A Probabilistic Automaton for Jazz Harmony

Julian Rosenblum

December 9, 2017

Abstract

In this paper, we define a computational model for analyzing and generating progressions of chords in jazz. We accomplish this by combining jazz harmonic theory with machine learning techniques. First, we define a model that produces a set of functional analyses for a given chord progression. Then, we train the model on a corpus of jazz standards to produce probabilities for transitions between particular chord functions. Finally, using these probabilities, we can generate new chord progressions stochastically.

1 Introduciton

In a broad sense, our goal is to create a model that understands jazz. Our definition of "understand" in this context is two-fold: our model needs to be able to produce jazz and it needs to provide some sort of explanation about what makes a particular piece of music jazz. Because of how complex, intricate, and difficult to define jazz is, this goal is quite lofty, and so we will restrict our model to only dealing with the harmonic progression of a piece. As a further simplification, we will also ignore durations when considering harmonic progression. Ignoring essential features such as melody, rhythm, voicing, instrumentation, etc., allows us to extract a finite, definable aspect of jazz from which we can draw meaningful conclusions. In addition, while songs with the same chord progression or even different renditions of the same song may differ vastly, our reduction allows us to gain insight about the commonalities that bring them together.

In order for our model to understand jazz harmony, we must employ a combination of music theory and machine learning. One could approach a project like this without music theory by, for example, creating an n-gram model and training it on chord progressions. This approach, however, fails our definition of "understand", because while it can tell us the likelihood of a particular chord progression, it provides no meaningful information to a jazz musician about why that progression works. We are also faced with the issue that we simply don't have enough training data to rely on machine learning alone. Alternatively, we could approach a project like this using only music theory and no data. In this case, we run into the problem that in music, particularly in jazz, theory

is simply an attempt to explain practice. Theory gives us parameters through which we can understand what jazz musicians play and attempt to emulate it, but it does not and cannot provide the full story on its own. Thus, we will construct a model based in theory that will then learn how to apply that theory idiomatically based on practice.

As we construct this model, we will attempt to strike a balance between adding sufficient features to the model and making the model restrictive enough to still be useful. If we are too conservative with the model, we risk modeling a subset of jazz harmony that is too small to produce interesting results. If we are not conservative enough, we risk accommodating so much that the results of the model are not meaningful. In striking this balance, we aim to be able to generate sequences that are not obvious or formulaic but that are still reasonable within the jazz idiom. This goal is aesthetic rather than mathematical, although we will discuss how to quantify it in section 3.2.

Lastly, on an ethical note, the goal of this project is not to replace human musicians with computers. We do not believe that computers will eventually surpass the artistic capabilities of humans, nor that they should. Instead, we seek to use this model as a means of both formalizing and learning from what humans have already created.

2 Defining the Model

2.1 Harmonic Language

For simplicity, we will reduce our harmonic language to a set of 72 symbols. These symbols represent chords relative to a given key. Our language is based on the Mehegan sixty chord system (Mehegan, 1992) but includes an additional chord quality for suspended chords. We define our language as follows.

A symbol c consists of a scale degree (represented in roman numeral form with an optional $\sharp$ or $\flat$ alteration) and a quality $q \in \{\texttt{M}, \texttt{m}, \texttt{x}, \texttt{o}, \texttt{ø}, \texttt{s}\}$. Symbols with enharmonic scale degrees are considered equivalent. For simplicity, we ignore inversions. Chord qualities are defined as follows.

- A chord with major quality (`M`) has a major third, either a perfect fifth or no fifth, and either a major seventh or no seventh.
- A chord with minor quality (`m`) has a minor third, either a perfect fifth or no fifth, and either a minor seventh or no seventh.
- A chord with dominant quality (`x`) has a major third and a minor seventh.
- A chord with diminished quality (`o`) has a minor third, a diminished fifth, and either a diminished seventh or no seventh.

- A chord with half-diminished quality (ø) has a minor third, a diminished fifth, and a minor seventh.
- A chord with suspended quality (`s`) has no third, a perfect fifth, and either a perfect fourth or a perfect second.

