The Prediction of Merged Attributes with Multiple Viewpoint Systems

Thomas Hedges and Geraint A. Wiggins

Queen Mary University of London

13th June 2016

Acknowledgements

The authors would like to thank Marcus Pearce for valuable discussions and feedback. This work is supported by the Media and Arts Technology programme, EPSRC Doctoral Training Centre EP/G03723X/1.

Abstract

Multiple viewpoint systems find statistical structure in multidimensional entities, such as music, by combining Markov-based models together in order to make probabilistic predictions. This paper empirically tests two contrasting techniques for predicting multiple attributes of a musical surface. The first, an established method, predicts each attribute in turn, whilst a second, a proposed alternative, merges attributes into a new representation in order to make predictions simultaneously. A set of optimal smoothing techniques are found for both prediction methods across several harmonic and melodic datasets. Results indicate that when surface attributes are highly correlated, predicting merged attributes outperforms predicting the attributes separately. This can allow viewpoint systems with correlated surface attributes to be optimised, giving a closer fit with the training data as measured by mean information content.

Keywords— Multiple viewpoint, Harmony, Information theory

1 Introduction

Music is fundamentally a multidimensional entity, whether considered at the level of the sound wave, or at the symbolic level. A complex sound wave is highly multidimensional, requiring frequency component analysis before useful information can be extracted. At the symbolic level, most Western tonal music (the focus of this paper) can be described as a sequence of notes, which themselves consist of dimensions such as pitch, onset, duration, dynamic, and timbre. Alternatively, since Western tonal music often consists of notes occurring simultaneously, a useful description could be as a sequence of chords. In this case, the dimensions could be the root, the bass (the lowest note), the set of pitches in the chord, the set of pitch *classes* in the chord, and the chord type (a label for the set of pitches in the chord). This work focusses on music at the symbolic level and on chord sequences in particular.

Multiple viewpoint systems (Conklin & Witten, 1995) are a useful way of tackling the multidimensional nature of music at the symbolic level. The different dimensions of a musical surface are represented by *basic attributes*, which are be modelled by *viewpoints*. The majority of research using multiple viewpoint systems predicts a single basic attribute, usually pitch (Conklin & Witten, 1995; Pearce, 2005). However, for chords, a natural minimal representation is to divide a chord symbol (e.g. C^7) into a root (C) and chord type (7) which describes the set of pitches in the chord. A key question in the current research is how to handle predicting multiple basic attributes at the same time. Usually, multiple basic attributes are predicted separately (Conklin, 1990, pp. 68-69). The current work explores an alternative method where basic attributes are merged into a single attribute, allowing them to be predicted simultaneously. The key contribution of this research is to compare these two methods quantitatively for a range of datasets and domains, and establish under which conditions they perform best.

A secondary contribution of this work is an extensive empirical test of *n*-gram techniques known as smoothing (Pearce & Wiggins, 2004) in melodic and harmonic domains. These methods, originating from data compression and natural language processing, have been comprehensively tested on monophonic melodic music, but a similar review of their performance for harmonic data has not been undertaken to the authors' knowledge.

This study continues a line of research in using multiple viewpoint systems as a cognitive model of musical expectation. Expectation has been identified as a central aspect of music cognition in eliciting meaning and emotion from music as well as learning musical structure (Huron, 2006; Meyer, 1956; Narmour, 1990). The Information Dynamics of Music (IDyOM) model (Pearce, 2005) is an empirically tested cognitive model of musical expectation, working primarily with melodic prediction. In part, this research hopes to extend this approach to the prediction of tonal harmonic sequences. The present research uses the existing IDyOM implementation of Pearce (2005), freely available as open source software,[1] with modifications and additional viewpoints for representing chord sequences.

[1]https://code.soundsoftware.ac.uk/projects/idyom-project

After reviewing related research (§2) and establishing a representation scheme for the corpora used (§3), this paper presents three experiments. Experiment 1 (§4) finds a set of optimal smoothing parameters for two harmonic and three melodic datasets. Having presented an alternative method for predicting multiple surface (or basic) attributes, Experiment 2 (§5) compares the established method of predicting multiple basic attributes separately against predicting them at the same time as a single merged attribute. Finally, Experiment 3 (§6) tests if the results from Experiment 2 hold when applied to a full multiple viewpoint system.

