Applying N-Gram Models to Musical Notes

Introduction

Much like writing, music contains uncertainty. Music can have severe ambiguity in context of notes (a first note of the key C is not the first note of any other keys. And depending on key, the note C# might be rewritten as Db.) This creates ambiguity in what makes a pleasant chord progression, how to evaluate musical phrases and whether a phrase is likely, or of a similar genre, or by a specific composer.

For my project, I parsed a corpus of classical music from midi files into an encoded format called MusicStrings (discussed in the section Prior Research.) I then wrote a Java program to parse these files into a map of notes that occurred together at different timestamps for each file, and the next set of notes to follow (discussed in the section System Design.) This architecture allowed me to create sample data (both plausible and implausible) to test whether a bigram or trigram back-off model could be used to distinguish plausible sample sequences from random ones.

Prior Research

Most of the research I found evaluating music with NLP methods, were interested in probabilistic grammars (Abdallah & Gold, 2014) (Bod). These were using highly annotated data to parse music based on phrases (Bod), which allowed a tree structure for musical sequences. This mirrors the NLP concept of NP, VP constituent parse trees.

Unfortunately, while their results were in the 80-85% range for predicting notes (Bod), I did not have access to annotated data as a gold standard.

System Design

I evaluated the sequence of notes using the Katz Trigram – Backoff model (Seymore & Rosenfeld, 1996). The probabilities of a sequence of notes are calculated with trigrams, unless a count for a particular trigram does not exist. Then the system "backs off" to a weighted value of the bigram probability. (This weighted number is calculated based on the probability mass that remains from the trigram model.)

$$P_n(w_n|w_1^{n-1}) = \begin{cases} (1-d)C(w_1^n) \,/\, C(w_1^{n-1}) & \textbf{if } C(w_1^n) > 0 \\ \alpha(C(w_1^{n-1})) \cdot P_{n-1}(w_n|w_2^{n-1}) & \textbf{if } C(w_1^n) = 0 \end{cases}$$

Input/Output

Training Data

The training data for this project consisted of a selection of midi files sequenced by Barnd Krueger (Krueger, 2015) of 25 classical composters whose works are in the public domain.[1]

The midi files were then converted to the MusicStrings encoding system from JFugue 4.0.3 (Koelle, 2015). The files were saved as text files, where they could be interpreted by the program.

Figure 1: List of Composers

14	**Albaniz**
3	**Bach**
1	**Balakirew**
27	**Beethoven**
7	**Borodin**
10	**Brahms**
9	**Burgmueller**
48	**Chopin**
17	**Clementi**
9	**Debussy**
2	**Godowsky**
3	**Granados**
16	**Grieg**
21	**Hayden**
16	**Liszt**
11	**Mendelssohn**
1	**Moszkowski**
21	**Mozart**
8	**Mussorgsky**
10	**Rachmaninov**
5	**Ravel**
29	**Schubert**
24	**Schumann**
1	**Sinding**
12	**Tchaikovsky**
325	**Total**