We notate a chord symbol by concatenating the scale degree and quality. For example, a dominant quality chord based on the flatted third scale degree would be written as `♭IIIx`. If the chord quality is omitted, the quality is inferred from the diatonic seventh chord on that scale degree (so `I` is a major chord and `V` is a dominant chord). Alterations to the scale degree do not affect the diatonic quality. By convention, minor, diminished, and half-diminished chords are often notated with lowercase numerals.

2.2 Automaton

We define the automaton $A = (S, T)$ where S is a set of states and T is a set of transitions $t = (s_{from}, c, s_{to})$ where a chord symbol c acts as a transition from state s_{from} to state s_{to}. In our automaton, we represent *chord functions* as *states* and *chord symbols* as *transitions*. An *analysis* of a sequence of chord symbols is a sequence of corresponding transitions that represent a valid path through the automaton. If we have a transition $t = (s_0, c, s_1)$, we say that the chord c has the function s_1. In other words, the function of a chord symbol in an analysis is determined by the state that the chord is used to transition to.

We will build up these states and transitions by starting with certain hard-coded chords and functions (henceforth referred to as *primitive harmonic material*) and then applying higher-order operations to the automaton to take into account different features of jazz harmony.

2.3 Operational Approach

Our approach of starting with primitive harmonic material and then applying higher-order operations has a philosophical and theoretical grounding in jazz. Jazz is unique in that different artists can have vastly different interpretations of the same song. In fact, it is often unclear what actually defines the song itself, since virtually every aspect of it can be changed in different renditions. In the world of jazz harmony (which we have confined our analysis to), there is a notion of reharmonization. Because we can compare different artists' takes on the same jazz standard, we have a good idea of the different reharmonizations (i.e. transformations of the original harmonic material) that are canonical in jazz. Thus, it makes sense to implement these reharmonization rules as operations on an automaton as opposed to individual states and transitions. These operations will add new pathways through the automaton based on reharmonizations of preexisting ones.

The challenge is then to define the primitive harmonic material to which we will apply reharmonization operations. In general, we will try to minimize our primitive harmonic material to make as much of the model derivable from reharmonization operations as possible. Practically, we want to derive our primitive harmonic material from a system that is relatively simple and easy to express as states and transitions. Aesthetically, we want this system to encompass enough that the music we eventually generate is interesting, but not so much that the music we generate sounds like it could have been generated randomly.

2.4 Primitive Harmonic Material

The system we will use for our primitive harmonic material comes from the theory of functional-bass analysis, which was developed by Ian Quinn and adapted for jazz harmony by Brian Kane. The idea is to group chords based on both their function (one of *tonic*, *subdominant*[1], and *dominant*) and their bass (which in our case is the same as their root because we ignore inversions). The system places an emphasis on bass and function over chord quality, which is particularly relevant in jazz where chord qualities are often subject to reharmonization. For example, the distinction between `IIm-Vx` and `IIx-Vx` is negligible. The meaning of the first chord comes from the fact that it is a `II` chord that precedes a `V` chord.

Thus, we will be grouping chords by bass and function. The possible chord functions for a particular bass note come from the functions of the diatonic chords built off of its scale degree (Terefenko, 2014, p.27). Thus, we get the following possible roots for each function:

- Tonic: 1, 3, 6
- Subdominant: 2, 4, 6
- Dominant: 3, 5, 7

The functions for ♭`III`, ♭`VI`, and ♭`VII` come from the parallel minor. We will also impose the additional property that for a flatted scale degree to have a particular function, the scale degree a tritone apart must also have that function. This keeps the categories balanced and also prevents us from getting weird results when performing tritone substitutions (see section 2.5.3). Thus, ♭`III` is a tonic chord, ♭`VI` is a subdominant chord, and ♭`VII` is a dominant chord.

Our primitive harmonic material will exclude diminished chords and suspended chords, because those will be derived later from reharmonizations.

For tonic and subdominant chords, we allow any primitive chord quality (major, minor, dominant, half-diminished). For example, `I`ø is a tonic chord, and `VIx` can be either a tonic or a subdominant chord.

[1] We use *subdominant* to refer to what Terefenko calls *predominant*.

