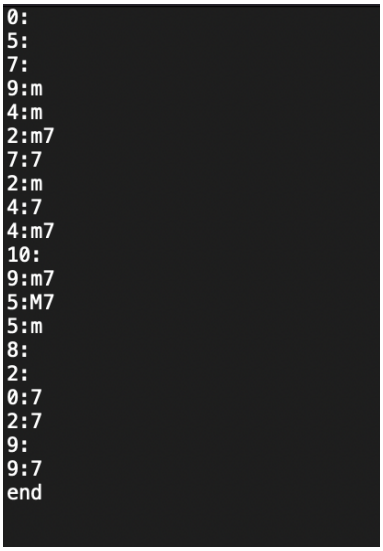

Figure 28: Example of the Data Format in Tsushima et al.

Here, each of the numbers are the pitch-class numbers which are followed by the chord quality. Following this, I created a file containing the top 20 most frequently used chords in the corpus which was used as an input for both models. This was a necessary step as ultimately it would be too computationally expensive and unwieldy to model the harmonic syntax of the corpus on every chord that occurred in the corpus. As a result, the HMM and PCFG models that I implemented were based only on top 20 chords in the corpus. In addition to specifying the number of symbols or top chords, I also specified the number of hidden states or syntactic categories that the models would have. Finally, I also specified the number of iterations on which the expectation-maximalization algorithm would run, and usually specified a number greater than or equal to 50 to ensure the that the models would converge or produce the optimal set of parameters. Using these most frequently occurring chords, both models were trained using the expectation-maximization algorithm. The resulting parameters were output in a file which contained the composition of each hidden state or syntactic category in terms of the top 20 symbols as well as the transition probabilities between all of these states. For the PCFG, I used the same files and specifications. The resulting parameters were output to a file which contained the composition of each of the syntactic categories and the list of production rules and their associated probabilities.

5.2 Results of the Hidden Markov Model on the Expanded Corpus

Running the HMM using the top 20 chords in the corpus, 5 syntactic categories and 50 iterations of the EM algorithm produced the following results:

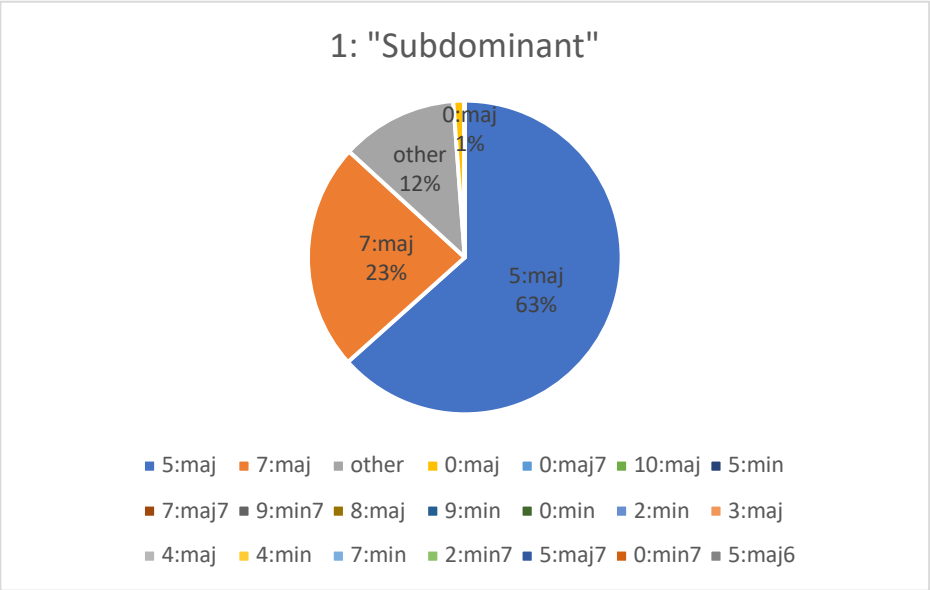

Figure 29a: Composition of the "Subdominant" Hidden State

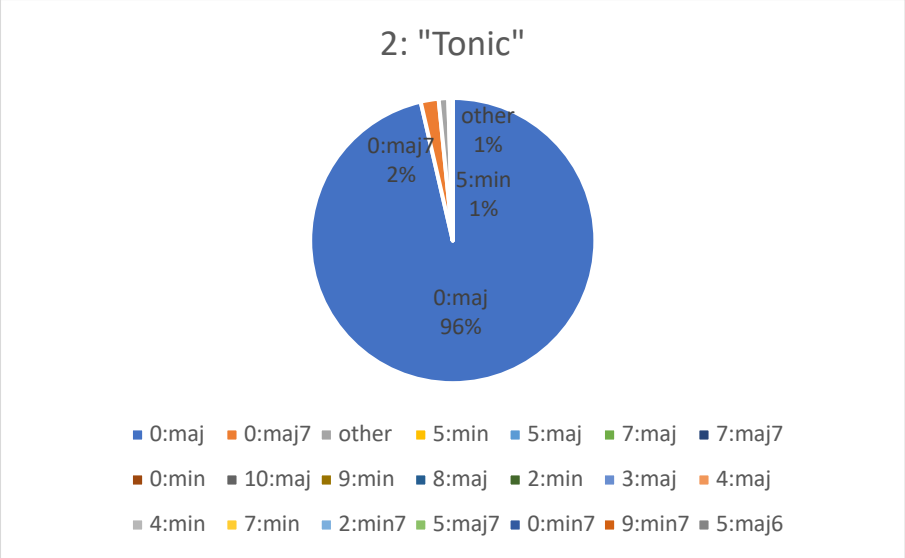

Figure 29b: Composition of the "Tonic" Hidden State

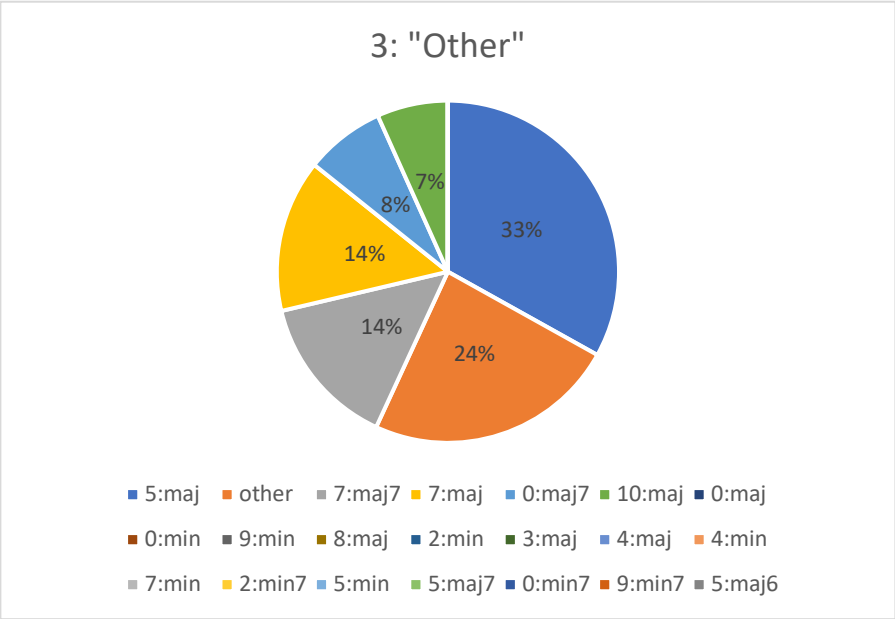

Figure 29c: Composition of the "Other" Hidden State

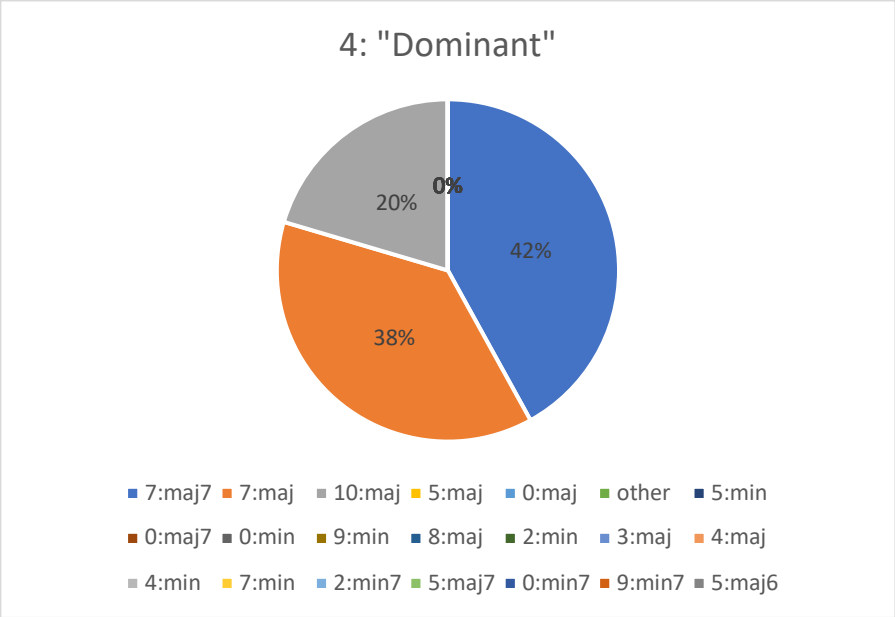

Figure 29d: Composition of the "Dominant" Hidden State

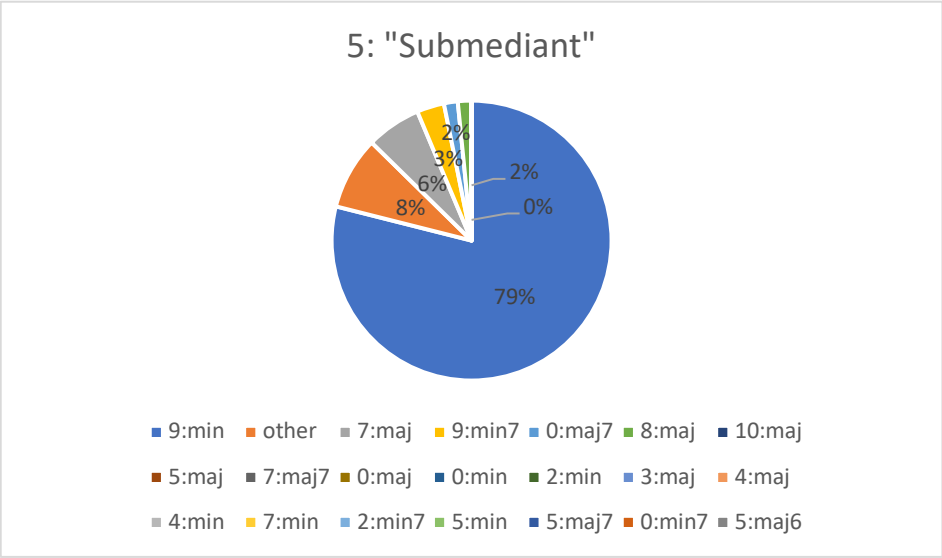

Figure 29e: Composition of the "Submediant" Hidden State

Transition Probabilities:					
	Subdominant	Tonic	Other	Dominant	Submediant
Subdominant	0.00	0.07	0.00	0.93	0.00
Tonic	0.22	0.00	0.42	0.02	0.34
Other	0.00	0.99	0.00	0.01	0.00
Dominant	0.00	0.97	0.00	0.03	0.00
Submediant	1.00	0.00	0.00	0.00	0.00

Figure 30: Table of Transition Probabilities Between Hidden States

In Figure 29, each pie chart represents the composition of one of the five hidden states of the HMM and the labels are the pitch class and followed by the chord quality. The labels in the pie charts correspond to the major, minor or seventh chords built on the chromatic pitch class associated with each number. The transition probabilities between each of the syntactic categories is given in Figure 30. Each of the main syntactic categories that arise could be labeled as "Tonic", "Subdominant", "Dominant", "Submediant" and "Other" in the theory of common practice harmonic syntax. While the composition of the "Tonic" and "Dominant" categories is similar to the composition of "Tonic" and "Dominant" syntactic categories in common practice harmony, note how 20% of the "Dominant" category consists of "10:maj" or "bVII" chords. Furthermore, note how there is a "submediant" category which primarily consists of "vi" chords.