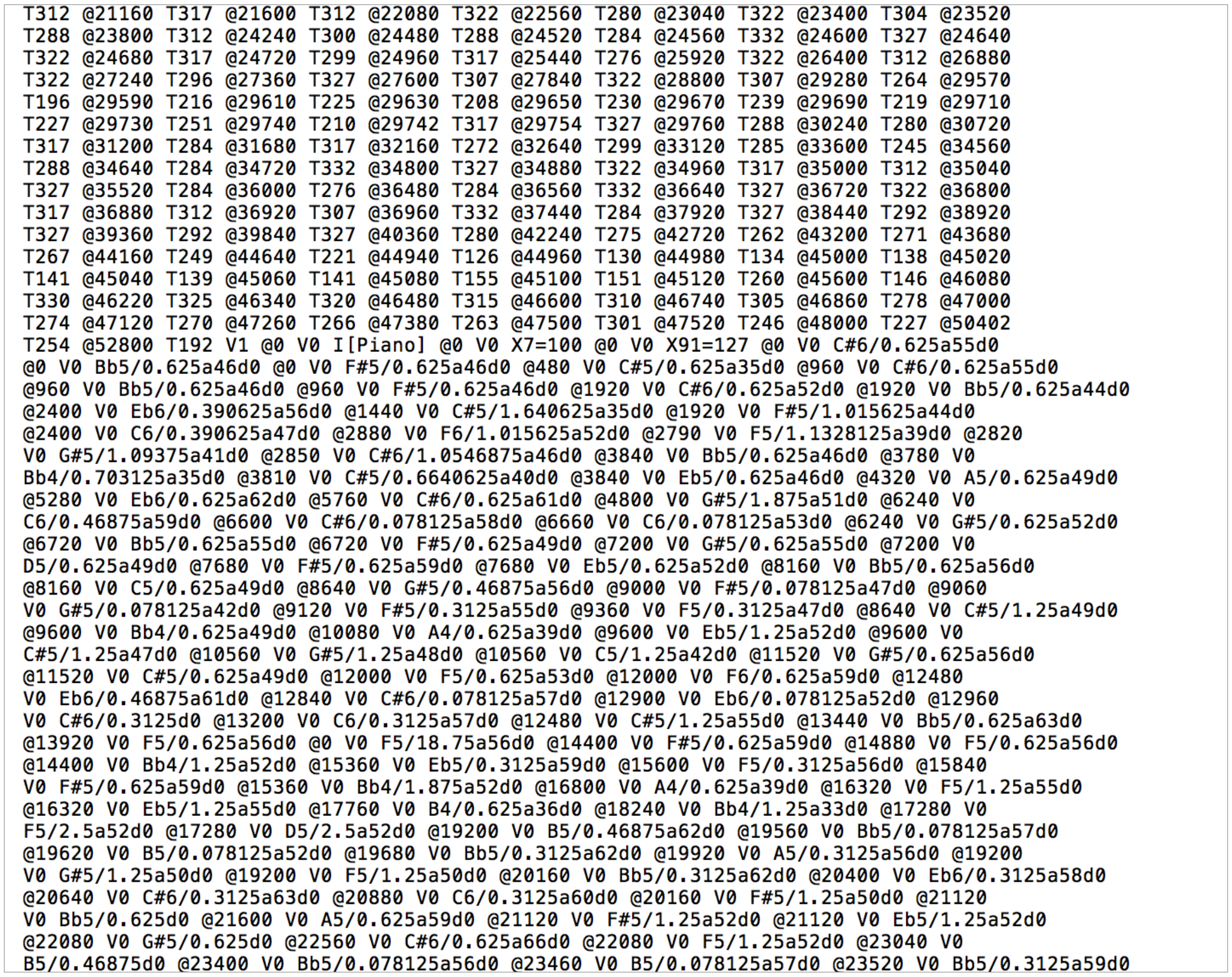
[1] At the time of this writing, Bernd Krueger, the copy write owner of the midi files placed these files under a share, adapt Creative Commons license under the conditions of giving him proper credit for the sequencing of the midis, and a share alike policy. Please see http://creativecommons.org/licenses/by-sa/3.0/de/deed.en for exact details.

A single note in a file of the training would be in the format:

F6/1.015625a52d0 @2790

The “F6” refers to the F note in the 6th octave, these notes might also have a modifier for either flat or sharp notes, denoted with “b” or “#” respectively. Every note, modifier, octave pair was treated as a separate word type in my corpus.

From the / until the space denotes the duration and attack velocity of the note, which were disregarded for this project. The final sequence “@2790” is a millisecond time stamp. Because notes could occur at the same time, and come from separate encoding channels on the midi, these time stamps allowed my parser to recreate the order of the training sequence. I also used this information to lookup following notes.

