Probabilistic modeling of musical chord sequences for music analysis

Metiss Team, IRISA

by Christophe Hauser
supervised by Emmanuel Vincent and Frédéric Bimbot

June 4, 2009
Abstract

The aim of this work is to present a model for representing chord sequences using n-grams, an approach from the language recognition field.

Previous works show that there exist various means to represent chords, with more or less accuracy. Underlined limitations in [14] are the modeling of rare chords and the representation of chords using an appropriate alphabet. Recent works in the METISS team [14] proposed different labelling schemes and smoothing methods to overcome these limitations. It has been shown that smoothing provides better generalization of the models, yielding better results in unmet situations. In this work we compare two methods from the language recognition field, in different situations using two different datasets and seven different chord notations. Finally, we present two possible applications of n-gram modeling regarding music classification.
Aknowledgements

I would like to address special thanks to my supervisors, Emmanuel Vincent and Frédéric Bimbot for their support and advices, for their great patience in improving my writings and for their guidance all along this period.

I am grateful to all the people in the Metiss Team for their valuable advices and sympathy.

Finally, I would like to thank to teaching staff for having provided us the theoretical background necessary to accomplish this work.
## Contents

References ........................................... 3

1 Introduction ....................................... 5

2 Background ......................................... 7
   2.1 Music databases and information representation ............ 7
   2.2 Possible applications .................................. 8
   2.3 Monophonic and polyphonic data .......................... 8
   2.4 Chords and MIR ....................................... 8

3 Modeling Chord Sequences ......................... 10
   3.1 About chords ......................................... 10
      3.1.1 Notes and scales ................................. 10
      3.1.2 Making up chords ............................... 11
      3.1.3 Tonality ....................................... 12
   3.2 Chord notation .................................... 12
      3.2.1 Baroque Figured bass ......................... 12
      3.2.2 Classical Harmony Analysis .................... 12
      3.2.3 Jazz and popular music ....................... 12
   3.3 Existing models .................................... 13
   3.4 N-Gram modelling .................................. 14
      3.4.1 Likelihood .................................... 14
      3.4.2 Perplexity & cross-entropy .................... 14
   3.5 Critical Issues .................................... 15
      3.5.1 Chord notation ................................ 15
      3.5.2 Modeling of rare chords ....................... 16

4 Previous works .................................... 17
   4.1 Labelling schemes .................................. 17
      4.1.1 Simple notation ................................ 17
      4.1.2 Shorthands notation ........................... 17
      4.1.3 Reduced notation ................................ 18
      4.1.4 Tonality independent transformation .......... 18
   4.2 Smoothing techniques .............................. 18
List of Figures

3.1 Intervals in music theory ........................................ 11
3.2 Baroque figured bass .............................................. 12
3.3 Classical harmony analysis ........................................ 12
3.4 Jazz and popular music ............................................ 13

7.1 Jelinek-Mercer - the Beatles ..................................... 26
7.2 Jelinek-Mercer - Jazz authors .................................... 27
7.3 All methods - The Beatles ......................................... 28
7.4 All methods - various jazz authors ............................... 29

8.1 A comparison of various jazz authors ............................. 31
8.2 Similarity graph ..................................................... 32
Chapter 1

Introduction

Computer and network technologies have improved consequently over the last years. Technology brought the ability for everyone to get involved in music, to listen, to compose, to record. With more and more online services, the amount of data to be stored and delivered is getting very large. Managing all these collections of data involves new aspects such as efficient storage methods and advanced information retrieval techniques. Music information retrieval is one example of technologies aimed to identify musical data within large music collections such as iTunes.

One key element in occidental music is the concept of chord. Identifying chords in a song gives a lot of information about it, and simplifies further analyses, such as determining the melody of a given instrument, or the bass line.

The aim of this work is to present a model for representing chord sequences using n-grams, an approach from the language recognition field. While other studies may consider rhythm and timber characteristics, our approach is exclusively focused on the chord sequences, reflecting the underlying musicological features.

Previous works show that there exist various means to represent chords, with more or less accuracy. Underlined limitations in [14] are the modeling of rare chords and the representation of chords using an appropriate alphabet. Recent works in the METISS team [14] proposed different labelling schemes and smoothing methods to overcome these limitations. It has been shown that smoothing provides better generalization of the models, yielding better results in unmet situations.

Many smoothing methods exist from the language recognition field and are likely to provide better performances in the context of music modeling. In this work, we focus on the performances of three different smoothing methods and attempt to measure the impact of having different labelling schemes and datasets over their generalization capabilities.

Two possible applications of our models are then presented regarding
music classification. The first one is a comparison of authors, the other one is a similarity network over a musical corpus.
Chapter 2

Background

Music information retrieval (MIR) is the interdisciplinary science of retrieving information from music. There are two main research branches in this field: the transcription of audio signal to symbols and the modeling of these symbols. In [7], the authors speak about Acoustic Modeling and Language modeling.