The probabilities in the transition matrix also seem to suggest that the harmonic syntax is similar to that of music from the classical era as chords in the "Dominant" category always move to chords in the tonic category, chords in the "Subdominant" category always move to chords in the "Dominant" category and chords in the "Submediant" category always move to chords in the "Subdominant" category.

These results can be viewed in light of Walter Everett's "Classification of Rock's Preeminent Tonal Systems" which uses various characteristics to classify the prominent tonal systems in rock music. While introducing these systems, Everett notes that while the underlying principles of tonality often apply to rock music, the genre has evolved different ways of relating to that tonal background.[37] In particular, Everett develops the following classification of rock's preeminent tonal systems.

Table 1. Classifications of Rock's Preeminent Tonal Systems

1a	Major-mode systems with common-practice harmonic and voice-leading behaviors. May be inflected by minor-mode or chromatic mixture.
1b	Minor-mode systems with common-practice harmonic and voice-leading behaviors. May be inflected by major-mode or chromatic mixture.
2	Diatonic modal systems with common-practice voice-leading but sometimes not with common-practice harmonic behaviors.
3a	Major-mode systems, or modal systems, with mixture from modal scale degrees. Common-practice harmonic and voice-leading behaviors would be common but not necessary.
3b	Major-mode systems with progressive structures. Common-practice harmonic and voice-leading behaviors would be typical at lower, but not higher, levels.
4	Blues-based rock: minor-pentatonic-inflected major-mode systems. Common-practice harmonic and voice-leading behaviors not always emphasized at the surface, but may be articulated at deeper levels and/or in accompaniment.
5	Triad-doubled or power-chord minor-pentatonic systems unique to rock styles: I - bIII - IV - V - bVII. Common-practice harmonic and even voice-leading behaviors often irrelevant on the surface.
6a	Chromatically inflected triad-doubled or power-chord doubled pentatonic systems of early metal. Common-practice harmonic and voice-leading behaviors often irrelevant on the surface.
6b	Chromatically related scale degrees with little dependence upon pentatonic basis. Common-practice harmonic and voice-leading behaviors often irrelevant at deeper levels as well as surface.

Figure 31: Everett's Classification of Rock's Preeminent Tonal Systems[38]

[37] Walter Everett, "Making Sense of Rock's Tonal Systems," *Music Theory Online*, Volume 10, no. 4 (December 2004)

[38] Ibid.

Everett notes how he classifies the tonal systems in rock music in terms of their dependencies upon harmonic and voice-leading functions and how each subsequent classification on the list is progressively removed from common practice tonal behaviors.[39] However, Biamonte in "Pop/Rock Tonalities" notes that the ordering of the categories primarily follow the tonal conventions of classical music rather than rock music and offers the revised and simplified set of categories shown below:

Tonal Systems in Pop/Rock
(scale degrees are shown in relation to major)

1: expanded major mode: Î 2̂ 3̂ 4̂ 5̂ 6̂ ♭7̂ 7̂
Harmony and melody are major, Mixolydian, or use flexible scale degree 7̂. Common subsets are the diatonic major hexachord (scale degrees Î to 6̂) and the major pentatonic scale (Î 2̂ 3̂ 5̂ 6̂).

2: minor mode: Î 2̂ ♭3̂ 4̂ 5̂ ♭6̂ 6̂ ♭7̂
Harmony and melody are natural minor/Aeolian or Dorian. A common subset is the minor pentatonic scale (Î ♭3̂ 4̂ 5̂ ♭7̂).

3: blues/"pentatonic union": Î 2̂ ♭3̂ 3̂ 4̂ 5̂ 6̂ ♭7̂ 7̂
Harmony is basically major, although chord extensions and embellishments may be minor; melody has flexible third and seventh degrees with microtonal embellishments. A common subset is the minor pentatonic scale.

4: other forms of minor
melodic minor: Î 2̂ ♭3̂ 4̂ 5̂ 6̂ 7̂
harmonic minor: Î 2̂ ♭3̂ 4̂ 5̂ ♭6̂ 7̂
Phrygian: Î ♭2̂ ♭3̂ 4̂ 5̂ ♭6̂ ♭7̂
Locrian: Î ♭2̂ ♭3̂ 4̂ ♭5̂ ♭6̂ ♭7̂

5: chromatic: Î ♯Î/♭2̂ 2̂ ♭3̂ 3̂ 4̂ ♯4̂/♭5̂ 5̂ ♭6̂ 6̂ ♭7̂ 7̂

Basic Harmonic Units
1: functional harmony: different chord qualities on different scale degrees
2: major-triad doublings
3: power chords (parallel fifths and fourths; may be doubled at the octave)
4: monophonic: single line; may be doubled at the octave

Figure 32: Biamonte's Classification of Tonal and Harmonic Structures in Pop/Rock (after Everett 2004)[40]

[39] Ibid, 2.

[40] Nicole Biamonte, "Pop/Rock Tonalities" in Tonality Since 1950, ed. Wörner, Scheideler, and Rupprecht, Franz Steiner Verlag, 95.

The results of the composition of hidden states and transition matrix from the HMM run on the Expanded Rock Corpus suggest how harmonic syntax of the songs in the corpus could predominantly fall into Everett's first category of tonal systems in rock as there are major and minor-mode systems with common-practice harmonic behavior which are at times inflected with minor-mode or chromatic mixture like bVII and bII. Furthermore, the results of the HMM suggests that the songs in the corpus fall into Biamonte's "expanded major mode" tonal system and that the basic harmonic units utilize functional harmony.

It is important to note that while the results of the HMM appear to capture some information about the harmonic syntax of the songs in the corpus, there is a fair amount of overlap between the composition of the syntactic categories. For instance, the "Subdominant" category consists of 23% "V" chords and the "Dominant" category is comprised of 38% "V" chords. Also, the "Other" category is comprised of 33% "IV" chords and the "Subdominant" category is comprised of 63% "IV" chords. The fact that several chords belong to multiple syntactic categories ultimately makes it difficult to understand how the chord is functioning in the harmonic syntax. Therefore, it appears that this HMM does not clearly and definitively capture the harmonic syntax of the songs in the corpus. It is interesting to note that White and Quinn also mentioned this issue of overlapping syntactic categories in their article and devised a more complicated composite HMM to work around this issue.[41]

While creating a composite HMM similar to White and Quinn's was not practical, I decided to implement a resampling technique to artificially augment the dataset, and further

[41] Christopher White and Ian Quinn, "Chord Context and Harmonic Function in Tonal Music" *Journal of New Music Research*, vol. 40, no. 2, (2018): 318.

decreased the number of syntactic categories to four in effort to obtain results with more clearly defined syntactic categories. This involved randomly selecting with replacement songs from De Clercq and Temperley's corpus 20,000 times and adding each selected song to a new larger dataset. The results from running a HMM with four syntactic categories on this larger dataset are shown below.

	IV	vi	V, V6, V7, other	Tonic
IV	0	0	0.43	0.66
vi	0.73	0	0	0.26
V, V6, V7, other	0	0.48	0	0.51
Tonic	0.32	0.29	0.38	0

Figure 32: Transition Matrix for HMM on the Resampled Data

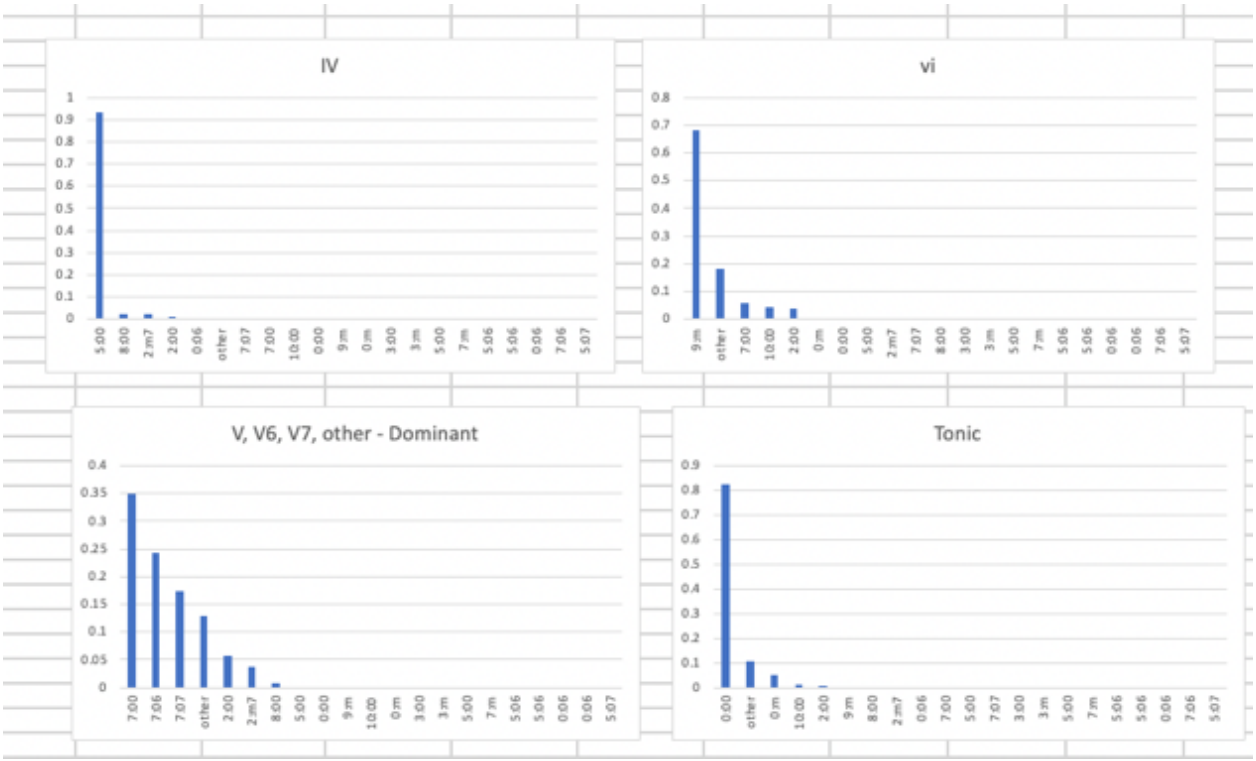

Figure 33: Composition of the Four Syntactic Categories

Here, the composition of the four syntactic categories in the figure above are represented in terms of bar plots instead of pie-charts and labels on the x-axis correspond to the different chords in pitch class notation and the y-axis corresponds to the percentage of which the chord makes up the syntactic category. Also, the labels for the syntactic categories in Figures 32 and 33 are the most common chords in the category and the labels "V, V6, V7, other – Dominant" refers to the fact that this syntactic category could be considered as "Dominant" and is primarily composed of V, V6, V7 and other chords. These results indicate a clearer picture of the harmonic syntax of the songs in the corpus as there is far less overlap in the composition of the syntactic categories. They also indicate how the harmonic syntax of the songs in the corpus is similar to the harmonic syntax of classical music. However, it is important to note how in the 20,000 elements in the resampled dataset, there were only 100 unique songs. Given that the model is only training on these 100 songs, any harmonic features unique to these 100 songs would greatly influence the results of the model. Therefore, the results of the model are not representative of the harmonic syntax of rock music.