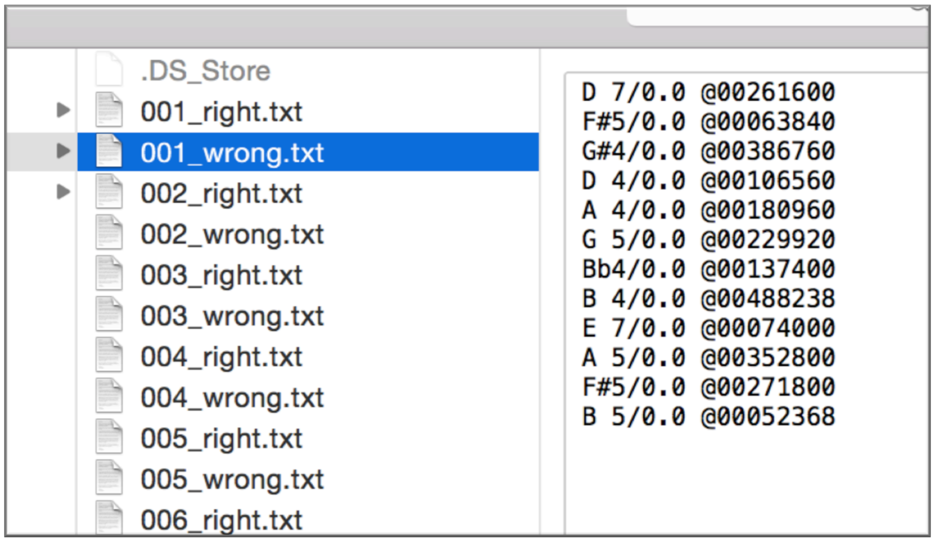


```
T312 @21160 T317 @21600 T312 @22080 T322 @22560 T280 @23040 T322 @23400 T304 @23520
T288 @23800 T312 @24240 T300 @24480 T288 @24520 T284 @24560 T332 @24600 T327 @24640
T322 @24680 T317 @24720 T299 @24960 T317 @25440 T276 @25920 T322 @26400 T312 @26880
T322 @27240 T296 @27360 T327 @27600 T307 @27840 T322 @28800 T307 @29280 T264 @29570
T196 @29590 T216 @29610 T225 @29630 T208 @29650 T230 @29670 T239 @29690 T219 @29710
T227 @29730 T251 @29740 T210 @29742 T317 @29754 T327 @29760 T288 @30240 T280 @30720
T317 @31200 T284 @31680 T317 @32160 T272 @32640 T299 @33120 T285 @33600 T245 @34560
T288 @34640 T284 @34720 T332 @34800 T327 @34880 T322 @34960 T317 @35000 T312 @35040
T327 @35520 T284 @36000 T276 @36480 T284 @36560 T332 @36640 T327 @36720 T322 @36800
T317 @36880 T312 @36920 T307 @36960 T332 @37440 T284 @37920 T327 @38440 T292 @38920
T327 @39360 T292 @39840 T327 @40360 T280 @42240 T275 @42720 T262 @43200 T271 @43680
T267 @44160 T249 @44640 T221 @44940 T126 @44960 T130 @44980 T134 @45000 T138 @45020
T141 @45040 T139 @45060 T141 @45080 T155 @45100 T151 @45120 T260 @45600 T146 @46080
T330 @46220 T325 @46340 T320 @46480 T315 @46600 T310 @46740 T305 @46860 T278 @47000
T274 @47120 T270 @47260 T266 @47380 T263 @47500 T301 @47520 T246 @48000 T227 @50402
T254 @52800 T192 V1 @0 V0 I[Piano] @0 V0 X7=100 @0 V0 X91=127 @0 V0 C#6/0.625a55d0
@0 V0 Bb5/0.625a46d0 @0 V0 F#5/0.625a46d0 @480 V0 C#5/0.625a35d0 @960 V0 C#6/0.625a55d0
@960 V0 Bb5/0.625a46d0 @960 V0 F#5/0.625a46d0 @1920 V0 C#6/0.625a52d0 @1920 V0 Bb5/0.625a44d0
@2400 V0 Eb6/0.390625a56d0 @1440 V0 C#5/1.640625a35d0 @1920 V0 F#5/1.015625a44d0
@2400 V0 C6/0.390625a47d0 @2880 V0 F6/1.015625a52d0 @2790 V0 F5/1.1328125a39d0 @2820
V0 G#5/1.09375a41d0 @2850 V0 C#6/1.0546875a46d0 @3840 V0 Bb5/0.625a46d0 @3780 V0
Bb4/0.703125a35d0 @3810 V0 C#5/0.6640625a40d0 @3840 V0 Eb5/0.625a46d0 @4320 V0 A5/0.625a49d0