2.1 Music databases and information representation

Managing and sorting large collections of music files involves these two aspects, each at a different layer.

Music databases contain files in various formats, with different levels of representation. Thus we need a common ground in order to compare different files together. For example, MIDI gives information about rhythm and pitch while raw audio only encodes the sound energy level over the time. This is achieved at low level where audio signal is analyzed and then represented using an alphabet of symbols, thus providing a mid-level representation of the input signal. [5, 6]. Various techniques are proposed, but this is outside the scope of this study.

Once a mid-level representation of data is reached, models and algorithms are needed to manage, sort or compare data given different criteria. By contrast with today’s databases, where human typed string description of content is used as meta-data, new approaches use information extracted from the music itself. This is done over a mid-level representation of data using an alphabet of symbols. [14, 5, 4, 11]. A number of existing models have been used in the past and will be presented later in this study.
2.2 Possible applications

Applications of IR models are promising in fields such as musicology analysis, computer assisted composition and browsing large music databases. A number of previous works exist. New approaches for browsing music databases include query by humming and query by example. With query by humming as presented in [5], the user hums in a microphone. Pitch and rhythm information are extracted from the resulting signal, converted into symbolic data and modeled to create meta-data. Query by example as shown in [5] [6] is similar except that it uses sound files extracts as queries. Some other works focused on artist identification with different approaches. In [18], artists are automatically identified from the acoustic signal using acoustic features modeling. This approach is based on the analysis of the spectral characteristics of the audio signal. A different approach is presented in [11], where chords sequences are analysed in order to make chord profiles representative of the composer style. This last approach uses higher level features and thus is closer to our goal. Melody harmonizing [15] is another example of possible applications, where a melody is synthesized given a chord sequence. This can be used for automatic composition, or to verify the consistency of a melody over the chords of a song.

2.3 Monophonic and polyphonic data

Transcription of an audio signal into sequences of symbols can be achieved using different models and algorithms [5, 6, 7]. However, the accuracy of the transcription highly depends on the nature of the audio signal (monophonic or polyphonic), especially in the case of polyphonic data, where it remains a challenge. For this reason, and because our study is focused on the modeling of symbols, we won’t use any automatic transcription systems but rather use manually transcribed corpora of songs as training data.

Polyphonic data can be modeled [6] using a short-term frequency analysis method, where a pitch vector is constructed (called pitch class profile and containing information about pitch and energy of each note). Even though this method doesn’t provide melody-level accuracy, it is enough to extract information about chords and provide a mid-level representation of data.

2.4 Chords and MIR

Chords are interesting regarding MIR, especially when it comes to analyse polyphonic data. Chords are central to most modern music. They represent musical attributes and contain rich information for music analysis. A Chord progression, also known as harmonic progression is series of chords played from one to another. Perception of music is tightly coupled with it. Chords
2.4 Chords and MIR

enforce establishment of tonalities [1], and are the background for melody. According to [11], it can also be incorporated into a tune retrieval system where tunes are indexed with meta-data and chord sequence.
Chapter 3

Modeling Chord Sequences

When modeling music, many parameters can be taken into account. The same song played with a banjo or a guitar won’t sound the same. Sound texture does matter to human ears. Rhythm is very important too, and the same song played at different tempo changes the way it sounds. All these parameters can be used to identify characteristics for music classification. Our approach focuses on the chord sequences themselves. Thus we discard any timbral aspect, and keep minimalistic information about rhythm: how many times each chord is played in terms of bars (often four beats in length). This limitation to chord sequences is voluntary: our goal is to be able to predict musical variations at the chord level, while keeping the model simple enough to effectively measure their accuracy in different situations. Thus, the comparison of songs in this manner does not directly reflect the difference one may perceive with human ears, but is rather a structural comparison based on the underlying musicological features.

3.1 About chords

Note: the following examples are using the jazz notation, as described in chapter 3.3.

3.1.1 Notes and scales

The perceived frequency of a note is called “pitch”.

In traditional western music, a scale consists in a group of seven notes and repeats at the “octave”, obtained by doubling the pitch of its first note. A note and its octave have the same label or name. The gap between two notes in a scale is quantified by a variable number of a minimal step called “semitone” (generally one or two semitones). The distance between two
3.1 About chords

notes is called interval, and is named after the number of notes from one to another within a scale. For example, the interval C-A is a sixth because \((C,D,E,F,G,A) = 6\). Because the steps in a scale are not uniform but defined by a variable number of semitones, (generally one or two), intervals exist in different manners. For example, an interval of third represents a step of three or four semitones depending if it is a so called “major” or “minor” third. This property is called quality. Fig 3.1 shows the different intervals and their qualities.