At this point, I decided that even resampling the Expanded Rock Corpus and running the model would still produce inaccurate or biased results that would not be able to capture the harmonic syntax of the songs in the corpus and decided to move on to the PCFG model.

5.3 Results of the Probabilistic Context-Free Grammar on the Expanded Corpus

After running the HMMs on the Expanded Rock Corpus, I implemented the PCFG model on the corpus. The PCFG model uses the syntactic categories that were generated by training the HMM and outputs the probabilities of the production rules. Using the HMM syntactic categories that were produced from running the model on the top 20 chords with 5 syntactic categories and 100 iterations of the EM algorithm, the PCFG produced the following results:

Subdominant:				Tonic				Other				Dominant				Submediant			
X -->	Y	Z	Prod Prob	X -->	Y	Z	Prod Prob	X -->	Y	Z	Prod Prob	X -->	Y	Z	Prod Prob	X -->	Y	Z	Prod Prob
0	0	0	0.03	1	0	0	0.00	2	0	0	0.00	3	0	0	0.00	4	0	0	0.00
0	0	1	0.07	1	0	1	0.00	2	0	1	0.00	3	0	1	0.01	4	0	1	0.00
0	0	2	0.08	1	0	2	0.00	2	0	2	0.00	3	0	2	0.01	4	0	2	0.00
0	0	3	0.12	1	0	3	0.00	2	0	3	0.00	3	0	3	0.00	4	0	3	0.00
0	0	4	0.14	1	0	4	0.00	2	0	4	0.00	3	0	4	0.00	4	0	4	0.00
0	1	0	0.06	1	1	0	0.00	2	1	0	0.00	3	1	0	0.00	4	1	0	0.00
0	1	1	0.00	1	1	1	0.03	2	1	1	0.01	3	1	1	0.08	4	1	1	0.04
0	1	2	0.01	1	1	2	0.07	2	1	2	0.03	3	1	2	0.01	4	1	2	0.02
0	1	3	0.06	1	1	3	0.01	2	1	3	0.00	3	1	3	0.02	4	1	3	0.00
0	1	4	0.02	1	1	4	0.02	2	1	4	0.01	3	1	4	0.05	4	1	4	0.01
0	2	0	0.02	1	2	0	0.00	2	2	0	0.00	3	2	0	0.01	4	2	0	0.00
0	2	1	0.01	1	2	1	0.06	2	2	1	0.00	3	2	1	0.09	4	2	1	0.08
0	2	2	0.00	1	2	2	0.02	2	2	2	0.05	3	2	2	0.11	4	2	2	0.09
0	2	3	0.04	1	2	3	0.03	2	2	3	0.00	3	2	3	0.07	4	2	3	0.03
0	2	4	0.00	1	2	4	0.05	2	2	4	0.04	3	2	4	0.07	4	2	4	0.04
0	3	0	0.09	1	3	0	0.00	2	3	0	0.00	3	3	0	0.00	4	3	0	0.00
0	3	1	0.01	1	3	1	0.01	2	3	1	0.00	3	3	1	0.02	4	3	1	0.00
0	3	2	0.04	1	3	2	0.02	2	3	2	0.01	3	3	2	0.05	4	3	2	0.01
0	3	3	0.00	1	3	3	0.00	2	3	3	0.00	3	3	3	0.01	4	3	3	0.00
0	3	4	0.01	1	3	4	0.01	2	3	4	0.00	3	3	4	0.05	4	3	4	0.01
0	4	0	0.13	1	4	0	0.00	2	4	0	0.00	3	4	0	0.00	4	4	0	0.00
0	4	1	0.01	1	4	1	0.03	2	4	1	0.02	3	4	1	0.05	4	4	1	0.03
0	4	2	0.00	1	4	2	0.02	2	4	2	0.02	3	4	2	0.06	4	4	2	0.04
0	4	3	0.02	1	4	3	0.01	2	4	3	0.00	3	4	3	0.01	4	4	3	0.00
0	4	4	0.02	1	4	4	0.03	2	4	4	0.00	3	4	4	0.06	4	4	4	0.04
0	I		0.00	1	I		0.00	2	I		0.00	3	I		0.00	4	I		0.00
0	IV		0.00	1	IV		0.00	2	IV		0.00	3	IV		0.00	4	IV		0.00
0	V		0.00	1	V		0.00	2	V		0.00	3	V		0.00	4	V		0.00
0	i		0.00	1	i		0.00	2	i		0.00	3	i		0.00	4	i		0.00
0	bVII		0.00	1	bVII		0.00	2	bVII		0.00	3	bVII		0.00	4	bVII		0.00
0	vi		0.00	1	vi		0.00	2	vi		0.00	3	vi		0.00	4	vi		0.00
0	bVI		0.00	1	bVI		0.00	2	bVI		0.00	3	bVI		0.00	4	bVI		0.00
0	ii		0.00	1	ii		0.00	2	ii		0.00	3	ii		0.00	4	ii		0.00
0	bIII		0.00	1	bIII		0.00	2	bIII		0.00	3	bIII		0.00	4	bIII		0.00
0	V7		0.00	1	V7		0.00	2	V7		0.00	3	V7		0.00	4	V7		0.00
0	III		0.00	1	III		0.00	2	III		0.00	3	III		0.00	4	III		0.00
0	iv		0.00	1	iv		0.00	2	iv		0.00	3	iv		0.00	4	iv		0.00
0	v		0.00	1	v		0.00	2	v		0.00	3	v		0.00	4	v		0.00
0	ii7		0.00	1	ii7		0.00	2	ii7		0.00	3	ii7		0.00	4	ii7		0.00
0	I7		0.00	1	I7		0.00	2	I7		0.00	3	I7		0.00	4	I7		0.00
0	iv		0.00	1	iv		0.00	2	iv		0.00	3	iv		0.00	4	iv		0.00
0	V7		0.00	1	V7		0.00	2	V7		0.00	3	V7		0.00	4	V7		0.00
0	i7		0.00	1	i7		0.00	2	i7		0.00	3	i7		0.00	4	i7		0.00
0	vi7		0.00	1	vi7		0.00	2	vi7		0.00	3	vi7		0.00	4	vi7		0.00
0	V6		0.00	1	V6		0.00	2	V6		0.00	3	V6		0.00	4	V6		0.00
0	other		0.00	1	other		0.57	2	other		0.80	3	other		0.12	4	other		0.55

Figure 34: Table of Production Rules with their Associated Probabilities for the PCFG

In Figure 34, the 0s refer to the "Subdominant" category, the 1s to the "Tonic" category, the 2s to the "Other" category, the 3s to the "Dominant" category and the 4s to the "Submediant" category. In each of the five groups of columns, the first column represents the symbol on the left-hand side of the production rules, the following two columns represent the two symbols that this symbol is replaced by, and the last column contains the probability of this production rule. For instance, in this PCFG, the production rule which replaces a chord in the "Subdominant" category with a chord in the "Tonic" category and a chord in the "Dominant" category occurs 6% of the time because the 9th row in the "Subdominant" section of the table contains a "0" followed by a "1" and "3" which is followed by ".06". It is important to note that in each of the categories, the probabilities of the production rules that would produce the chord symbols (that is the bottom rows containing the roman numerals) are zero except for the "other" symbol. This

means that the grammar does not produce any of the top 20 chords in the corpus which is clearly incorrect. In the following section, I will further discuss some of the issues with the results from these models, problems that were encountered while implementing the models that may have affected the result, and further steps that could be taken to achieve better results using these models.

5.5. Problems Encountered while Implementing the HMM and PCFG on the Expanded Corpus

One issue that I encountered that could have affected the performance of both models was finding the correct format of the data to run the model on. Given that both the HMM and PCFG grammars dealt with input data that was in multiple formats, (for instance a "I" chord could be formatted as both "0:" and "0:maj") and that the Readme.txt file that I was provided with by Eita Nakamura through email correspondence did not indicate when to use each type of format, it was very difficult to determine if the data that the model was training on was in the right format and involved a lot of guessing and checking. In addition to the ambiguous formatting of the data that HMM and PCFG models accepted, there was also the issue of converting the data from the format that De Clercq and Temperley used.

Aside from the format of the data, another reason the that the models did not perform well could have been due to the small size of the corpus. For comparison, the datasets that were used by in Tsushima et al. in their "Generative Statistical Models with Self-Emergent Grammar of Chord Sequences" consist of 3000 chord progressions for the J-pop data and 1531 chord progressions for the Billboard data whereas the Expanded Rock Corpus only consists of 439 songs.

Another reason why the models that I implemented did not perform well could have been the result of the initial randomness that is inherent in training HMM models. Before the parameters of the HMM and PCFG are adjusted to fit the training data, they are initialized to some initial values according to a prespecified distribution. As a result, training the model with the EM algorithm multiple times on the same data will produce similar but not identical results. One important difference between the HMM I implemented and the one that White and Quinn implemented in their article is that the HMM they implemented was a composite HMM and was trained with a different optimization algorithm. Their composite HMM consisted of 300 HMMs, each of which had different randomized parameters and was trained on a training subset and used to classify each chord in the testing subset into one of k different syntactic categories. They then looked at how each of the 300 HMMs classified each chord and derived new syntactic categories for the composite HMM using a k-medoids algorithm. As a result of using a k-medoids algorithm, they were also able to quantify the extent to which each of the syntactic categories overlapped. For a more detailed discussion of their approach, I refer the interested reader to pg. 318 and pg. 335 of their article.[42] The fact that White and Quinn calculated silhouette widths in order to measure the extent to which the syntactic categories of their HMMs were overlapping suggests that they most likely also encountered the issue of overlapping syntactic categories that I encountered when running a single HMM on the Expanded Rock Corpus data. While implementing a composite HMM similar to White and Quinn's would have most likely produced better results for the Expanded Rock Corpus, I decided that it would ultimately be difficult and would most likely require more computational power than I had available.

[42] Ibid, 318-335.