For dominant chords, we will hand-pick more arbitrarily, since the dominant function in jazz is very particular. Dominant chords will include the following: `IIIm`, `IIIx`, `Vx`, `bVIIx`. `IIIm` and `Vx` are diatonically dominant chords. `bVIIx` is used as part of a subtonic cadence (also known as a backdoor progression). `IIIx` is an alteration of `IIIm` that also happens to be a tritone substitution of `bVIIx`. Lastly, note that `VIIo` is not listed, because it will be derived using the rules we will establish for diminished chords (see section 2.5.2).

Henceforth, we will consider functional-bass pairings to be chord functions (i.e., states) in our automaton. To avoid overloading the term "function", we will now refer to tonic, subdominant, and dominant as *function groups*.

Now, based on the above principles, we have the following chord functions for our primitive harmonic material:

- Tonic$_1$: `IM`, `Im`, `Ix`, `Iø`
- Tonic$_{\flat 3}$: `♭IIIM`, `♭IIIm`, `♭IIIx`, `♭IIIø`
- Tonic$_3$: `IIIM`, `IIIm`, `IIIx`, `IIIø`
- Tonic$_6$: `VIM`, `VIm`, `VIx`, `VIø`
- Subdominant$_2$: `IIM`, `IIm`, `IIx`, `IIø`
- Subdominant$_4$: `IVM`, `IVm`, `IVx`, `IVø`
- Subdominant$_{\flat 6}$: `♭VIM`, `♭VIm`, `♭VIx`, `♭VIø`
- Subdominant$_6$: `VIM`, `VIm`, `VIx`, `VIø`
- Dominant$_3$: `IIIm`, `IIIx`
- Dominant$_5$: `Vx`
- Dominant$_{\flat 7}$: `♭VIIx`

We will now add these chord symbols to the automaton based on the following rules about progression among the three function groups. We will define a set of progression rules P consisting of entries in the form $F \Rightarrow G$ where F and G are function groups, and then construct the set of transitions T as follows:

$$T = \{(F_i, c, G_j) : (F \Rightarrow G) \in P \wedge c \in G_j\}$$

For example, if we allow progression from Tonic $\Rightarrow$ Subdominant, we add a transition from each tonic state to each subdominant state for each symbol in that particular subdominant chord function.

Finally, we will add the following progressions rules to P to complete the construction of our primitive harmonic material.

- Tonic ⇒ Subdominant
- Subdominant ⇒ Dominant
- Dominant ⇒ Tonic
- Tonic ⇒ Tonic
- Subdominant ⇒ Subdominant
- Dominant ⇒ Dominant

2.5 Reharmonization Operations

We will now define reharmonization operations that we will perform on the automaton. In general, an operation will be performed on transitions that meet certain conditions, and add new states and transitions to the automaton based on each transition it was applied to.

Formally, a reharmonization operation $R(A)$ on an automaton $A = (S, T)$ consists of a condition $C(t)$ and a reharmonization function $r(t)$ where $t = (X, c, Y)$ is a transition and returns a set of new states $r(t)_S$ and a set of new transitions $r(t)_T$. We define $R(A)$ as follows:

$$R(A) = (S \cup \bigcup_{t \in T \wedge C(t)} r(t)_S, T \cup \bigcup_{t \in T \wedge C(t)} r(t)_T)$$

In general, we can infer $r(t)_S$ from $r(t)_T$, so we will represent $r(t)$ as $r(t)_T$.

Symbol Transformation Notation Since most transitions on $r(t)$ will be some sort of transposition and/or chord quality alteration of the input transition symbol c, we will adopt some notational shorthand to represent these transformations:

- c_q refers to the chord specified by symbol c but with the quality q. For example, if c is `IM`, $c_{\texttt{x}}$ is `Ix`.
- c^i refers to the chord specified by symbol c transposed up by the interval i. For example, if c is `IM`, c^{P4} is `IVM`.

We will now define our reharmonization operations:

2.5.1 Suspended Chords

Condition c has dominant quality ($q(c) = \texttt{x}$).

Output transitions

- $(X, c_{\texttt{s}}, Y)$

Rationale Suspended chords (or sus chords) are typically used to reharmonize `V` chords and `ii-V` progressions (Levine, 1995, p.262). We can extend this idea to allow any dominant chord to be suspended and let the data tell us how idiomatic different sus chords actually are.