3.1.2 Making up chords

A chord is a set of at least three different notes played simultaneously (though some theorists claim that it can be called chord from two notes). A chord composed of three notes is called a triad. The first note is called the root and corresponds to the fundamental frequency of the chord (thus it is sometimes called fundamental instead of root). The two following notes are named after the interval they form with the root. Therefore, in the case of a triad, the second note is called the third, and the last note is called the fifth.

As for intervals, a chord has a quality which depends on the quality of the intervals of its composing notes relatively to the root. Ex: Cm is a chord of C minor and is composed of C (root), \(E_b\) (minor third) and G (fifth) C. Cdim is composed of C, E (minor third), \(G_b\) (diminished fifth).

Other notes can be added on top of triads, such as the 7th, the 9th, the 11th, respectively forming an interval of 7th, 9th and 11th with the root of the chord.

Chords are generally played with the root note being the bass note (ie the lowest one in pitch), but sometimes another note is used instead. This is called an inversion.

Here are some examples of chords. Am7 = A C G ; A is the root, C is the minor third and G is the seventh. Bm7b5 = B D F A ; B is the root, D is the minor third, F is the flat fifth and A is the seventh.
3.1.3 Tonality

A tonality, is an ensemble of notes in which relationships are centered on a special note called the tonic. For example, the tonality of C is the ensemble containing all the notes without any sharp or flat (A B C D E F G). Within a tonality, a certain number of chords and scales exist, and some others don’t. Thus, given a sequence of chords, it is possible to determine their tonality.

3.2 Chord notation

In music, chords can be represented in many different ways. According to Harte [2], three are commonly used.

3.2.1 Baroque Figured bass

(Fig. 3.2) Using these notations, figures show which notes can be played above the bass line to complete the correct harmony.

![Figure 3.2: Baroque figured bass](image)

3.2.2 Classical Harmony Analysis

As shown on fig 3.3, the chords are noted in the context of a given key (the key defines a tonality, it is equivalent to the tonic). Notes are not explicitly described. For example, when a seventh chord is annotated, the seventh can actually be major or minor depending on the context. Inversions are marked as well.

![Figure 3.3: Classical harmony analysis](image)

3.2.3 Jazz and popular music

(Fig 3.4) In jazz, musicians often play at sight and need an explicit manner of representing chords. The quality of each chord is explicitly marked.
3.3 Existing models

This is the most commonly used notation today because it provides more information and it is context independent. Harte’s notation is a formalized form of the jazz notation and has been used in many previous works. This is the one we will use as a starting point for our study, along with simplified derived notations in [14].

![Chord Sequence](image)

Figure 3.4: Jazz and popular music

3.3 Existing models

Sequences of chord labels can be seen as word sequences in the natural language, where grammar rules would be the rules of harmony [7]. Training as in language modeling is then a reasonable solution, and the same models are likely to apply to model chord sequences. Many machine learning models exist. Commonly used models include

- **Hidden Markov Model (HMM)**, as proposed in [12] is a model where the system being modeled is assumed to be a Markov process with unknown parameters. In a Markov process, the likelihood of a given future state depends only on the present state, and not on any of the past states. Therefore it is said to be memoryless. This is actually a particular case of the n-gram model with n=2. It would only consider the transition probability between two consecutive chords.

- **Markov random field or Markov network** is a graphical model in which a set of random variables have a Markov property described by an undirected graph. According to [13], this is a model of joint probability distribution of a set of random variables having the Markov property. This model is a generalization of Markov chains to multidimensional spatial processes.

- **Conditional random fields (CRF)** is used in language modeling. It can be seen as a generalization of the Markov random field model. According to [8] it has advantages over HMM, and is related to it in that it include the ability to relax strong independance assumptions.

---

1 The Markov property defines a stochastic process where the conditional probability distribution of future states of the process depends only on the present state and on a defined set of past states.
3.4 N-Gram modelling

N-gram modeling is a good compromise between simplicity and effectiveness. It is less complex than random fields, and it is adapted to chord sequences modeling as it would consider the transitions from a set of chords to another chord within a sequence, as explained in [7]. A number of previous works used n-gram modeling for different applications, such as polyphonic music retrieval [6, 5] composer style representation [10, 11], chords recognition from audio [12], and harmonisation [15]. Deterministic approaches such as in [5] offer efficient approach in term of computation cost but cannot model the likelihood of chord sequences as needed in our study. Therefore, probabilistic models as in [14] and [10] will be used in our study.

In the following, $x_1...x_n$ represents a sequence of $n$ chords.

N-gram models are a type of probabilistic model for predicting the next item in a sequence. n-grams are used in various areas of statistical natural language processing and genetic sequence analysis. An N-gram is a sub-sequence of $n$ items from a given sequence.