2 Related Research

The modelling of tonal harmony with computational models is an established topic of research in computer science, cognitive science, and musicology. To position the current research, most computational approaches to modelling harmony can be considered as processing information in either a *top-down* or *bottom-up* manner. Bottom-up approaches make minimal assumptions of the domain, learning structure and regularities purely from the musical surface, whereas typically top-down models define rules which describe or parse the musical surface.

2.1 Computational Models of Harmony

One of the earliest computational approaches to modelling harmony is a top-down technique by Ulrich (1977), who tackles the problem of generating melodic jazz improvisations to a chord structure. Functional harmonic analysis is an important part of the system. It is achieved with a chord grammar used to label chords and a key analysis which attempts to parse a piece into the minimal number of segments that comply with a set of music theoretic rules. Ebcīoğlu (1986) approaches a similar problem, harmonising Bach chorales at the level of individual voices. 350 production rules, constraints, and heuristics in first-order predicate logic form are carefully derived from the chorale set (Riemenschneider, 1941) and various harmonic treatise. The hand-selected output of the system is stylistically convincing and could be described as approaching that of a competent music student. However, the fact that the number of production rules (350) almost approaches the number of chorales the knowledge base is derived from suggests that the system is unlikely to generalise well or be a plausible cognitive model for musical composition and understanding. Pachet (2000) presents a jazz chord sequence parser based on identifying common pre-defined harmonic patterns (or *shapes*) and labelling them as part of a hierarchical structure. Steedman (1984) uses a small set of six transformation rules to form a generative grammar capable of generating a large number of variations on a 12-bar blues. Generative grammars are also used by Rohrmeier (2011) who builds on and formalises aspects of the Generative Theory of Tonal Music (Lerdahl & Jackendoff, 1983) and tests the model on parsing examples of Western Tonal music. Rules are presented in a hierarchical structure consisting of phrase, functional scale-degree, and surface levels. It is often the case that strict (non-statistical) rule-based systems are unable to handle ambiguous cases, or are not able to quantify how typical a sequence is for a given style. Granroth-Wilding and Steedman (2014) address this problem with a model combining a *combinatory categorical grammar* and a statistical supervised learning approach trained on a small annotated jazz corpus. The problems of chord function and tonal centre labelling are approached with a preference rule system by Temperley (2001). The preference rules formulate musicological principles, such as accepting chord transitions which minimise distance on a circle of fifths.

Bottom-up approaches are usually unsupervised, often learning structure with statistical models such as Markov (*n*-gram) models or Bayesian networks. An early Markovian approach to modelling harmony is described by Ponsford, Wiggins and Mellish (1999), who generate novel pieces from *n*-gram models trained on seventeenth century dances. Perez-Sancho, Rizo and Inesta (2009) successfully apply Markov modelling to a supervised classification problem; identifying jazz composers from their chord sequences. In the domain of computational creativity, Pachet and Roy (2014) use *Markov constraints* (Pachet & Roy, 2011) to generate novel harmonisations in hybrid styles for jazz lead sheets. Rohrmeier and Graepel (2012) test a number of statistical models such as a Hidden Markov Model (HMM), an autoregressive HMM, a Dynamic Bayesian Model (DBN) and a *feature-based* (or multiple viewpoint) model on a corpus of jazz lead sheets. A DBN which incorporates information of features outperforms the Markovian models, but deteriorate when temporal information is added to the model. Another DBN model is presented by Paiement, Eck and Bengio (2005), who models jazz chord sequences with a hierarchical graphical model capable of capturing global structure, local dependencies and chord substitutions.