6. Conclusion and Summary of Findings

In this thesis I first introduced the concept of harmonic syntax and discussed several models that music theorists have used to study it; I discussed n-gram and Markov models, hidden Markov models and probabilistic context-free grammars. Following this, I discussed White and Quinn's previous work using a composite HMM to study the harmonic syntax of the Kostka-Payne, McGill Billboard and Bach Chorale Corpuses. Following this, I assessed the work of De Clercq and Temperley and their corpus of rock music and subsequently expanded their corpus to create the Expanded Rock Corpus. I then replicated the basic statistical analysis that De Clercq and Temperley performed on their corpus on the Expanded Rock Corpus and found that the distribution of chord roots was similar to De Clercq and Temperley's analysis in that the I, IV and V followed by bVII, VI and II were the most frequently occurring chord roots. I also found that the most frequently occurring bigrams of chord roots were V - I, I - V, IV - V and V - IV followed by I - bVII, bVII - I and bVII - IV which were similar to the values that De Clercq and Temperley found. I also observed how the top five chord root bigrams in the corpus had strong bidirectional tendencies. Following this, I created chord vectors for the twelve chord roots, measured the correlation between chord vectors and found the vectors tended to group together as follows:

[[I, [IV, V]], [VII, [III, [II, VI]]]] [[bII, bV], [bIII, [bVI, bVII]]]

After conducting basic statistical analyses, I implemented HMMs and PCFGs on the Expanded Rock Corpus and found that while the results from the HMM suggested that the harmonic syntax of the works in the corpus were similar to the harmonic syntax of common practice music, the composition of the syntactic categories were different. In particular, I observed that the

"Dominant" category from one of the HMMs also contained "bVII" chords. I also found that after running the PCFG on the Expanded Rock Corpus, the results from the PCFG were inconclusive. Finally, I discussed several of the issues from these models and problems that were encountered while implementing them that may have affected the results.

In conclusion, while this thesis was unable to definitively provide a detailed description of the harmonic syntax of rock music using HMM and PCFGs, White and Quinn's work using composite HMMs and the work of Tsushima, Hiroaki, et al. using larger datasets illustrate that it is possible to use these models to model the harmonic syntax of rock music. One further direction of research could involve expanding the Expanded Rock Corpus and implementing composite HMMs in order to better understand the harmonic syntax of rock music. Another direction for future research might focus on conducting more basic statistical analysis on the Expanded Rock Corpus and could investigate if there are trends in the frequency of chromatic roots or specific chords over time. Finally, another direction of future research could use other statistical clustering methods such as fuzzy clustering to determine how chromatic roots, chords or individual songs group together.

7. Appendix

7.1 Link to Expanded Rock Corpus, Further Discussion of Algorithms and Code Base

https://github.com/giwdulttam/Expanded-Rock-Corpus

7.2 Discussion of the "Forward-Backward" Algorithm for Training HMMs:

In this algorithm X_t for t in {1,…,T} are random variables and correspond to the hidden state variable taking on some value at time t in the sequence. In our case X_t will take on a value in a hidden or syntactic category (tonic, dominant, etc…) and P(X_t) is the probability of it taking on some value. The distribution of the random variable X_t is the collection of probabilities of X_t taking on values (which in this thesis are tonic, dominant, etc..); since it takes on a discrete set of values one could imagine this as a vertical bar graph.

This algorithm computes the marginal posterior probabilities of all hidden state variables given a sequence of observations. For a given hidden variable X_t of a HMM, it computes P(X_t | o_{1:T}) – that is, the probability of a certain hidden variable taking on some value given that I have a series of observations o_{1:T}. In order to compute this posterior marginal, the algorithm will compute the probability of the hidden variable X_t having a certain value given the observations o_{1:t} where t < T. Then it will compute a set of backward probabilities; that is, the probability P(o_{t+1 : T} | X_t) of observing the remaining observations o_{t+1:T} given that that the hidden variable X_t has some value or is at some state or chord category. Combining these two probabilities through Bayes' rule, I can obtain the probability: P(X_t | o_{1:T}), which is the distribution of X_t given the sequence of observations:

$$P(X_t \mid o_{1:T}) = P(X_t \mid o_{1:t}, o_{t+1:T}) \propto P(o_{t+1:T} \mid X_t) P(X_t | o_{1:t})$$

In this case, the posterior marginal P(X_t | o_{1:T}) is the distribution over the possible syntactic chord categories (tonic, dominant, etc…) that the process will take on at some point in the sequence t given that I have the sequence of chords o_{1:T}. These marginal posterior

probabilities are shown by the probabilities on the blue arrows in the diagram of the HMM in Figure 2.

In the algorithm outlined above, the Bayes' Rule, also known as Bayes' Theorem or Bayes' Law, is used to provide a way to updating the probability of an event occurring given that some other event has occurred. In particular, it provides a convenient formula to express this conditional probability in terms of probabilities are more likely to be known or easier to compute. In particular, it allows us to express the posterior probability of an event as an expanded version of the join density of the events over the marginal probability of the event that is being conditioned on. This can be represented as follows:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

7.5 Python Code to Find Patterns in the Expanded Corpus

```
# -*- coding: utf-8 -*-
"""Python Code for Rock Corpus Analysis

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1AXDoMCDpEMQdKL2MUmvPYYQEOiH9-dLO

#Import the Corpus
"""

from google.colab import files
uploaded = files.upload()

import pandas as pd
big_frame = pd.read_excel('Expanded Rock Corpus.xlsx', index_col=0)

big_frame

analysis_dates = big_frame[['Date','Harmonic Analysis']].to_numpy()

analysis = big_frame['Harmonic Analysis'].to_numpy()

date = big_frame[['Date']].to_numpy()

analyses_Copy = analysis

for i in range(len(analyses_Copy)):
  analyses_Copy[i] = analyses_Copy[i].replace("[C]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[C#]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[Db]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[D]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[Eb]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[E]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[F]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[F#]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[G]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[G#]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[Ab]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[A]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[A#]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[Bb]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[B]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("[Gb]", "")
  analyses_Copy[i] = analyses_Copy[i].replace("modulation", "")