@5280 V0 Eb6/0.625a62d0 @5760 V0 C#6/0.625a61d0 @4800 V0 G#5/1.875a51d0 @6240 V0
C6/0.46875a59d0 @6600 V0 C#6/0.078125a58d0 @6660 V0 C6/0.078125a53d0 @6240 V0 G#5/0.625a52d0
@6720 V0 Bb5/0.625a55d0 @6720 V0 F#5/0.625a49d0 @7200 V0 G#5/0.625a55d0 @7200 V0
D5/0.625a49d0 @7680 V0 F#5/0.625a59d0 @7680 V0 Eb5/0.625a52d0 @8160 V0 Bb5/0.625a56d0
@8160 V0 C5/0.625a49d0 @8640 V0 G#5/0.46875a56d0 @9000 V0 F#5/0.078125a47d0 @9060
V0 G#5/0.078125a42d0 @9120 V0 F#5/0.3125a55d0 @9360 V0 F5/0.3125a47d0 @8640 V0 C#5/1.25a49d0
@9600 V0 Bb4/0.625a49d0 @10080 V0 A4/0.625a39d0 @9600 V0 Eb5/1.25a52d0 @9600 V0
C#5/1.25a47d0 @10560 V0 G#5/1.25a48d0 @10560 V0 C5/1.25a42d0 @11520 V0 G#5/0.625a56d0
@11520 V0 C#5/0.625a49d0 @12000 V0 F5/0.625a53d0 @12000 V0 F6/0.625a59d0 @12480
V0 Eb6/0.46875a61d0 @12840 V0 C#6/0.078125a57d0 @12900 V0 Eb6/0.078125a52d0 @12960
V0 C#6/0.3125d0 @13200 V0 C6/0.3125a57d0 @12480 V0 C#5/1.25a55d0 @13440 V0 Bb5/0.625a63d0
@13920 V0 F5/0.625a56d0 @0 V0 F5/18.75a56d0 @14400 V0 F#5/0.625a59d0 @14880 V0 F5/0.625a56d0
@14400 V0 Bb4/1.25a52d0 @15360 V0 Eb5/0.3125a59d0 @15600 V0 F5/0.3125a56d0 @15840
V0 F#5/0.625a59d0 @15360 V0 Bb4/1.875a52d0 @16800 V0 A4/0.625a39d0 @16320 V0 F5/1.25a55d0
@16320 V0 Eb5/1.25a55d0 @17760 V0 B4/0.625a36d0 @18240 V0 Bb4/1.25a33d0 @17280 V0
F5/2.5a52d0 @17280 V0 D5/2.5a52d0 @19200 V0 B5/0.46875a62d0 @19560 V0 Bb5/0.078125a57d0
@19620 V0 B5/0.078125a52d0 @19680 V0 Bb5/0.3125a62d0 @19920 V0 A5/0.3125a56d0 @19200
V0 G#5/1.25a50d0 @19200 V0 F5/1.25a50d0 @20160 V0 Bb5/0.3125a62d0 @20400 V0 Eb6/0.3125a58d0
@20640 V0 C#6/0.3125a63d0 @20880 V0 C6/0.3125a60d0 @20160 V0 F#5/1.25a50d0 @21120
V0 Bb5/0.625d0 @21600 V0 A5/0.625a59d0 @21120 V0 F#5/1.25a52d0 @21120 V0 Eb5/1.25a52d0
@22080 V0 G#5/0.625d0 @22560 V0 C#6/0.625a66d0 @22080 V0 F5/1.25a52d0 @23040 V0
B5/0.46875d0 @23400 V0 Bb5/0.078125a56d0 @23460 V0 B5/0.078125a57d0 @23520 V0 Bb5/0.3125a59d0
```