3.4.1 Likelihood

The likelihood of a sequence $s$ composed of $n$ chords is noted

$$p(s) = p(x_1, ..., x_i)$$  \hspace{1cm} (3.1)

In term of probability, the likelihood of an n-gram is

$$p(x_i | x_{i-1}, ..., x_{i-n+1})$$  \hspace{1cm} (3.2)

and therefore, the likelihood of a chord sequence $s$ is

$$p(s) = \prod_{i=1}^{s+1} p(x_i | x_{i-n+1}^{i-1})$$  \hspace{1cm} (3.3)

with $|s|$ the length of the sequence.

As in [7], this model is consistent with the idea of chords progression because the likelihood of a chord depends on its $n$ ancestors. Rather than only considering very short-term dependencies, n-gram modeling does consider the history of a given chord. Short / middle term dependencies are then reflected in the model. Furthermore, the number of symbol used to represent chords is rather small compared to what is used when processing natural language. This makes the n-gram model an effective approach.

3.4.2 Perplexity & cross-entropy

When it comes to evaluate the quality of a model and its generalization to unknown situations, or to compare it with other models, the measures
of “cross-entropy” and “perplexity” are often used. Empirically, this reflects how “perplex” is the model when faced with a new situation (new in the sense that it hasn’t been seen during the training stage). The level of perplexity gives information about how well the model was able to predict words in a sentence, according to their probability.

**Cross-entropy**

Performance of language models is often estimated using cross-entropy. According to [3], it has been shown that the performance of models is strongly correlated with their cross entropy on test data. The cross-entropy is used in [14] to compare the likelihood of chord sequences of different lengths, as it is a normalized measure.

The cross-entropy \( H_p(x) \) of a model \( p(x_i|x_{i-n+1}) \) on a chord sequence \( s \) is defined as

\[
H_p(x) = - \frac{1}{|s|} \log_2 p(x) \tag{3.4}
\]

where \(|s|\) is the length of the considered chord sequence. It is expressed in bits/symbols.

As stated by the relation between prediction and compression [3], given a language model that assigns a probability \( p(s) \) to a sequence \( s \), we can derive a compression algorithm that encodes the sequence \( s \) using \(-log_2 p(s)\) bits. This value can be interpreted as the average number of bits needed to encode each of the chords \( x_i \) in the sequence \(|s|\) using the compression algorithm associated with the model \( p(x_i|x_{i-n+1}) \)

**Perplexity**

The perplexity is defined as

\[
2^{H_p(x)} \tag{3.5}
\]

## 3.5 Critical Issues

### 3.5.1 Chord notation

Harte’s notation is aimed to be simple and intuitive to musically trained individuals to write and understand. Using this notation, every possible chord can be described. The total number of possibilities is huge, and has an impact on models performances.

According to a previous analysis in [14], considering different chord dictionaries can sensibly affect the results and the computation cost of the modeling: *perplexity* and *cross-entropy* are influenced by the number of different symbols as well as their meaningfulness.
3.5 Critical Issues

3.5.2 Modeling of rare chords

Machine learning algorithms are trained using some sets of examples. The aim is to become able to predict the right output with other examples that were not present in the learning set and generalize to any set of data. When the model is trained, it is expected to reach a certain level of generalization, i.e., to reach a state where it will be able to give consistent results for other examples. In order to get best generalization, the chosen training set should be as large and representative as possible. One common limitation is rare elements.

Elements non observed in the training set will be assigned a zero probability, even though there might be some probability they occur. Then any sequence composed by such an element will be estimated as nonexistent in the model, no matter if they exist or not. Furthermore, the presence of so-called 'zeroprobs' leads to infinite perplexity measure, which makes it difficult to compare models together.

To overcome this problem, smoothing techniques can be used. Also called discounting, these methods generally “smooth” the model by discounting the probabilities of seen words and assigning the extra probability mass to unseen words.

Chord sequences are no exception to this rule, and as words, the existence of some rare chords leads to consider the application of smoothing techniques to the n-gram model. [14]
Chapter 4

Previous works

As mentioned in previous chapter, the two main issues when modeling chord sequences are the considered set of symbols and the presence of rare chords. In order to address these problems, previous works proposed different chord labelling schemes and smoothing methods.

4.1 Labelling schemes

While most previous studies used a limited set of symbols, only considering M/m and diminished/augmented chords, the Metiss team used a more realistic set of symbols with the representation proposed by Harte [2]. Furthermore, they tried to simplify the notation to limit the number of symbols to a subset of what Harte’s notation permits.

Three labelling schemes derived from Harte’s notation were proposed by [14], reducing the number of symbols.

4.1.1 Simple notation

This notation is similar to Harte’s notation except that enharmonic \(^1\) roots are considered as equivalent (A#/Bb are considered as the same note), resulting in a set of 12 possible roots. In theory, the number of possible symbols is still about several millions, but only 392 symbols were observed on the Beatles set introduced later in this paper.