2.5.2 Diminished Chords

(1)

Condition c has dominant quality ($q(c) = \texttt{x}$).

Output transitions

- $(X, c_{\texttt{o}}^{\text{M3}}, Y)$

Rationale Because a diminished chord shares many notes with four different dominant b9 chords, it could be substituted for any of them (Levine, 1995, p.336). However, we pick the dominant chord a major third below, because it comes from the relationship between `V` and `vii` in classical harmony.

(2)

Condition c has minor quality ($q(c) = \texttt{m}$).

Output transitions

- $(X, c_{\texttt{o}}^{\text{m2}}, S_{\text{dim}})$
- (S_{dim}, c, Y)

Rationale Preceding a minor chord with a diminished chord a half-step above works because the two chords share many notes. This progression appears in many standards such as "Wave," "All of Me," and "I Can't Get Started."

2.5.3 Tritone Substitution

Condition c has dominant quality ($q(c) = \texttt{x}$) and there exists no transition $(X', c^{\text{dim5}}, Y') \in T$ such that Y and Y' are in the same function group.[2]

[2]This second condition reduces redundancy in our analyses. For example, since 3 and ♭6 are both tonic chords, the result of performing a tritone substitution on either of them is already present in our primitive harmonic material. In other words, there is no need to call a `♭VIx` chord a tritone subbed Tonic 3 since we can just call it a Tonic ♭6.

Output transitions

- (X, c^{dim5}, Y)

Rationale The tritone substitution is one of the most popular reharmonizations in jazz. It works harmonically, because for two dominant seventh chords a tritone apart, the third of one is the seventh of the other and vice-versa (Levine, 1995, p.262).

2.5.4 Applied Dominants

Condition c has major or minor quality ($q(c) \in \{\texttt{M}, \texttt{m}\}$).

Output transitions

- $(X, c^{\text{P5}}_{\texttt{x}}, S_{\text{app}})$
- (S_{app}, c, Y)

Rationale Any major or minor chord can be approached by its `V` chord (Levine, 1995, p.336).

2.5.5 Chromatic Approaching Chords

Condition c has major or minor quality ($q(c) \in \{\texttt{M}, \texttt{m}\}$).

Output transitions

- $(X, c^{\text{M7}}_{\texttt{x}}, S_{\text{chr}})$
- (S_{chr}, c, Y)

Rationale Any major or minor chord can be approached by a dominant chord a half-step above or below it (Levine, 1995, p.331). The approach from above will already be produced as an applied dominant that has been tritone subbed, so we only need to add the approach from below.

2.5.6 Tonicization

Condition c has major, minor, or half-diminished quality and it's numeral is not `I` ($q(c) \in \{\texttt{M}, \texttt{m}, \emptyset\} \wedge \text{num}(c) \neq \texttt{I}$).

Output transitions

- $(X, c^{\mathrm{M2}}_{\mathtt{m}}, S_{\mathrm{ii}})$
- $(X, c^{\mathrm{M2}}_{\mathtt{ø}}, S_{\mathrm{ii}})$
- $(X, c^{\mathrm{M2}}_{\mathtt{x}}, S_{\mathrm{ii}})$
- $(S_{\mathrm{ii}}, c^{\mathrm{P5}}_{\mathtt{x}}, S_{\mathrm{V}})$
- (S_{V}, c, Y)

Rationale The arrival of a major or minor chord can be set up with a `ii-V-I` or `iiø-V-i` progression in that key (Terefenko, 2014, p.367). However, in keeping with the spirit of functional-bass theory, we will extend the chord qualities that are allowed, as it is the progression of bass notes that is fundamental, and we can let the song data determine the probabilities of different qualities. We also allow half-diminished chords to be tonicized to accommodate for sequences like `iiiø-VIx-iiø` that might be part of a larger sequence setting up the tonic.

2.5.7 Unpacked Chords

(1)

Condition c has dominant quality ($q(c) = \mathtt{x}$).