```

```
47
48
49    analyses_Copy[i] = analyses_Copy[i].replace("|", "")
50    analyses_Copy[i] = analyses_Copy[i].replace(".", "")
51    analyses_Copy[i] = analyses_Copy[i].replace("[4/4]", "")
52    analyses_Copy[i] = analyses_Copy[i].replace("[2/4]", "")
53    analyses_Copy[i] = analyses_Copy[i].replace("[0]", "")
54    analyses_Copy[i] = analyses_Copy[i].replace("[R]", "")
55    analyses_Copy[i] = analyses_Copy[i].replace("[12/8]", "")
56    analyses_Copy[i] = analyses_Copy[i].replace("[7/8]", "")
57    analyses_Copy[i] = analyses_Copy[i].replace("R", "")
58    analyses_Copy[i] = analyses_Copy[i].replace("[3/4]", "")
59    analyses_Copy[i] = analyses_Copy[i].replace("[6/8]", "")
60    analyses_Copy[i] = analyses_Copy[i].replace("[5/4]", "")
61    analyses_Copy[i] = analyses_Copy[i].replace("Warning: 'Vr5' is defined but never
62 used", "")
63    analyses_Copy[i] = analyses_Copy[i].replace("Warning: 'Ch5' is defined but never
64 used", "")
65
66    analyses_Copy[i] = analyses_Copy[i].split()
67
68 analyses_Copy
69
70 #make numpy arrays for each decade
71
72 df_1940s = []
73 df_1950s = []
74 df_1960s = []
75 df_1960s = []
76 df_1970s = []
77 df_1980s = []
78 df_1990s = []
79 df_2000s = []
80
81 for i in range(analysis_dates.shape[0]):
82    if(1940 <= analysis_dates[i,0] < 1950):
83      df_1940s.append(analyses_Copy[i])
84
85    if(1950 <= analysis_dates[i,0] < 1960):
86      df_1950s.append(analyses_Copy[i])
87
88    if(1960 <= analysis_dates[i,0] < 1970):
89      df_1960s.append(analyses_Copy[i])
90
91    if(1970 <= analysis_dates[i,0] < 1980):
92      df_1970s.append(analyses_Copy[i])
93
94    if(1980 <= analysis_dates[i,0] < 1990):
95      df_1980s.append(analyses_Copy[i])
96
97    if(1990 <= analysis_dates[i,0] < 2000):
98      df_1990s.append(analyses_Copy[i])
```

```
 99
100   if(2000 <= analysis_dates[i,0]):
101     df_2000s.append(analyses_Copy[i])
102
103 """#Determine frequency counts of symbols and build dictionary to replace on top 50
104 or so...."""
105
106 from collections import Counter
107
108 c = Counter()
109
110 for d in analyses_Copy: #***
111   c.update(d)
112
113 # for k, v in c.items():
114 #   print(f'{k} = {v}')
115
116 #create lists of chords that are on each chromatic root
117
118 zero = ['I', 'I7', 'I6', 'V7/IV', 'i', 'i6', 'I64', 'Va/IV', 'Isus4', 'Id7',
119 'V42/IV', 'i7', 'V11/IV', 'ii7/bVII', 'Id7#9', 'Isus2', 'I42', 'i42', 'Id42',
120 'V/IV', 'I#9', 'I9', 'i64']
121 one = ['bII', 'bIId7', 'viio/ii', 'bII7']
122 two = [ 'V7/V', 'II', 'V/V', 'ii', 'ii7', 'II7', 'IId7', 'ii65', 'iio6', 'ii11',
123 'V6/V', 'II65', 'ii64', 'iis4', 'ii42', 'II9', 'V/v', 'II11', 'iih43', 'iih42',
124 'ii9']
125 three = ['bIII', 'bVId7/V', 'biii7', 'biii', 'bIII7', 'bIII64',  'V/bVI', 'V42/bVI',
126 'bIII6']
127 four = ['V/VI', 'iii', 'III', 'V/vi', 'V7/vi', 'III7', 'iii64', 'iii7', 'bIV',
128 'iii6', 'V6/vi', 'biv7', 'III64', 'iii43',  'Va7/vi',  'Va65/vi']
129 five = ['IV', 'IVd7', 'iv', 'IV9', 'IV64', 'IV6', 'IV7', 'iv7', 'iv6', 'Iv7', 'Iv',
130 'iV', 'IV42', 'bVb5', '#IV', 'IVsus4', 'IVssu4', 'IVssus4', 'iv64']
131 six = ['bV', 'viix7/V', 'viio/V', 'viix43/V', 'bV7', 'bVd42', 'viix42/V', 'V/VII',
132 'V42/VII', 'bv', 'V+11', 'v64']
133 seven = ['V', 'V7', 'V7s4', 'V64', 'V7sus4', 'V43', 'v', 'V13', 'V6', 'v7', 'Vs4',
134 'V11', 'Vsus4', 'iv/ii', 'V42', 'V65', 'V9', 'v6', 'v9', 'v7s4',  'iv6/ii']
135 eight = ['bVI', 'bVI7', 'bVId7', 'bVi7', 'bvi', 'bVI6', 'bVIb5', 'bVi', 'bVIs4',
136 'V7/bII', 'viix7/vi']
137 nine = ['vi7','vi', 'VI', 'V/ii', 'VI7', 'V7/ii', 'V7/ii', 'V43/ii', 'vi6', 'VId7',
138 'vi64', 'V6/ii', 'vi42', 'vih7', 'VI6', 'VI9', 'ii/IV']
139 ten = ['bVII', 'bVII7', 'bVIId7', 'bVId7/ii', '#VI', 'bvii', 'ii7/bVII', 'bViI',
140 'bVI64/ii', 'V7/bIII', 'bVII64', 'bVII6', 'bVII9', 'bvii7',  'IV/IV']
141 eleven = ['V7/iii', 'VII', 'vii', 'viix43', 'VII7', 'iih7/vi', 'ii7/vi', 'viix42',
142 'V7/III', 'vii7', 'VII9', 'V/iii', 'viix7', 'viio6', 'vii64']
143
144 dictionary_of_chords_and_counts = dict(c)
145 dictionary_of_chords_and_counts
146
147 from itertools import chain
148 from collections import Counter
149 import operator
150
```

```
chord_counts_dict = dict(Counter(chain.from_iterable(analyses_Copy))) #***

print(chord_counts_dict)


most_frequent_chords = sorted(chord_counts_dict, key=chord_counts_dict.get,
reverse=True)

top_20_chords = most_frequent_chords[0:20]
top_10_chords = most_frequent_chords[0:10]


top_20_chord_dic = dict((k, chord_counts_dict[k]) for k in top_20_chords)
top_10_chord_dic = dict((k, chord_counts_dict[k]) for k in top_10_chords)

"""#Create Histogram of Top Chords"""

import matplotlib.pyplot as plt

plt.bar(list(top_20_chord_dic.keys()), top_20_chord_dic.values(), color='g')

plt.title("Frequency of Top 20 Chords")
plt.show()

from itertools import chain
from collections import Counter
import operator


#counts chords in analyses_Copy
chord_counts_dict = dict(Counter(chain.from_iterable(analyses_Copy)))

print(chord_counts_dict)


most_frequent_chords = sorted(chord_counts_dict, key=chord_counts_dict.get,
reverse=True)

print(len(most_frequent_chords))

top_50_chords = most_frequent_chords[0:50]
# top_50_chords

chromatic_roots_maj = ['I', 'bII', 'II', 'bIII', 'III', 'IV', 'bV', 'V', 'bVI',
'VI', 'bVII', 'VII']
chromatic_roots_counts_maj = []

chromatic_roots_min = ['i', 'bii', 'biii', 'iii', 'iv', 'bv', 'v', 'bvi', 'vi',
'bvii', 'vii']
chromatic_roots_counts_min = []

#counts instances of chords from chord_counts_dict
```

```
for i in range(len(chromatic_roots_maj)):
  chromatic_roots_counts_maj.append(chord_counts_dict[chromatic_roots_maj[i]])

#count number of chromatic roots

zero_counts = []
one_counts = []
two_counts = []
three_counts = []
four_counts = []
five_counts = []
six_counts = []
seven_counts = []
eight_counts = []
nine_counts = []
ten_counts = []
eleven_counts = []

for i in zero:
  zero_counts.append(dictionary_of_chords_and_counts[i])

for i in one:
  one_counts.append(dictionary_of_chords_and_counts[i])

for i in two:
  two_counts.append(dictionary_of_chords_and_counts[i])

for i in three:
  three_counts.append(dictionary_of_chords_and_counts[i])

for i in four:
  four_counts.append(dictionary_of_chords_and_counts[i])

for i in five:
  five_counts.append(dictionary_of_chords_and_counts[i])

for i in six:
  six_counts.append(dictionary_of_chords_and_counts[i])

for i in seven:
  seven_counts.append(dictionary_of_chords_and_counts[i])

for i in eight:
  eight_counts.append(dictionary_of_chords_and_counts[i])

for i in nine:
  nine_counts.append(dictionary_of_chords_and_counts[i])

for i in ten:
  ten_counts.append(dictionary_of_chords_and_counts[i])

for i in eleven:
```

```
  eleven_counts.append(dictionary_of_chords_and_counts[i])

zeros = sum(zero_counts)
ones = sum(one_counts)
twos = sum(two_counts)
threes = sum(three_counts)
fours = sum(four_counts)
fives = sum(five_counts)
sixes = sum(six_counts)
sevens = sum(seven_counts)
eights = sum(eight_counts)
nines = sum(nine_counts)
tens = sum(ten_counts)
elevens = sum(eleven_counts)



total_chromatic_roots = ['zero', 'one', 'two', 'three', 'four', 'five', 'six',
'seven', 'eight', 'nine', 'ten', 'eleven']

total_chromatic_roots_counts = []
total_chromatic_roots_counts.append(zeros)
total_chromatic_roots_counts.append(ones)
total_chromatic_roots_counts.append(twos)
total_chromatic_roots_counts.append(threes)
total_chromatic_roots_counts.append(fours)
total_chromatic_roots_counts.append(fives)
total_chromatic_roots_counts.append(sixes)
total_chromatic_roots_counts.append(sevens)
total_chromatic_roots_counts.append(eights)
total_chromatic_roots_counts.append(nines)
total_chromatic_roots_counts.append(tens)
total_chromatic_roots_counts.append(elevens)

total_chromatic_roots_counts

total_chromatic_roots

chromatic_roots_counts_maj

TandDC = [3058, 46, 336, 240, 174, 2104, 23, 1516, 372, 674, 748, 38]


plt.bar(total_chromatic_roots, total_chromatic_roots_counts, color='g')
plt.title("Frequency of Chromatic Roots in Expanded Rock Corpus")
plt.show()

plt.bar(chromatic_roots_maj, TandDC, color='b')
plt.title("Frequency of Chromatic Roots in TDC Rock Corpus")
plt.show()

"""#Bar Graph of number of unique chords in songs by decade - NOT CHROMATIC"""
```

```
307
308 #get top 10 overall chords in corpus
309 top_10_chord_dic
310 top_10_chords
311
312 from itertools import chain
313 from collections import Counter
314 import operator
315
316
317 #******* --> go through each df_1990s and get number of times one of the top 10
318 overall chords occurs
319
320
321
322 decades = [df_1940s, df_1950s, df_1960s, df_1970s, df_1980s, df_1990s, df_2000s]