4.1.2 Shorthands notation

This intermediate scheme consists of representing a chord by its root and type chosen among 12 non-enharmonic roots and the 17 shorthands types

\(^1\)Enharmonic notes are notes with the same pitch, but using different labels. For example, A\(_b\) and B\(_b\) are enharmonic.
defined by Harte. Additionally, it discards dissonances and bass information. There are 204 possible symbols with this notation.

4.1.3 Reduced notation

As used in most of the current approaches, this scheme assumes a set of 24 chords considering only major and minor chords. Augmented and diminished are transformed into major and minor chords respectively. Dissonance and bass information is discarded.

4.1.4 Tonality independent transformation

[14] also proposed a transformation function which derives a tonality independent label from a tonality and a label. The root is then transformed into one of the 12 independent roots representing its relation to the initial tonality. For example, a Dm7 chord in tonality of C would become IIm7. This allows better generalization by estimating equivalent chords in different tonalities. This transformation can be applied to the previous three notations resulting in six different set of symbols for representing chords.

4.2 Smoothing techniques

There are many existing smoothing methods, especially from the language modelling world. Several smoothing techniques have been used in previous works. [6] uses a Universal Background Model. But this approach is impractical because according to [14] the UBM may itself suffer from overfitting. Additional smoothing techniques have been proposed in [14, 9]

4.2.1 Additive smoothing

Additive smoothing, or Laplace smoothing consists in adding an arbitrary number $\delta$ to the counts of every chords sequences (where $D$ is the considered dictionary of chord symbols and $|D|$ the number of symbols). In other words, we assume it occurs $\delta$ times more than it actually does.

$$p_{add}(x_i|x_{i-n+1} \cdots x_{i-1}) = \frac{\delta_n + c(x_{i-n+1}, \cdots, x_i)}{\delta_n |D| + \sum_{x_i} c(x_{i-n+1}, \cdots, x_i)}$$ (4.1)
4.2 Smoothing techniques

4.2.2 Jelinek-Mercer smoothing

Jelinek-Mercer smoothing involves a linear interpolation of the maximum likelihood model with the collection model, using a coefficient $\lambda$ to control the influence of each model. For example, let us consider a test set containing the bi-gram “A B”, which was not seen in the training set, and a training set containing the uni-gram “A”. In this case, $p_{ML}(B|A) = 0$ and $p_{ML}(A) > 0$. Using Jelinek-Mercer, we obtain $p_{JM}(B|A) = \lambda p_{ML}(B|A)+(1-\lambda)p_{ML}(A) > 0$ The idea behind Jelinek-Mercer is to recursively interpolate the probability of unseen $n$-grams with seen lower-order $n$-grams. In other words, the nth-order smooth model is defined recursively as a linear interpolation between the nth-order maximum likelihood model and the (n-1)th-order smoothed model.

Here is the general form of the equation.

$$p_{JM}(x_{i}|x_{i-n+1} \cdots x_{i-1}) = \lambda p_{ML}(x_{i}|x_{i-n+1} \cdots x_{i-1}) + (1-\lambda)p_{JM}(x_{i}|x_{i-n+2} \cdots x_{i-1})$$

(4.2)

It has been shown in [14] that Jelinek-Mercer smoothing performs significantly better than additive smoothing. Also, [3] reports poor results when using additive smoothing in the context of spoken language recognition.
Chapter 5

Our contributions

As mentioned before, there is a number of parameters that have an impact on the performances of the models. Critical issues are the presence of rare chords in the training data, as well as the number of symbols used to represent chords and their meaningfulness. Smoothing methods have been shown to provide better generalization in previous works. The following smoothing methods have been presented in [3] along with proposed implementation details. We focused on the use of these methods from the language recognition field in the context of music modeling.

5.1 Smoothing methods

Note: in the following, the notation $c(w^j_i)$ represents the number of counts of the n-gram composed of the words from $w_i$ to $w_j$

5.1.1 Jelinek Mercer Smoothing

In [14], Jelinek-Mercer smoothing parameters were obtained using an exhaustive approach. $\lambda$ interpolation coefficients were chosen and tested from a set of 20 logarithmically spaced values. Also, higher order models were re-using lower order model’s $\lambda$ coefficients instead of recalculating them, for computation cost convenience. As proposed in [3], Jelinek-Mercer coefficients can be determined using the Baum-Welch algorithm. Given initial values for the $\lambda$ coefficients, the Baum-Welch algorithm adjusts these parameters iteratively to minimize the entropy of some data. The algorithm generally decreases the entropy with each iteration and guarantees not to increase it.