Output transitions

- $(X, c^{\mathrm{P5}}_{\mathtt{m}}, S_{\mathrm{unp}})$
- (S_{unp}, c, Y)

Rationale Because `ii-V` progressions are so iconic, any dominant chord can be preceded by its corresponding `ii` chord (Levine, 1995, p.260). We will refer to this technique as unpacking.

(2)

Condition c has minor quality ($q(c) = \mathtt{m}$).

Output transitions

- (X, c, S_{unp})
- $(S_{\mathrm{unp}}, c^{\mathrm{P4}}_{\mathtt{x}}, Y)$

Rationale Alternatively, minor chords can also be unpacked by placing the corresponding `V` chord after. Occasionally, a `ii-V`-like sequence can also be "packed," replacing the whole thing with the `ii` chord, but allowing for this in general opens too many possibilities that we don't want for the model, so instead, we just special-case the packed `IVm` by making it a dominant chord.

2.5.8 Neighbor Chords

Condition c has major, minor, or dominant quality ($q(c) \in \{\texttt{M}, \texttt{m}, \texttt{x}\}$).

Output transitions

- (X, c, S_{nei1})
- $\{(S_{\text{nei1}}, c', S_{\text{nei2}}) \mid c' \in C\}$ where C is the set of all chord symbols.
- (S_{nei2}, c, Y)

Rationale Neighbor chords go in between two chords that are the same and serve as a means of emphasizing the chord. Since we have data to tell us what chords work well as neighbor chord, our model can allow any chord to serve as a neighbor.

2.5.9 Diatonic Passing Chords

The operation for diatonic passing chords is somewhat more complicated to define. Consider this ordered sequence of diatonic chords:

$$DS = \texttt{I}, \texttt{ii}, \texttt{iii}, \texttt{IV}, \texttt{V}, \texttt{vi}, \texttt{vii}, \texttt{I}$$

. We will define a set of passing sequences $PS = \{(c_1, c_2, c_3) : (c_1, c_2, c_3) \subset DS \vee (c_3, c_2, c_1) \subset DS\}$.[3] For example, $\texttt{ii}, \texttt{iii}, \texttt{IV}$ and $\texttt{IV}, \texttt{iii}, \texttt{ii}$ are both subdominant passing sequences.

Condition c is part of a passing sequence (a, b, c) for function group F and Y is a state in function group F. More formally,

$$\exists (a, b, c) \in PS : \exists (X_a, a, Y_a), (X_c, c, Y_c) \in T : Y_a, Y_c \in F$$

Output transitions

- (X, a, S_{pas1})
- $(S_{\text{pas1}}, b, S_{\text{pas2}})$
- (S_{pas2}, c, Y)

[3] We use the $\subset$ symbol here to represent subsequence, not subset.

Rationale When moving between chords of the same function, a passing chord can be inserted to create an ascending or descending bassline progression. For simplicity, we will only allow for diatonic passing chords.

2.5.10 Order of Reharmonization Operations

We apply the above reharmonization operations in the following order:

- Tonicization
- Applied Chords
- Diminished Chords
- Tritone Substitutions
- Unpacked Chords
- Suspended Chords
- Chromatic Approaching Chords
- Neighbor Chords
- Diatonic Passing Chords

3 Using Song Data

3.1 Corpus

For this project, we use the iRb Corpus, a corpus of nearly 1200 jazz standards made available for music research by The Ohio State University's Cognitive and Systematic Musicology Laboratory. The corpus contains chord progressions for jazz standards as one would find in a fake book, but in a more readily parsable `.jazz` format.

As jazz standards often have song forms that have repeats, `.jazz` files divide songs into sections and then specify the order of the sections, for example, $AABA$. When interacting with the model, we often examine these sections separately. However, there is some inconsistency surrounding what constitutes a section. For example, two A sections that differ in the last bar or two might be treated as different sections, or the parts that are the same might be considered the A section with the differing bars treated as separate sections.

3.2 Evaluating the Model

Up until now, we have defined our model independently of the training data. But before we train the model, we want some sort of indication that our automaton is a reasonable model for jazz harmony, and we can use the training data to evaluate this.