323
324 dec = ['1940s', '1950s', '1960s', '1970s', '1980s', '1990s', '2000s']
325
326
327 count = [0,0,0,0,0,0,0,0,0,0]
328
329
330
331 for i in range(len(decades)):
332   count = [0,0,0,0,0,0,0,0,0,0]
333
334   for j in range(len(decades[i])):
335     for k in range(len(decades[i][j])):
336
337       if(decades[i][j][k] == 'I'):
338         count[0] = count[0] + 1
339
340       if(decades[i][j][k] == 'IV'):
341         count[1] = count[1] + 1
342
343       if(decades[i][j][k] == 'V'):
344         count[2] = count[2] + 1
345
346       if(decades[i][j][k] == 'bVII'):
347         count[3] = count[3] + 1
348
349       if(decades[i][j][k] == 'VI'):
350         count[4] = count[4] + 1
351
352       if(decades[i][j][k] == 'II'):
353         count[5] = count[5] + 1
354
355       if(decades[i][j][k] == 'III'):
356         count[6] = count[6] + 1
357
358       if(decades[i][j][k] == 'bVI'):
```

```
359         count[7] = count[7] + 1
360
361       if(decades[i][j][k] == 'bIII'):
362         count[8] = count[8] + 1
363
364       if(decades[i][j][k] == 'VII'):
365         count[9] = count[9] + 1
366
367   plt.bar(top_10_chords, count, color='g')
368
369   plt.title("Frequency of Top 10 Chord in the " + dec[i])
370   plt.show()
371
372 count_40 = [0,0,0,0,0,0,0,0,0,0]
373 count_50 = [0,0,0,0,0,0,0,0,0,0]
374 count_60 = [0,0,0,0,0,0,0,0,0,0]
375 count_70 = [0,0,0,0,0,0,0,0,0,0]
376 count_80 = [0,0,0,0,0,0,0,0,0,0]
377 count_90 = [0,0,0,0,0,0,0,0,0,0]
378 count_20 = [0,0,0,0,0,0,0,0,0,0]
379
380 count = [count_40, count_50, count_60, count_70, count_80, count_90, count_20]
381
382 decades = [df_1940s, df_1950s, df_1960s, df_1970s, df_1980s, df_1990s, df_2000s]
383
384 dec = ['1940s', '1950s', '1960s', '1970s', '1980s', '1990s', '2000s']
385
386
387 for i in range(len(decades)):
388   for j in range(len(decades[i])):
389     for k in range(len(decades[i][j])):
390
391       if(decades[i][j][k] == 'I'):
392         count[i][0] = count[i][0] + 1
393
394       if(decades[i][j][k] == 'IV'):
395         count[i][1] = count[i][1] + 1
396
397       if(decades[i][j][k] == 'V'):
398         count[i][2] = count[i][2] + 1
399
400       if(decades[i][j][k] == 'bVII'):
401         count[i][3] = count[i][3] + 1
402
403       if(decades[i][j][k] == 'VI'):
404         count[i][4] = count[i][4] + 1
405
406       if(decades[i][j][k] == 'II'):
407         count[i][5] = count[i][5] + 1
408
409       if(decades[i][j][k] == 'III'):
410         count[i][6] = count[i][6] + 1
```

```
411
412       if(decades[i][j][k] == 'bVI'):
413         count[i][7] = count[i][7] + 1
414
415       if(decades[i][j][k] == 'bIII'):
416         count[i][8] = count[i][8] + 1
417
418       if(decades[i][j][k] == 'VII'):
419         count[i][9] = count[i][9] + 1
420
421
422 for i in range(len(count)):
423   print(count[i])
424
425 total_1940s = sum(count_40)
426 total_1950s = sum(count_50)
427 total_1960s = sum(count_60)
428 total_1970s = sum(count_70)
429 total_1980s = sum(count_80)
430 total_1990s = sum(count_90)
431 total_2000s = sum(count_20)
432
433
434 count = [count_40, count_50, count_60, count_70, count_80, count_90, count_20]
435 totals = [total_1940s, total_1950s, total_1960s, total_1970s, total_1980s,
436 total_1990s, total_2000s]
437 percentages = []
438
439
440 for i in range(len(count)):
441   for j in range(len(count[i])):
442     count[i][j] = count[i][j] / totals[i]
443
444 for i in range(len(count)):
445   for j in range(len(count[i])):
446     count[i][j] = count[i][j]*100
447
448 count
449
450 for i in range(len(count)):
451
452   plt.bar(top_10_chords, count[i], color='g')
453   plt.title("Percentages of Top 10 Chord of: " + dec[i])
454   plt.show()
455
456 import numpy as np
457
458 def euclidean(v1, v2):
459     return sum((p-q)**2 for p, q in zip(v1, v2))/10 ** .5
460
461
462 euclid_dist_matrix = np.zeros((7,7))
```

```
463
464 for i in range(len(count)):
465   for j in range(len(count)):
466     if(i!=j):
467       euclid_dist_matrix[i][j] = euclidean(count[i], count[j])
468
469
470 euclid_dist_matrix
471
472 import seaborn as sns
473 import matplotlib.pyplot as plt
474
475 ax = sns.heatmap(euclid_dist_matrix, annot=True, xticklabels=dec, yticklabels=dec,
476 vmin=0, vmax=500, fmt='.3g')
477
478 plt.title("Euclidean Distance Between Histograms of Top 10 Chords in each Decade")
479 plt.show()
480
481 cg = sns.clustermap(euclid_dist_matrix, annot=True, xticklabels=dec,
482 yticklabels=dec, vmin=0, vmax=500, fmt='.3g')
483
484 plt.title("Clustermap of Similarity of Top 10 Chords in Corpus by Decade based on
485 Euclidean Distance between Histograms")
486
487 """#Chord Transition Matrix in Expanded Rock Corpus"""
488
489 analyses_Copy_original = analyses_Copy
490
491 """NLPTK to check if certain bigrams occur..."""
492
493 #Create a copy of analysis_Copy that just uses chromatic roots --> calculate bigrams
494 and trigrams from this
495 # ***** this will convert analyses_Copy to a list of chromatic numbers *****
496
497 for i in range(analyses_Copy.shape[0]):
498   for j in range(len(analyses_Copy[i])):
499     if analyses_Copy[i][j] in zero:
500       analyses_Copy[i][j] = 'I'
501     if analyses_Copy[i][j] in one:
502       analyses_Copy[i][j] = 'bII'
503     if analyses_Copy[i][j] in two:
504       analyses_Copy[i][j] = 'II'
505     if analyses_Copy[i][j] in three:
506       analyses_Copy[i][j] = 'bIII'
507     if analyses_Copy[i][j] in four:
508       analyses_Copy[i][j] = 'III'
509     if analyses_Copy[i][j] in five:
510       analyses_Copy[i][j] = 'IV'
511     if analyses_Copy[i][j] in six:
512       analyses_Copy[i][j] = 'bV'
513     if analyses_Copy[i][j] in seven:
514       analyses_Copy[i][j] = 'V'
```

```
    if analyses_Copy[i][j] in eight:
      analyses_Copy[i][j] = 'bVI'
    if analyses_Copy[i][j] in nine:
      analyses_Copy[i][j] = 'VI'
    if analyses_Copy[i][j] in ten:
      analyses_Copy[i][j] = 'bVII'
    if analyses_Copy[i][j] in eleven:
      analyses_Copy[i][j] = 'VII'

from nltk import bigrams
from collections import Counter

bgrms = []

for i in range(analyses_Copy.shape[0]):
  bgrms.append(list(bigrams(analyses_Copy[i])))

bgrms

import itertools

all_chromatic_bigrams = list(itertools.permutations(('I', 'bII', 'II', 'bIII',
'III', 'IV', 'bV', 'V', 'bVI', 'VI', 'bVII', 'VII'), 2))
all_chromatic_bigrams

import numpy as np
import operator
import itertools

Ant_Cons = pd.DataFrame(np.zeros((12,12)), columns = chromatic_roots_maj)
Ant_Cons.set_axis(chromatic_roots_maj, axis='index')

#create dictionary to store counts of bigrams
keyList = all_chromatic_bigrams
# Using Dictionary comprehension
bigram_count = {key: 0 for key in keyList}
print(bigram_count)


#sets the key = 0 for all chromatic bigrams

for i in range(len(bgrms)):
  for j in range(len(bgrms[i])):
    for k in range(len(all_chromatic_bigrams)):
      if(bgrms[i][j] == all_chromatic_bigrams[k]):
        bigram_count[all_chromatic_bigrams[k]] += 1

bigram_count

#counts of chroamtic bigrams

names = ['I', 'bII', 'II', 'bIII', 'III', 'IV', 'bV', 'V', 'bVI', 'VI', 'bVII',
```

```
        'VII']
numbers = [0,1,2,3,4,5,6,7,8,9,10,11]

dic = dict(zip(names, numbers))
dic

list(bigram_count.keys())[0][0]

for i in range(len(all_chromatic_bigrams)):
  Ant_Cons.at[dic.get(list(bigram_count.keys())[i][0]),
list(bigram_count.keys())[i][1]] = bigram_count.get(list(bigram_count.keys())[i])

# Commented out IPython magic to ensure Python compatibility.
import seaborn as sns
# %matplotlib inline

plt.figure(figsize=(15,15))
sns.heatmap(Ant_Cons, annot=True, fmt='g')
sns.heatmap(Ant_Cons, xticklabels=names, yticklabels=names)
plt.title('Chord Transitions in the Expanded Rock Corpus: (Rows --> Columns)')

"""#Got top bigrams to create tornado charts in excel"""

bigram_count

d = Counter (bigram_count)
total = sum(d.values())
d.most_common()

print('Total Number of Bigrams:', total)
for k, v in d.most_common(10):
  print('%s: %i' % (k, v))

bigram_count[('II', 'I')]

x = range(5)

top_bigram = ['I - IV', 'I - V', 'I - bVII', 'I - VI', 'I - II']
positive_data = [4813/37254, 2384/37254, 1513/37254, 1041/37254, 817/37254]
negative_data = [-4995/37254, -3324/37254, -1294/37254, -492/37254, -608/37254]

fig = plt.figure()
ax = plt.subplot(111)
ax.bar(x, negative_data, width=1, color='r')
ax.bar(x, positive_data, width=1, color='b')

plt.xticks(ticks = [0,1,2,3,4], labels = top_bigram, rotation = 'vertical')

plt.title('Percentage of Most Common Chromatic Bigrams - (37,254 Total Occurences)')
plt.xlabel('Bigrams')
plt.ylabel('Percentage of Total')