There is several possible implementations of this algorithm. According to [9], the recursion can be ended either by interpolation with the 1st-order smoothed model, or by taking the 0-th order model to be the uniform dis-
5.1 Smoothing methods

In previous algorithms, the lower order distribution is obtained by smoothing the lower order maximum likelihood distribution. However, this lower order distribution will have a significant role only in the case where the higher order distribution has few or no counts. Kneser-Ney smoothing method is optimized to perform well in these situations. In addition to the smoothing methods used by the Metiss team in [14], we focused on two well known methods from the language recognition field, as presented in [3]. These are the original Kneser-Ney methods, and its modified version as proposed by Chen & Goodman [3]. Kneser-Ney is defined as an extension of absolute discounting where the higher order distribution is created by subtracting a fixed discount $D \leq 1$ from each nonzero counts, rather than performing a linear interpolation with the lower-order maximum likelihood estimate as used in Jelinek Mercer. Thus the lower order distribution is interpolated with all words, not only the words having zero counts in the higher-order distribution.

$$p_{abs}(w_i|w_{i-n+1}^{i-1}) = \frac{\max[c(w_{i-n+1}^i) - D, 0]}{\sum_{w_i} c(w_{i-n+1}^i)} N_{1+}(w_{i-n+1}^{i-1})$$  \hfill (5.1)

where $N_{1+}$ describes the number of unique words that follow the history $w_{i-n+1}^{i-1}$, defined as

$$N_{1+}(w_{i-n+1}^{i-1}) = |w_i : c(w_{i-n+1}^{i-1} w_i) > 0|$$  \hfill (5.2)

It is suggested to set $D$ through deleted estimation on the training data. According to [3] this gives

$$D = \frac{n_1}{n_1 + 2n_2}$$  \hfill (5.3)

where $n_1$ and $n_2$ are the total number of n-grams with exactly one and two counts in the training data, respectively.

5.1.3 Modified Kneser-Ney

Modified Kneser-Ney has been introduced by Chen & Goodman in [3] and is said to have excellent performance. Instead of using a single discount $D$ for all nonzero counts as in Kneser-Ney, three constants $D_1$, $D_2$, and $D_3$ are applied to n-grams with one, two and three or more counts respectively. According to [3], it is motivated by the fact that the ideal average discount
5.1 Smoothing methods

for n-grams with one or more counts is substantially different from the ideal average discount for n-grams with higher counts.

\[
p_{KN}(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i-1}) - D(c(w_{i-n+1}^{i-1})))}{\sum w_{i} c(w_{i-n+1}^{i-1})} + \gamma(w_{i-n+1}^{i-1})p_{KN}(w_i|w_{i-n+2}^{i-1})
\]

(5.4)

where \(D(c) = \begin{cases} 0 & \text{if } c = 0 \\ D_1 & \text{if } c = 1 \\ D_2 & \text{if } c = 2 \\ D_3 & \text{if } c \geq 3 \end{cases} \)

To make the distribution sum to 1, we take

\[
\gamma(w_{i-n+1}^{i-1}) = \frac{D_1 N_1 (w_{i-n+1}^{i-1} \bullet) + D_2 N_2 (w_{i-n+1}^{i-1} \bullet) + D_3 N_3 (w_{i-n+1}^{i-1} \bullet)}{\sum w_{i} c(w_{i-n+1}^{i-1})}
\]

(5.5)

According to [3], the following equations can be used to estimate optimal values for \(D_1, D_2, D_3\)

\[
Y = \frac{n_1}{n_1 + 2n_2}
\]

(5.6)

\[
D_1 = 1 - 2Y \frac{n_2}{n_1}
\]

(5.7)

\[
D_2 = 2 - 3Y \frac{n_3}{n_2}
\]

(5.8)

\[
D_3 = 3 - 4Y \frac{n_4}{n_3}
\]

(5.9)

where \(n_1, n_2, n_3, n_4\) are the total number of n-grams with exactly one, two, three and four counts in the training data, respectively.
Chapter 6
Implementation

Previously proposed contributions were implemented using the following tools and data.

6.1 SRILM Toolkit

SRILM [16] is an extensible language modeling toolkit from the speech technology and research laboratory (SRI) written by Andreas Stolcke. SRILM is a collection of C++ libraries, executable programs, and helper scripts designed to allow both production of and experimentation with statistical language models for speech recognition and other applications. SRILM is freely available for noncommercial purposes. The toolkit supports creation and evaluation of a variety of language model types based on N-gram statistics, as well as several related tasks, such as statistical tagging and manipulation of N-best lists and word lattices.

SRILM supports most of the smoothing methods proposed by Chen & Goodman in [3] with the same implementation. We used SRILM for our tests.

6.2 Dataset

As mentioned before, one essential element in machine learning is the data we use for training. When modeling music, the musical genre is likely to have an effect on the behavior of algorithms. For example, some chords may be rare in pop music and very common in jazz. For this reason, we used two different datasets in our experiments.