As we discussed before, we want our model to strike a balance between being flexible enough to represent the vast world of jazz harmony and being restrictive enough to still be interesting. To quantify this, we will construct something like a classification problem. When we run a chord progression through the model, there will either be a valid pathway of transitions or there won't. This lets us classify a chord progression as being jazz or not.

We run into an issue that while we have some data on what is jazz, we have no data on what is not jazz. To get around this, we will evaluate our model by comparing how much of the training data it classifies as jazz compared to randomly generated sequences of chords. We want our model to have a classify a large percentage of real jazz songs as jazz while classifying a small percentage of randomly generated songs as jazz.

In our comparison, we also want to make sure that the comparison between real jazz and random chords is fair. We will use individual sections instead of whole songs to reduce the impact of repetition in real jazz. We will also generate sequences of random chords such that the distribution of sequence lengths is the same as the actual distribution of section lengths in the corpus (Table 1). The results of our model are below. We include the model's performance on entire songs from the corpus as a point of interest, even though it is not particularly significant mathematically.

Real jazz (sections, 4242 count)	58.27%
Real jazz (songs, 1185 count)	27.17%
Random chords (sections, 5000 count)	7.5%

The important takeaway here is that our model is about eight times more likely to classify a section of a real jazz standard as jazz than a sequence of random chords of the same length. While it may seem concerning that the model's performance on real jazz isn't higher, recall that the end goal of our model is not to classify jazz (a nearly impossible task), but to understand jazz harmony, or at least some definable subset of it.

3.3 Training the Model

We will now extend our automaton model to have probabilities associated with each transition. To do this, we will count how many times transitions are taken as we train the model on sequences of chords. We can then compute probabilities based on these counts. If a transition is never taken in training the model, we will assign it a count (and also a probability) of zero.

We will only train on sequences with at least one valid path through the automaton. However, because our states represent chord functions and our data is not tagged with the corresponding chord functions for each chord, it is

Table 1: Distribution of section sizes in corpus

Section size	# of sections
2	258
3	241
4	323
5	81
6	94
7	68
8	445
9	263
10	283
11	215
12	339
13	207
14	241
15	189
16	325
17	150
18	118
19	80
20	71
21	52
22	29
23	39
24	33
25	19
26	9
27	9
28	12
29	9
30	5
31	4
32	12
33	6
34	4
35	3
38	1
39	1
40	2
41	2

very likely that a given training sequence will have multiple valid paths through the automaton. This is reasonable, because for a given sequence of chords, there will very likely be different valid theoretical interpretations of the chord functions.

We don't want our counts to be skewed based on the number of possible paths for a given sequence, so we will use pseudocounts that can have non-integer values. For a sequence of length n, the total amount that will be added to various transition counts will be n, regardless of the number of paths. For each chord, we will divide up a pseudocount of 1 among the different possible transitions for that chord. For simplicity, we will divide it evenly so that for a given chord in the sequence, each possible transition gets the same amount. This ignores the fact that some transitions may appear in more pathways than others, but has the benefit that the probabilities for a particular chord are not influenced by other chords in the sequence. In other words, the fact that a later chord creates more pathways should not influence the pseudocounts assigned to previous chords.

Formally, we train a sequence as follows. Let $k(t)$ be the pseudocount for transition t. Given a sequence of chords $c_1, c_2, \ldots, c_n$, we will generate a set of pathways P such that each pathway is a sequence of transitions corresponding to the sequence of chords. Let t_{ij} be the jth transition of the ith pathway. We update[4] transition pseudocounts as follows:

$$k(t_{ij}) = k(t_{ij}) + \frac{1}{|\{u \mid \exists x : u = t_{xj}, 1 \leq x \leq |P|\}|}$$

3.4 Generated Chord Progressions

Once the model has been trained, we can use it to generate chord progressions by giving it certain starting and stopping criteria. In this section, we will talk about some sequences that were generated by starting on a `I` chord, taking transitions based on their probabilities, and stopping once another `I` chord is reached. For each transition we take, we consider the symbol and destination state together to be a chord / function pairing, which we will then list out. Each example will have Mehegan symbols, chord functions, and also corresponding chord symbols in the key of C. All of these examples were actually generated by the model. We've selected a few that exhibit interesting attributes of the model.