```

```
colors = {'Foward':'blue', 'Backward':'Red'}
labels = list(colors.keys())
handles = [plt.Rectangle((0,0),1,1, color=colors[label]) for label in labels]
plt.legend(handles, labels)

plt.show()

"""#Group data from each songs into segments of trigrams --> nested array

"""

# all_trigrams = []

# for i in range(len(analyses_Copy)):
#   for j in range(len(analyses_Copy[i])-3):
#     all_trigrams.append(analyses_Copy[i][j:j+3])

# all_trigrams

from nltk import trigrams
from collections import Counter

trigrms = []

for i in range(analyses_Copy.shape[0]):
  trigrms.append(list(trigrams(analyses_Copy[i])))

trigrms

all_chromatic_trigrams = list(itertools.permutations(('I', 'bII', 'II', 'bIII',
'III', 'IV', 'bV', 'V', 'bVI', 'VI', 'bVII', 'VII'), 3))
all_chromatic_trigrams

#create dictionary to store counts of bigrams
keylist = all_chromatic_trigrams
# Using Dictionary comprehension
trigram_count = {key: 0 for key in keylist}
print(trigram_count)

for i in range(len(trigrms)):
  for j in range(len(trigrms[i])):
    for k in range(len(all_chromatic_trigrams)):
      if(trigrms[i][j] == all_chromatic_trigrams[k]):
        trigram_count[all_chromatic_trigrams[k]] += 1

trigram_count

import heapq
top_trigrams = heapq.nlargest(20, trigram_count, key=trigram_count.get)
top_trigrams

top_trigrams_list = []
```

```
671
672 for i in range(len(top_trigrams)):
673   top_trigrams_list.append(str(top_trigrams[i]))
674
675 import numpy as np
676 import matplotlib.pyplot as plt
677
678 fig = plt.figure(figsize = (10, 5))
679
680 # creating the bar plot
681 plt.bar(top_trigrams_list, top_trigram_counts, color ='g',
682         width = 0.4)
683
684 plt.xlabel("Top 20 Trigrams")
685 plt.ylabel("Frequency")
686 plt.title("Frequency of Top 20 Trigrams of Chromatic Roots")
687 plt.xticks(rotation='vertical')
688 plt.show()
689
690 """#Get Histograms of Occurences of Chords / Bigrams for Each Decade --> Clustering
691 to Produce Histograms and track historical trends
692
693 #*** Everything Below is Chord Vectors****
694
695 #Correlation between chord vectors of all works
696 """
697
698 if('V' in analyses_Copy[1]):
699   print("yes")
700
701 chromatic_roots_maj = ['I', 'bII', 'II', 'bIII', 'III', 'IV', 'bV', 'V', 'bVI',
702 'VI', 'bVII', 'VII']
703
704 I = np.zeros(436)
705 bII = np.zeros(436)
706 II = np.zeros(436)
707 bIII = np.zeros(436)
708 III = np.zeros(436)
709 IV = np.zeros(436)
710 bV = np.zeros(436)
711 V = np.zeros(436)
712 bVI = np.zeros(436)
713 VI = np.zeros(436)
714 bVII = np.zeros(436)
715 VII = np.zeros(436)
716
717 for i in range(analyses_Copy.shape[0]):
718   if('I' in analyses_Copy[i]):
719     I[i] = 1
720   if('bII' in analyses_Copy[i]):
721     bII[i] = 1
722   if('II' in analyses_Copy[i]):
```

```
    II[i] = 1
  if('bIII' in analyses_Copy[i]):
    bIII[i] = 1
  if('III' in analyses_Copy[i]):
    III[i] = 1
  if('IV' in analyses_Copy[i]):
    IV[i] = 1
  if('bV' in analyses_Copy[i]):
    bV[i] = 1
  if('V' in analyses_Copy[i]):
    V[i] = 1
  if('bVI' in analyses_Copy[i]):
    bVI[i] = 1
  if('VI' in analyses_Copy[i]):
    VI[i] = 1
  if('bVII' in analyses_Copy[i]):
    bVII[i] = 1
  if('VII' in analyses_Copy[i]):
    VII[i] = 1

from string import ascii_letters
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_theme(style="white")
d = pd.DataFrame({'I' : I, 'bII': bII, 'II':II, 'bIII':bIII, 'III':III, 'IV':IV,
'bV':bV, 'V':V, 'bVI':bVI, 'VI':VI, 'bVII':bVII, 'VII':VII})

# Compute the correlation matrix
corr = d.corr()
# Generate a mask for the upper triangle
mask = np.triu(np.ones_like(corr, dtype=bool))
# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))
# Generate a custom diverging colormap
cmap = sns.diverging_palette(230, 20, as_cmap=True)
# Draw the heatmap with the mask and correct aspect ratio

corr = corr.replace(np.nan, 0)

sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
plt.title('Correlation between Chromatic Roots vectors (1940-2000)')

cg = sns.clustermap(corr, annot=True, xticklabels=chromatic_roots_maj,
yticklabels=chromatic_roots_maj, vmin=-1, vmax=1, fmt='.3g')

plt.title("Clustermap Between the Chromatic Chord Vectors")

"""#Redo Correlation matrix of chromatic roots"""
```

```
775
776 # import pandas as pd
777 # import seaborn as sns
778 # import matplotlib.pyplot as plt
779 # from sklearn.preprocessing import StandardScaler
780 # from sklearn.cluster import KMeans
781 # from sklearn.decomposition import PCA
782 # scaler =StandardScaler()
783
784
785 # features =scaler.fit(d)
786 # features =features.transform(d)
787
788 # # Convert to pandas Dataframe
789 # scaled_df =pd.DataFrame(features,columns=d.columns)
790 # # Print the scaled data
791 # scaled_df
792
793 # X=scaled_df.values
794
795 # wcss = {}
796 # for i in range(1, 11):
797 #     kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
798 #     kmeans.fit(X)
799 #     wcss[i] = kmeans.inertia_
800
801 # plt.plot(wcss.keys(), wcss.values(), 'gs-')
802 # plt.xlabel("Values of 'k'")
803 # plt.ylabel('WCSS')
804 # plt.show()
805
806 # kmeans=KMeans(n_clusters=8)
807 # kmeans.fit(X)
808
809 # dates
810
811 """#Perform Clustering on Chord Vectors"""
812
813 #Make chord vectors for songs based on the top 20 most common chords
814 #create 20 dim vectors of counts for top 20 chords in each song
815
816 songs = []
817
818 for i in range(analyses_Copy.shape[0]):
819   top = np.zeros(20)
820   for j in range(len(analyses_Copy[i])):
821       if(analyses_Copy[i][j] == 'I' ):
822         top[0] += 1
823       if(analyses_Copy[i][j] == 'IV' ):
824         top[1] += 1
825       if(analyses_Copy[i][j] == 'V' ):
826         top[2] += 1
```

```
827       if(analyses_Copy[i][j] == 'i' ):
828         top[3] += 1
829       if(analyses_Copy[i][j] == 'bVII' ):
830         top[4] += 1
831       if(analyses_Copy[i][j] == 'vi' ):
832         top[5] += 1
833       if(analyses_Copy[i][j] == 'bVI' ):
834         top[6] += 1
835       if(analyses_Copy[i][j] == 'ii' ):
836         top[7] += 1
837       if(analyses_Copy[i][j] == 'bIII' ):
838         top[8] += 1
839       if(analyses_Copy[i][j] == 'V7' ):
840         top[9] += 1
841       if(analyses_Copy[i][j] == 'III' ):
842         top[10] += 1
843       if(analyses_Copy[i][j] == 'iii' ):
844         top[11] += 1
845       if(analyses_Copy[i][j] == 'v' ):
846         top[12] += 1
847       if(analyses_Copy[i][j] == 'ii7' ):
848         top[13] += 1
849       if(analyses_Copy[i][j] == 'I7' ):
850         top[14] += 1
851       if(analyses_Copy[i][j] == 'iv' ):
852         top[15] += 1
853       if(analyses_Copy[i][j] == 'IV7' ):
854         top[16] += 1
855       if(analyses_Copy[i][j] == 'i7' ):
856         top[17] += 1
857       if(analyses_Copy[i][j] == 'vi7' ):
858         top[18] += 1
859       if(analyses_Copy[i][j] == 'IV6' ):
860         top[19] += 1
861   songs.append(top)
862
863 songs
864
865 import pandas as pd
866 import seaborn as sns
867 import matplotlib.pyplot as plt
868 from sklearn.preprocessing import StandardScaler
869 from sklearn.cluster import KMeans
870 from sklearn.decomposition import PCA
871
872 wcss = {}
873 for i in range(1, 10):
874     kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
875     kmeans.fit(songs)
876     wcss[i] = kmeans.inertia_
877
878 plt.plot(wcss.keys(), wcss.values(), 'gs-')
```

```
plt.xlabel("Values of 'k'")
plt.ylabel('WCSS')
plt.show()

kmeans=KMeans(n_clusters=3)
kmeans.fit(songs)

kmeans.cluster_centers_
kmeans.labels_

pca=PCA(n_components=2)

reduced_X=pd.DataFrame(data=pca.fit_transform(songs),columns=['PCA1','PCA2'])

centers=pca.transform(kmeans.cluster_centers_)

# Commented out IPython magic to ensure Python compatibility.
# %pip install adjustText

from adjustText import adjust_text

plt.rcParams.update({'font.size': 8})

plt.figure(figsize=(7,5))

# Scatter plot
plt.scatter(reduced_X['PCA1'],reduced_X['PCA2'],c=kmeans.labels_)
plt.scatter(centers[:,0],centers[:,1],marker='x',s=300,c='red')
plt.xlabel('PCA1')
plt.ylabel('PCA2')
plt.title('Song Clusters')


for i in range(len(songs)):
    plt.annotate(date[i], (reduced_X['PCA1'][i], reduced_X['PCA2'][i] + 0.2),
arrowprops={"arrowstyle":"->", "color":"gray"})


plt.rcParams.update({'font.size': 10})

pca.components_

component_df=pd.DataFrame(pca.components_,index=['PCA1',"PCA2"],columns=top_20_chord
s)
sns.heatmap(component_df)
plt.show()

"""#Turn training and testing data into from strings to encoded data:"""

!pip install Scikit-learn
import sklearn
from sklearn import preprocessing
```

```
top_50_chords = np.append(top_50_chords, 'other')
top_50_chords

top_10_chords = np.append(top_10_chords, 'other')
top_10_chords

le = preprocessing.LabelEncoder()
le.fit(top_10_chords)
list(le.classes_)

#transform training and testing data

train_data = le.transform(analyses_Copy) #***
train_data = np.expand_dims(train_data, axis=1)

test_data = le.transform(analyses_Copy)   #***
test_data = np.expand_dims(test_data, axis=1)

test_data.shape

total_train = np.squeeze(train_data)
total_test = np.squeeze(test_data)


total_data = np.concatenate((total_train, total_test), axis = 0)

total_data = np.expand_dims(total_data, axis=1)
```