Figure 1: A selection of the training data in MusicString notation

Test Data

There were 100 test cases in the test folder of the corpus. Each file contained 12 note patterns in the format /[A-G][b|#|]\d/, then a duration marker, which was added for the parser but contained added information, and a millisecond timestamp in the format /@\d+/. These test cases were created in pairs. I create higher probability sequences by slicing linear selections from the original corpus. These were labeled "right." I used a random number generator to take a selection of random notes to produce the lower probability sequences, which were stored in the file suffix "wrong."



```
D 7/0.0 @00261600
F#5/0.0 @00063840
G#4/0.0 @00386760
D 4/0.0 @00106560
A 4/0.0 @00180960
G 5/0.0 @00229920
Bb4/0.0 @00137400
B 4/0.0 @00488238
E 7/0.0 @00074000
A 5/0.0 @00352800
F#5/0.0 @00271800
B 5/0.0 @00052368
```

Figure 2: A screenshot of the test data and file naming

Results

Here's a sample of output from two files.

Bigram results of file 050_right.txt and 050_wrong.txt

```
P(F 4|C#3) is: 0.04730105731775181
P(C 4|F 4) is: 0.04119804400977995
P(C#4|C 4) is: 0.031867757327880025
P(Bb5|C#4) is: 0.048867077372872506
P(F#5|Bb5) is: 0.05036983019064486
P(C#4|F#5) is: 0.03132042762207454
P(C#6|C#4) is: 0.08301138143468727
P(G#5|C#6) is: 0.06684235745088644
P(Bb5|G#5) is: 0.06301669860388721
P(F 5|Bb5) is: 0.08042504427544536
P(Eb5|F 5) is: 0.06951138624806544
P(D 6|C 4) is: 0.037917518745739606
```

```
P(B 5|D 6) is: 0.07672689547851647
P(D 7|B 5) is: 0.011922583404619333
P(F#5|D 7) is: 0.028049748610743584
P(G#5|F#5) is: 0.05928922276798613
P(B 6|G#5) is: 0.016041609635915687
P(F#6|B 6) is: 0.05639662544782156
P(C 7|F#6) is: 0.02114181645419273
P(F 4|C 7) is: 0.03493950771798081
P(C#5|F 4) is: 0.045965770171149146
P(E 5|C#5) is: 0.0993884665942733

The bigram chain probability of the first notes is 4.39176988763834E-17
The bigram chain probability of the second notes is 2.6987902121323685E-18
The first notes are more likely.
```

Trigram results of file 050_right.txt and 050_wrong.txt

```
P(C 4|F 4) is: 0.04119804400977995
P(C#4|C 4) is: 0.031867757327880025
P(Bb5|C#4) is: 0.048867077372872506
P(F#5|Bb5,C#4) is: 0.00641025641025641
P(C#4|F#5,Bb5) is: 0.004136504653567736
P(C#6|C#4) is: 0.08301138143468727
P(G#5|C#6,C#4) is: 0.01509433962264151
P(Bb5|G#5,C#6) is: 0.008064516129032258
P(F 5|Bb5,G#5) is: 0.0026064291920069507
P(Eb5|F 5,Bb5) is: 0.0038860103626943004

P(B 5|D 6,C 4) is: 0.029213483146067417
P(D 7|B 5,D 6) is: 5.633802816901409E-4
P(F#5|D 7) is: 0.028049748610743584
P(G#5|F#5) is: 0.05928922276798613
P(B 6|G#5) is: 0.016041609635915687
P(F#6|B 6,G#5) is: 0.006825938566552901
P(C 7|F#6,B 6) is: 0.0020491803278688526
P(F 4|C 7) is: 0.03493950771798081
P(C#5|F 4) is: 0.045965770171149146
P(E 5|C#5,F 4) is: 0.003989361702127659

The trigram chain probability of the first notes is 6.999492477295349E-22
The trigram chain probability of the second notes is 5.146076140377186E-22
The first notes are more likely.
```

Final output

```
--------------------------------------------------
Precision for Bigram Back-off model: 0.78
--------------------------------------------------
--------------------------------------------------
Precision for Trigram Back-off model: 0.42
--------------------------------------------------
The bigram model is more precise.
```

Conclusion

The trigram method was less effective than the bigram model for predicting whether a sequence of notes was from the corpus or random.

Other methods might improve the work, suck as differentiating between chords (notes that occur together) rather than evaluate sequences at the note level. Because this would create a larger vocabulary, I would also have to implement smoothing methods in order to create the bigrams and trigrams.

The lack of an annotated corpus was a huge limiting factor, because I was not able to identify or parse musical phrases with any accuracy.

References

Abdallah, S. A., & Gold, N. E. (2014, September). Comparing models of symbolic music using probabilistic grammars and probabilistic programming. (G. K. A. Georgaki, Ed.) *Proceedings ICMC|SMC* .
Bod, R. *Probabilistic Grammars for Music.* University of Amsterdam.
Koelle, D. (2015). Retrieved March 2015, from JFugue - The Java API for Music Programming: http://www.jfugue.org/4/
Krueger, B. (2015, March). *Classical Piano Midi Page*. Retrieved from http://www.piano-midi.de/borodin.htm