[4] If two transitions t_{ij} and $t_{i'j}$ are the same, we only perform the update once.

3.4.1

I	CΔ7	Tonic 1
vi	Am7	Tonic 6
ii	Dm7	Subdominant 2
V	G7	Dominant 5
I	CΔ7	Tonic 1

The vi-ii-V-I sequence is one of the most iconic progressions in jazz, so it's good that our model generates it. Also note that there are other possible chord functions that could have been generated for the same sequence. For example, the vi could also be a subdominant chord, and the ii could be part of an unpacked V chord.

3.4.2

I	CΔ7	Tonic 1
IVm	Fm7	ii / bIIIM
bVIIx	Bb7	V / bIIIM
bIIIM	EbΔ7	Tonic b3
bVIx	Ab7	Subdominant b6
V	G7	Dominant 5
I	CΔ7	Tonic 1

This example demonstrates two strengths of our model. First, the model understands that when it takes the ivm transition, it is as a means of tonicizing bIIIM. However, this tonicization sequence is itself also subject to variation. Inspecting the probabilities of the model tells us that from the ivm, there was an .887 probability of going to bVIIx, a .106 probability of using a tritone substitution IIIx), and a .07 probability of using a suspended chord bVIIs.

Another notable progression here is bIIIM-bVIx-V. This progression seems might seem a bit strange to a jazz musician, and we can quantify this. In the corpus, $P(\texttt{bVIx}, \texttt{V} \mid \texttt{bIIIM})$ (the probability of bIIIM being followed by bVIx and V) is .0063. However, in our model, $P(\texttt{bVIx}, \texttt{V} | \texttt{bIIIM})$ is .043, nearly 7 times higher. Aesthetically, this demonstrates how our model is flexible enough to explore less obvious possibilities. While this progression is not very common in practice, it does make musical sense: the bVIx acts as a pivot chord, serving as a IVx chord in the key of Eb and also as an applied dominant of V in the key of C.

3.4.3

`I`	CΔ7	Tonic with passing chord
`ii`	Dm7	Passing chord
`iii`	Em7	Tonic 3
`IVm`	Fm7	IVm with neighbor chord
`bVIIx`	Bb7	Neighbor of IVm
`IVm`	Fm7	Subdominant 4
`bVIIx`	Bb7	Dominant b7
`bIIx`	Db7	V / Im
`Im`	Cm7	Tonic 1
`Vm`	Gm7	ii / IVM
`Ix`	C7	V / IVM
`IV`	FΔ7	Subdominant 4
`VIIx`	B7	Subdominant 4
`bVIIx`	Bb7	Dominant b7
`I`	CΔ7	Tonic 1

This example shows off how elaborate the progressions generated by the model can be. This progression has neighbor chords, passing chords, tonicization, applied dominants, and tritone substitutions. However, underneath all these embellishments, the general motion from tonic to subdominant to dominant is still clear.

Another thing to note is that some of the function labels appear incorrect (e.g. `bIIx` as "V / Im" and `VIIx` as "Subdominant 4"). This is because these transitions were created as the result of the tritone substitution operation, but the name of the destination state didn't change. In the future, we hope to provide clearer naming in these sorts of cases, however the current naming does reveal to an extent the way in which reharmonization techniques are used.

3.4.4

To help get a sense of what the model generates, we generated 10,000 progressions from `I` to `I`. The most common results are listed in Table 2.

4 Conclusion

The results of the model show that we have made great progress in teaching a computer to understand jazz. There is a lot of work still to be done, which we will divide into three categories.

4.1 Issues with the Current Model

Chord Function Naming As mentioned in the previous section, our chord function naming could be improved. Currently, the name of a chord function

Table 2: Most common `I` to `I` progressions generated by model, written out in the key of C