6.2.1 The Beatles

One is composed of the 13 studio albums of the Beatles labeled by Harte using his grammar [2]. It includes 14194 chords occurrences in 180 songs.
6.3 Methodology

The labelling schemes we used for this dataset are the six derived labelling schemes proposed in [14].

6.2.2 Various jazz authors

The other one is composed of 300 songs totalizing 42929 chords from ten various authors including John Coltrane, Chick Corea, Duke Ellington, Herbie Hancock, Wayne Shorter and others. This dataset is using its own file format, and thus has been converted to Harte’s notation.

6.3 Methodology

In order to efficiently measure the effect of smoothing techniques on the modeling, separate tests were performed for each labelling scheme and for each smoothing method. Every test was performed according to the so called “leave one out” method for cross validation on each of the albums in the training set. For the Beatles, each test was cross validated in 13 folds, using one album as the test set and all the others as the training set. The results for each fold were then averaged, so as to avoid the “Album effect” as mentioned in [18].

All of the six derived labelling schemes from [14] as well as Harte’s labelling scheme were independently used in our tests, along with the three previously exposed smoothing methods : Jelinek Mercer, kneser-Ney and Chen & Goodman’s modified kneser-Ney.

For Jelinek-Mercer smoothing, the $\lambda$ coefficients were trained on the training set (giving optimal $\lambda$ for each model). Also, we used only one “bucket” per model order, resulting in a $\lambda$ vector of length $N$ for an n-gram model of order $n = N$.

6.4 Code

In order to realize our experiments, the Ruby programming language was used. Ruby is a simple an elegant full object programming language with powerful regular expressions support. Graphics were drawn using Ruby in conjunction with the well known opensource R statistics package. A set of Ruby classes were realized in order to provide the necessary abstraction over the SRILM toolkit and the different datasets. Therefore, it can be easily generalized to perform n-gram modeling experiments with different datasets and smoothing methods.
Chapter 7

Results

The following is a summary of the results we obtained in our experiments, using the previously introduced implementation.

7.1 Jelinek Mercer

7.1.1 The Beatles

As shown on figure 7.1, Jelinek-Mercer smoothing using the so called “simple” labelling scheme (derived from Harte’s notation [2]) generalizes well up to order 5. Higher order models did not give a noticeable gain in performance (less than 0.01 bit/symbol for each higher order). Using the five others labelling schemes on the Beatles set gave different cross-entropy measures but similar results: there is no noticeable improvement for \( n \geq 5 \).

Vertical axis shows the level of cross entropy for each model. Horizontal axis shows the model order.

7.1.2 Jazz Authors

The same test over various jazz authors using Harte’s labelling scheme, as shown on fig.(7.2), does not show any improvement for \( n \geq 5 \) neither. However, the overall cross-entropy is lower compared to the Beatles.

Jelinek-Mercer smoothing generalization capabilities appear to be comparable with those two datasets whatever labelling scheme is used. Even though cross-entropy may vary from one labelling scheme to another, no significant improvements were observed above \( n = 5 \).

7.2 kneser-Ney and modified kneser-Ney

In the following, “ukndiscount” stands for “Unmodified kneser-Ney” while “kndiscount” stands for “kneser-Ney discounting.”
7.3 Conclusion

Figure 7.1: Jelinek-Mercer - the Beatles

7.2.1 The Beatles
Using Chen & Goodman’s modified kneser-Ney, best results were observed for \( n = 6 \) using the “simple” labelling scheme, with a slight decrease in performance for higher order levels. This has been observed as well using the others labelling schemes. Modified kneser-Ney performs significantly better than the original kneser-Ney.

7.2.2 Jazz authors
Using this dataset, both kneser-Ney and modified kneser-Ney did not generalize well with higher order models. It even gave worse results from \( n=3 \) to \( n=5 \).

7.3 Conclusion
In every case, both kneser-Ney and modified kneser-Ney smoothing methods yielded lower cross-entropy than using Jelinek-Mercer. In best cases, modified kneser-Ney smoothing increased results by more than 0.4 bits/symbol.
7.3 Conclusion

Figure 7.2: Jelinek-Mercer - Jazz authors

on the Beatles dataset, and by more than 0.9 bits/symbol on various jazz authors.

However, where Jelinek-Mercer always gives better or equal performances than the lower order models at any order, kneser-Ney smoothing does not generalize well at any order.

This can be explained by the fact that Jelinek-Mercer $\lambda$ interpolation coefficient are trained on the test set for every n-gram order while kneser-Ney and modified kneser-Ney use a fixed constant for discounting, independently of the dataset.
Figure 7.3: All methods - The Beatles
7.3 Conclusion

Figure 7.4: All methods - various jazz authors
Chapter 8

Application

8.1 Artists comparison

Using n-gram modeling can be used to detect different songs and similar songs inside an album, or between artists. In the following experiments (as shown on fig. 8.1, the “held one out” method was used : each one of the artist was considered as the test set while all the remaining artists were considered as the training set. In other words, this shows how difficult it is to predict the songs of an artist given the others.