7.6 Python Code to Find and Replace Algorithm

```
# -*- coding: utf-8 -*-
"""Copy of Roman Numerals --> Chords.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1_FS2SCKXl9SOw44MKbuKhx7jmm3AJcU2

#Load Rock Corpus Data
"""

from google.colab import files
uploaded = files.upload()

import pandas as pd
big_frame = pd.read_excel('Expanded Rock Corpus.xlsx', index_col=0)

```

```
# iterating the columns
# for col in big_frame.columns:
#     print(col)

big_frame['Harmonic Analysis'] = big_frame['Harmonic Analysis'].map(str)
big_frame['Harmonic Analysis'][2]

big_frame['Harmonic Analysis']

# analyses = []

# for i in range(len(big_frame['Harmonic Analysis'])):
#   analyses.append(big_frame['Harmonic Analysis'][i])

analyses = big_frame['Harmonic Analysis']

analyses_copy = analyses

analyses_Copy = analyses_copy
analyses_Copy

analyses_Copy.iloc[0]

analyses_Copy[2]

from functools import reduce

replacement_dict = {"[C]": "",
                    "[C#]": "",
                    "[Db]": "",
                    "[D]": "",
                    "[Eb]": "",
                    "[E]": "",
                    "[F]": "",
                    "[F#]": "",
                    "[G]": "",
                    "[G#]": "",
                    "[Ab]": "",
                    "[A]": "",
                    "[A#]": "",
                    "[Bb]": "",
                    "[B]": "",
                    "|": "",
                    ".": "",
                    "[4/4]": "",
                    "[2/4]": "",
                    "[0]": "",
                    "[R]": "",
                    "[12/8]": "",
                    "[7/8]": "",
                    "R": "",
                    "[3/4]": "",
```

```
                    "[6/8]": "",
                    "[5/4]": "",
                    "Warning: 'Vr5' is defined but never used": "",
                    "Warning: 'Ch5' is defined but never used": "",
                    "Warning:": "",
                    "Ch2": "",
                    "is": "",
                    "defined": "",
                    "but": "",
                    "never": "",
                    "used": ""

                    }



  # analyses_Copy[i] = analyses_Copy[i].replace("[C]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[C#]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[Db]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[D]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[Eb]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[E]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[F]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[F#]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[G]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[G#]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[Ab]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[A]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[A#]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[Bb]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[B]", "")

  # analyses_Copy[i] = analyses_Copy[i].replace("|", "")
  # analyses_Copy[i] = analyses_Copy[i].replace(".", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[4/4]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[2/4]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[0]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[R]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[12/8]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[7/8]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("R", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[3/4]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[6/8]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("[5/4]", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("Warning: 'Vr5' is defined but never
used", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("Warning: 'Ch5' is defined but never
used", "")

  # analyses_Copy[i] = analyses_Copy[i].replace("Warning:", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("Ch2", "")
  # analyses_Copy[i] = analyses_Copy[i].replace("Ch5", "")
```

```
122   # analyses_Copy[i] = analyses_Copy[i].replace("is", "")
123   # analyses_Copy[i] = analyses_Copy[i].replace("defined", "")
124   # analyses_Copy[i] = analyses_Copy[i].replace("but", "")
125   # analyses_Copy[i] = analyses_Copy[i].replace("never", "")
126   # analyses_Copy[i] = analyses_Copy[i].replace("used", "")
127
128
129
130
131
132 for i in range(len(analyses_Copy)):
133   analyses_Copy.iloc[i] = reduce(lambda x, y: x.replace(*y), [analyses_Copy.iloc[i],
134 *list(replacement_dict.items())])
135   analyses_Copy.iloc[i] = analyses_Copy.iloc[i].split()
136
137 print(analyses_Copy)
138
139 """#Determine frequency counts of symbols and build dictionary to replace on top 50 or
140 so...."""
141
142 from collections import Counter
143
144 c = Counter()
145
146 for d in analyses_Copy:
147   c.update(d)
148
149 for k, v in c.items():
150   print(f'{k} = {v}')
151
152 from itertools import chain
153 from collections import Counter
154 import operator
155
156 chord_counts_dict = dict(Counter(chain.from_iterable(analyses_Copy)))
157
158 print(chord_counts_dict)
159
160
161 most_frequent_chords = sorted(chord_counts_dict, key=chord_counts_dict.get,
162 reverse=True)
163
164 print(len(most_frequent_chords))
165
166 top_20_chords = most_frequent_chords[0:20]
167 top_20_chords
168
169 """#Now get top 50 Chords:"""
170
171 from itertools import chain
172 from collections import Counter
173 import operator
```

```
174
175 chord_counts_dict = dict(Counter(chain.from_iterable(analyses_Copy)))
176
177 print(chord_counts_dict)
178
179
180 most_frequent_chords = sorted(chord_counts_dict, key=chord_counts_dict.get,
181 reverse=True)
182
183 print(len(most_frequent_chords))
184
185 top_50_chords = most_frequent_chords[0:50]
186 top_50_chords
187
188 """#(Treat the chords that are not in the top 20 as in the "Other" Category)
189
190 #Go through each element in the list and see if it has a [X] character -- if it does,
191 switch to the [X] dictionary and start converting using that dictionary...
192 """
193
194 # #find and replace roman numerals with chords for determined keys
195
196 # C = {
197
198 # 'I': '0:',
199 #  'IV': '5:',
200 #  'V': '7:',
201 #  'i' : '0:m',
202 #  'bVII': '10:',
203 #  'vi': '9:m',
204 #  'bVI': '8:',
205 #  'ii': '2:',
206 #  'bIII': '3',
207 #  'iii': '3:m',
208 #  'iv': '5:',
209 #  'V7': '7:7',
210 #  'v': '7:m',
211 #  'IV6': '5:6',
212 #  'ii7': '2:m7',
213 #  'IV64': '5:6',
214 #  'I6': '0:6',
215 #  'I64': '0:6',
216 #  'V6': '7:6',
217 #  'vi7': '5:7'
218
219 #     }
220
221
222 # # dict_of_mappings = {"[C]": C, "[C#]": Cs, "[Db]": Db, "[D]": D, "[Eb]":Eb, "[E]":E
223 , "[F]":F, "[F#]": Fs, "[Gb]":Gb, "[G]":G, "[G#]":Gs, "[Ab]":Ab, "[A]":A, "[B]":B}
224
225
```

```
# chord_numbers = ['0:','5:','7:','0:m','10:','9:m','8:','2:',
'3:','3:m','5:','7:7','7:m',
# '5:6',
# '2:m7',
# '5:6',
# '0:6',
# '0:6',
# '7:6',
# '5:7']

# chord_numbers

"""#Update the the data with terms from the dictionary

"""

# for i in range(len(analyses_Copy)):
#   for j in range(len(analyses_Copy[i])):
#     for k in range(len(C)):
#       if(analyses_Copy[i][j] == list(C)[k]):
#         analyses_Copy[i][j] = list(C.values())[k]

# analyses_Copy

from collections import Counter

c = Counter()

for d in analyses_Copy:
  c.update(d)

for k, v in c.items():
  print(f'{k} = {v}')

"""#Create dictionary of top 50 chords and convert to numbers format...."""

top_20_chords

# D = {

#  'I': '0:',
#  'IV': '5:',
#  'V': '7:',
#  'i' : '0:m',
#  'bVII': '10:',
#  'vi': '9:m',
#  'bVI': '8:',
#  'ii': '2:',
#  'bIII': '3',
#  'V7': '7:7',
#  'III': '4:',
#  'iii': '4:m',
```

```
278 #  'v': '7:m',
279 #  'ii7': '2:m7',
280 #  'I7': '0:7',
281 #  'iv': '5:m',
282 #  'IV7': '5:7',
283 #  'i7': '0:m7',
284 #  'vi7': '9:m7',
285 #  'IV6': '5:6'
286
287
288 # }
289
290 D = {
291
292  'I': '0:maj',
293  'IV': '5:maj',
294  'V': '7:maj',
295  'i' : '0:min',
296  'bVII': '10:maj',
297  'vi': '9:min',
298  'bVI': '8:maj',
299  'ii': '2:maj',
300  'bIII': '3:maj',
301  'V7': '7:maj7',
302  'III': '4:maj',
303  'iii': '4:min',
304  'v': '7:min',
305  'ii7': '2:min7',
306  'I7': '0:maj7',
307  'iv': '5:min',
308  'IV7': '5:maj7',
309  'i7': '0:min7',
310  'vi7': '9:min7',
311  'IV6': '5:maj6'
312
313
314 }
315
316 #find and replace roman numerals with chords for determined keys
317
318
319
320 C = {
321
322 'I': '0:',
323  'IV': '5:',
324  'V': '7:',
325  'i' : '0:m',
326  'bVII': '10:',
327  'vi': '9:m',
328  'bVI': '8:',
329  'ii': '2:',
```

```
 'bIII': '3',
 'iii': '3:m',
 'iv': '5:m',
 'V7': '7:7',
 'v': '7:m',
 'IV6': '5:6',
 'ii7': '2:m7',
 'IV64': '5:6',
 'I6': '0:6',
 'I64': '0:6',
 'V6': '7:6',
 'vi7': '5:7',
 'IVd7': '5:dim7',
 'v7': '7:m7',
 'V11': '11:',
 'II': '2:',
 'iv6': '5:m6',
 'Id7': '0:dim7',
 'IV7': '5:7',
 'V42/IV':'0:42',
 'bII':'1:',
 'V+11':'7:11',
 'V/V':'2:',
 'vi64':'9:m64',
 'bVI6':'8:6',
 'bVII6':'10:6',
 'Id9':'0:dim9',
 'iv64':'5:m64',
 'V/vi':'4:',
 'Vs4':'7:sus4',
 'IV9':'5:9',
 'i7':'0:m7',
 'v7s4':'7:msus4',
 'V7/IV':'0:7',
 'bVII64':'10:64',
 'V7/ii':'9:7',
 'III':'4:',
 'V64':'7:64',
 'ii65':'2:m65',
 'V7/vi':'4:7',
 'I7':'0:7',
 'iii7':'3:m7'

}

import numpy as np
top50 = np.array(list(C.values()))
top20 = np.array(list(D.values()))

print(D.values())

for i in range(len(analyses_Copy)):
```

```
  for j in range(len(analyses_Copy.iloc[i])):
    for k in range(len(D)):
      if(analyses_Copy.iloc[i][j] == list(D)[k]):
        analyses_Copy.iloc[i][j] = list(D.values())[k]

analyses_Copy

"""#Export analyses_Copy which is a nested list into a series of .txt files

#(This gives a .txt files with all the songs)
"""

# with open('analyses_Copy_formatted.txt', 'w') as file:
#     for item in analyses_Copy:
#             file.write("// new_song.txt \n")
#             file.write(" \n".join(map(str,item)))
#             file.write("\n")

"""#Split up all the songs into train and test and make 2 .txt files"""

with open('Large_Train.txt', 'w') as file:
    for item in analyses_Copy:
            # file.write("// new_song.txt \n")
            file.write(" \n".join(map(str,item)))
            file.write("\n")

analyses_Copy_train = analyses_Copy.iloc[0:218]
analyses_Copy_test = analyses_Copy.iloc[218:436]

analyses_Copy.shape

with open('Rock_Train.txt', 'w') as file:
    for item in analyses_Copy_train:
            # file.write("// new_song.txt \n")
            file.write(" \n".join(map(str,item)))
            # file.write("\n")

with open('Rock_Test.txt', 'w') as file:
    for item in analyses_Copy_test:
            # file.write("// new_song.txt \n")
            file.write(" \n".join(map(str,item)))
            # file.write("\n")

with open('Symbol_Rock_20.txt', 'w') as file:
    for item in top20:
            # file.write("// new_song.txt \n")
            file.write("".join(map(str,item)))
            file.write("\n")
    file.write("end")

with open('Symbol_Rock_50.txt', 'w') as file:
    for item in top50:
```

```
        # file.write("// new_song.txt \n")
        file.write("".join(map(str,item)))
        file.write("\n")
file.write("end")
```

Bibliography

Burns, L. "Analytic Methodologies for Rock Music: Harmonic and Voice-Leading Strategies in Tori Amos's 'Crucify'." *In Expression in Pop-Rock Music: Critical and Analytical Essays*, 2nd ed., edited by W. Everett, Routledge, 2008.

Brown, Matthew. "Little Wing: A Study in Musical Cognition." *In Understanding Rock: Essays in Musical Analysis*, edited by John Covach and Graeme M. Boone, 199-215. New York: Oxford University Press, 1997.

Chomsky, Noam. "Three Models for the Description of Language," *IEEE Transactions on Information Theory*, vol. 2, no. 3 (1956): 113–124. https://doi.org/10.1109/tit.1956.1056813.

De Clercq, Trevor. "Computational Musicology in Rock," *The Bloomsbury Handbook of Rock Music Research*, (2020): 149–164. https://doi.org/10.5040/9781501330483.ch-010.

De Clercq, Trevor, and David Temperley. "A Corpus Analysis of Rock Harmony," *Popular Music*, vol. 30, no. 1 (2011): 47–70. Edited by Martin Cloonan and Sarah Hill. https://doi.org/10.1017/s026114301000067x.

Everett, Walter. "Making Sense of Rock's Tonal Systems," *Music Theory* Online, vol. 10, no. 4 (December 2004).

Everett, Walter. "Pitch Down the Middle." *In Expression in Pop-Rock Music: Critical and Analytical Essays*, 2nd ed., edited by Walter Everett, Routledge, 2008.

Granroth-Wilding, Mark, and Mark Steedman. “A Robust Parser-Interpreter for Jazz Chord Sequences.” *Journal of New Music Research*, vol. 43, no. 4, 2014, pp. 355–374, https://doi.org/10.1080/09298215.2014.910532.

Harasim, Daniel, Martin Rohrmeier, and Timothy J. O’Donnell. "A Generalized Parsing Framework for Generative Models of Harmonic Syntax." *In 19th International Society for Music Information Retrieval Conference*, Paris, France, 2018.

Jurafsky, Dan, and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Pearson Prentice Hall, 2009.

McDonald, Chris. “Exploring Modal Subversions in Alternative Music.” *Popular Music*, vol. 19, no. 3, 2000, pp. 355–363, https://doi.org/10.1017/s0261143000000210.

McHose, Allen Irvine. The Contrapuntal Harmonic Technique of the 18th Century. Eastman School of Music Series. Hardcover, January 1, 1947.

Nobile, Drew. “Harmonic Function in Rock Music.” *Journal of Music Theory*, vol. 60, no. 2, 2016, pp. 149–180, https://doi.org/10.1215/00222909-3651838.

Patel, Aniruddh D. "Syntax." *In Music, Language, and the Brain*, 239–298. Oxford: Oxford University Press, December 2007. https://doi.org/10.1093/acprof:oso/9780195123753.003.0005.

Pearce, Marcus, and Martin Rohrmeier. "Musical Syntax II: Empirical Perspectives." *Springer Handbook of Systematic Musicology*, 2018, pp. 487–505. Edited by Rolf Bader. https://doi.org/10.1007/978-3-662-55004-5_26.

Rohrmeier, Martin. "Towards a Generative Syntax of Tonal Harmony." *Journal of Mathematics and Music*, vol. 5, no. 1, (2011), pp. 35–53, https://doi.org/10.1080/17459737.2011.573676.

Rohrmeier, Martin, and Marcus Pearce. "Musical Syntax I: Theoretical Perspectives." *Springer Handbook of Systematic Musicology*, 2018, pp. 473–486. Edited by Rolf Bader. https://doi.org/10.1007/978-3-662-55004-5_25.

Shanahan, Daniel, John Ashley Burgoyne, and Ian Quinn, eds. *The Oxford Handbook of Music and Corpus Studies*. Oxford: Oxford University Press, 2022. https://doi.org/10.1093/oxfordhb/9780190945442.001.0001.

Temperley, David. "Harmonic Analyses." A Corpus Study of Rock Music, http://rockcorpus.midside.com/harmonic_analyses.html

Temperley, David. *The Musical Language of Rock*. Oxford University Press, 2018.

Temperley, David, and Trevor De Clercq. "Statistical Analysis of Harmony and Melody in Rock Music." *Journal of New Music Research*, vol. 42, no. 3, 2013, pp. 187–204, https://doi.org/10.1080/09298215.2013.788039.

Tsushima, Hiroaki, et al. “Generative Statistical Models with Self-Emergent Grammar of Chord Sequences.” *Journal of New Music Research*, vol. 47, no. 3, 2018, pp. 226–248, https://doi.org/10.1080/09298215.2018.1447584.

White, Christopher William. *The Music in the Data: Corpus Analysis, Music Analysis, and Tonal Traditions*. Routledge, 2023

White, Christopher William, and Ian Quinn. “Chord Context and Harmonic Function in Tonal Music.” *Music Theory Spectrum*, vol. 40, no. 2 (2018), https://doi.org/10.1093/mts/mty021.