Progression	# of occurrences (out of 10,000)
CΔ7,Dm7,G7,CΔ7	516
CΔ7,A7,Dm7,G7,CΔ7	271
CΔ7,Am7,Dm7,G7,CΔ7	222
CΔ7,G7,CΔ7	192
CΔ7,Ebo7,Dm7,G7,CΔ7	125
CΔ7,Dø,G7,CΔ7	107
CΔ7,Am7,D7,G7,CΔ7	106
CΔ7,D7,G7,CΔ7	99
CΔ7,F7,CΔ7	67
CΔ7,A7,D7,G7,CΔ7	62
CΔ7,C#o7,Dm7,G7,CΔ7	59
CΔ7	54
CΔ7,Bø,E7,Am7,Dm7,G7,CΔ7	53
CΔ7,Em7,A7,Dm7,G7,CΔ7	52
CΔ7,E7,A7,Dm7,G7,CΔ7	44
CΔ7,FΔ7,Bb7,CΔ7	43
CΔ7,Ab7,G7,CΔ7	32
CΔ7,G7sus,CΔ7	31
CΔ7,F7,Em7,A7,Dm7,G7,CΔ7	29
CΔ7,Am7,G7,CΔ7	29
CΔ7,Aø,Dm7,G7,CΔ7	28
CΔ7,Fm7,CΔ7	26
CΔ7,F7,Bb7,CΔ7	25
CΔ7,E7,Am7,Dm7,G7,CΔ7	24
CΔ7,FΔ7,CΔ7	24
CΔ7,FΔ7,G7,CΔ7	24
CΔ7,D7,Dm7,G7,CΔ7	22
CΔ7,Eø,A7,Dm7,G7,CΔ7	22
CΔ7,FΔ7,Fm7,CΔ7	21
CΔ7,Bb7,A7,Dm7,G7,CΔ7	21
CΔ7,Am7,D7,Dm7,G7,CΔ7	20
CΔ7,Fm7,Bb7,CΔ7	20

is uniquely determined by the state that the chord is used to transition to. When we perform reharmonization operations such as tritone substitutions or diminished chords, we wind up generating transitions like `bIIx` and `vii` that are substitutes for a "Dominant 5" chord (and thus serve the same function), but should be given a more accurate name.

State Precedence Currently, every possible analysis of a sequence of chords is considered to be equally likely. However, in practice, we can wind up with analyses that are worse than others. For example, a sequence of `IVm-bVIIx-I` could be parsed as Dominant 4, Tonic 3 (tritone subbed), Tonic 1. This is rather nonsensical compared to the more accurate Subdominant 4, Dominant b7, Tonic 1. The case of a Dominant 4 is more of an exception than a rule, so there should be a way to indicate some sort of precedence rules for different transition / state combinations.

Major vs. Minor Our model does not distinguish between major and minor keys. Because jazz is so full of modal mixture, this is not outrageous. It is not uncommon for jazz standards to use both major and minor versions of the tonic (e.g. "I Love Paris"). However, because our model has no distinction when it comes to training, it is perhaps too accommodating of modal mixture.

4.2 Interacting with the Model

There are many different ways people could want to interact with both the model and the corpus. Over the course of this semester, we have written functions to do things like producing analyses of chord sequences, generating chord sequences, and determining probabilities of particular chords and functions. We have also explored what information we can extract from the corpus independently of the model, such as what songs a particular chord sequence appears in and n-gram probabilities of chords. The continuation of this project next semester will likely focus on deciding the most useful ways to interact with the model and corpus and presenting this functionality in an accessible interface.

4.3 Extending the Model

Chord Duration Currently, the model does not consider chord durations, although in practice they are extremely important. The goal for this project has been to focus exclusively on progression among chords. A human can then look at generated chord sequences and make decisions about duration. However, it would be useful if the model could produce durations as well.

Song Form A logical next step after durations is to consider generating full songs. Because jazz standards tend to use set song forms, it would be useful to have the model be able to generate songs in particular forms. The model could either be given a song form and generate sections accordingly, or the form could be determined stochastically based on training data.

Modulation The model currently assumes that every song stays in the same key the entire time. Although our model allows for non-diatonic chords, there are many instances in jazz where a particular section can only be reasonably interpreted in a different key. Though what constitutes a full modulation is subject to interpretation, it would be useful if the model could handle these cases.

References

[1] Mark Levine. *The Jazz Theory Book*. Sher Music, Petaluma, CA, 1995.

[2] Dariusz Terefenko. *Jazz Theory: From Basic to Advanced Study*. Routledge, New York, NY, 2014.