The same approach can be used at the album level to identify special songs that are distant from the others within the same album, or at the artist level to differentiate albums.

8.2 Similarity network

A similarity network on a musical corpus is one possible application of modeling chord sequences. Musically speaking, similarity between two songs can reflect different elements. The song may have been composed by the same author or may just belong to the same musical style. Furthermore, as stressed in [18], the “Album effect” would be linked to the fact instruments, musicians and post-production are likely to be the same within the same album. Today, some online musical databases provide a graphical representation of artists sorted by their different musical styles. Links are explicitly represented between artists, and musical styles encompass them. The meta-data used for classification comes from human input (tags…). This information is dependent on the human perception of the different individuals involved in the characterization of the music. This kind of information is not always complete or accurate.

With chord sequences modeling, this application could be automated and graphically represent all the elements of a musical corpus given the same criteria in a fair fashion, not depending on human senses.
8.2 Similarity network

8.2.1 Implementation

Dataset

This application was realized using a dataset of previously mentioned jazz authors, transcribed to Harte's notation.

Model

For each author, a n-gram model was created, trained on each song and smoothed using the Jelinek-Mercer technique. Then the cross-entropy of this model was measured over every other authors. Note that this “distance” is not symmetric: the cross-entropy of A given B differs from the cross-entropy of B given A. Thus, a bidirectional graph was used to represent this corpora.

8.2.2 Graph

Distance between two songs on the represented graph 8.2 reflects the distance\footnote{Defined by the cross-entropy of the models} of their chord progressions in the model.
Figure 8.2: Similarity graph
Chapter 9

Proposed Tracks

9.1 Symbols for chords representation

As previously introduced, the number of possible symbols for representing chords has an impact on the models. Some further simplifications could be considered.

Many chords with different names are in fact the same chord considered differently. For example, an Am7/C and C6 are both composed of the following notes: C A E G. The only difference is that the notes are not necessarily in the same order in both chords. Therefore considering only one notation for equivalent chords can be used to reduce the number of labels representing them.

In musicology [1], chords can be classified by “harmonic functions” relative to different musical modes, where several chords with the same structure and main notes in common would be considered as equivalent.

9.2 Rhythm information

Rhythm is an important aspect of human perception of music. The same song played at different temp may sound differently. As we focused exclusively on chord sequences, the only aspect we (voluntarily) model are the underlying musicological features. Thus, two songs with similar harmony progression but with different rhythms would be still be considered as similar in our model. To give realistic performances in term of music recognition according to human ears, beat information (rhythm) could be used along with our models to provide a more accurate modeling, and particularly closer to human perception of musical genres. The authors of [12] proposed a model including beating information. This could be incorporated with n-gram modeling.
9.3 Modulation

Depending on the considered musical style, the same song can incorporate several tonalities. The fact of changing tonality is called modulation. It is widely used in jazz as well as musically advanced genres (using rich “vocabulary” of chords). The way modulation is performed in a song could give information about characteristics of a particular artist, or a particular musical genre. Modulation information could also help modeling more accurately the dependencies between chords and between tonalities.
Chapter 10

Conclusion

The main aspects in term of music information retrieval (MIR) have been presented, notably Acoustic Modeling and Language Modeling.

Possible models and approaches permit a wide variety of new applications. N-gram modeling is one of the main used models in the language modeling field, and has been shown to give realistic performances for chord sequences. The main limitations of previous n-gram approaches have been identified as well as some possible improvements. In this extent, our work focused on various smoothing methods over different chords labelling schemes and datasets. As a first step, we tried different smoothing methods and compared them. As the efficiency of smoothing methods depends on the context (model order, training sets), tests involved different situations. We used two distinct datasets : 300 songs from various jazz authors [17] and the discography of the Beatles. We show that newly introduced smoothing methods from the language recognition field known as kneser-Ney and modified kneser-Ney perform well and even outperform the previously introduced Jelinek-Mercer method. We presented two possible applications of n-gram modeling on a corpus of songs. One is a comparison of individual artists with a mass of other artists. The other is a similarity network on a corpus.

In order to improve our models, some improvements may be considered in future works. Although modern music is centered on chords progression, no previous study seem to exist about chords representation based on musical modes or such chords classification from music theory. This approach could sensibly reduce the number of symbols while keeping an accurate representation of chords. Also, some features from other models could be incorporated, such as rhythm and modulation.

At the personal level, this was a rich experience in a brand new domain for me. As a musician, I naturally enjoy music. Having a scientific approach over music was interesting and helped me to increase my knowledge in both computing, research methodology and musicology. This was my first re-
search experience as well, and confirmed my willing to continue in this area which corresponds to my aims.
Bibliography