The current research positions itself firmly within the bottom-up approaches to modelling harmony with an unsupervised, Markovian, multiple viewpoint system. The application of multiple viewpoint modelling to the harmonic domain is not novel in itself. Whorley (2013) builds a multiple viewpoint system to harmonise Bach chorales at the note level, representing chords as vertically aligned pitches (Conklin, 2002). It is important to note that the representation level of the present research is at the chord symbol level, rather than at the level of individual voices. This removes the need to generalise over the many possible permutations for voicing a chord. Conklin (2010) uses a multiple viewpoint system over functional chord symbols to identify distinctive, idiomatic patterns in music. This work aims to develop this approach further by investigating different harmonic viewpoints, proposing and testing methods for predicting basic attributes and testing a collection of smoothing techniques (Pearce & Wiggins, 2004) to aid *n*-gram prediction.

Further work by Whorley, Rhodes, Wiggins and Pearce (2013) deals specifically with the order in which multiple basic attributes are predicted. As notes in voices are predicted they marginalise the remaining probability distributions predicting the other voices. The manner in which this marginalisation occurs affects the model performance measured by mean information content. However, when building a computational model as a method for investigating a cognitive process, the order in which different attributes are processed cannot be assumed. An approach which doesn't use predicted information from the current event is preferred in the current research, with basic attributes of a single event assumed to be predicted at the same time.

2.2 Statistical Learning of Sequential Data

Statistical, and especially Markovian, approaches to modelling sequential data frequently encounter two main problems when employing fixed-order models. Firstly, symbols which are novel to the context may be encountered, resulting in probabilities of zero being given to new events. This has been identified by Witten and Bell (1991) as the *zero-frequency problem*. Secondly, it is difficult to determine the context length (or model order) that will give the best predictions. In general, longer contexts can give more specific predictions, however, they are more likely to encounter sparsity issues than shorter contexts. Many approaches address this by combining models with different context lengths (Ron, Singer & Tishby, 1996), a naive implementation of which would result in polynomial time and space algorithms. A large number of proven methods and techniques have been established to tackle these problems, a selection of which are summarised below.

The well-known lossless data compression algorithm by Ziv and Lempel (1978), applicable to sequence prediction (Rissanen, 1983), parses left-to-right adding unique phrases to a dictionary used to construct a prediction tree. Encountering symbols novel to a given context (which may be empty) triggers a return to the root of the tree. A Prediction Suffix Tree (PST) (Bejerano & Yona, 2001; Ron et al., 1996) forms a suffix set consisting of all substrings in the training set not exceeding an order bound and occurring sufficiently frequently. Additionally, given the prediction of a symbol, suffixes are retained only if their maximum likelihood estimate is larger (defined by a user-defined parameter) than the corresponding parent suffix which is one symbol shorter. Context Tree Weighting (CTW) models (Willems, Shtarkov & Tjalkens, 1995) combines predictions from all suffix trees within a bounded depth with probability estimates calculated using a Krichevsky-Trofimov estimator (Krichevsky & Trofimov, 1981) with a computationally efficient recursive process. Originally implemented for binary alphabets, Volf (2002) proposes an extension to finite alphabets of arbitrary size with a hierarchical decomposition into binary decisions. Prediction by Partial Match (PPM) (Bunton, 1997; Cleary & Witten, 1984) is a lossless data compression method most efficiently built as a suffix tree with an online construction algorithm (Ukkonen, 1995). Various escape and smoothing methods can be applied with the method, most commonly predictions from different context lengths are combined recursively with a weighted prediction of the $(n-1)^{th}$ and $(n-2)^{th}$ length contexts. Cleary and Teahan (1997) propose an unbounded version called PPM* which dynamically selects the best order bound at each step. In terms of data compression performance metrics, Begleiter, El-Yaniv and Yona (2004) provides a comprehensive comparison of the Lempel-Ziv, CTW, PST and PPM algorithms over English text (the Calgary copurs), music in MIDI format and protein sequences. CTW and PPM were found to perform best with an average log-loss of 3.02 / 3.03 bits/symbol for English text, 1.21 / 1.30 bits/symbol for MIDI files and 4.56 / 4.48 bits/symbol for protein sequences.

Similar methods have been developed as the underlying mechanism behind symbolic music generation systems capable of composing or improvising in any style given an appropriate training corpus. For modelling musical style and generation, Dubnov, Assayag, Lartillot and Bejerano (2003) compares Lempel-Ziv and PST methods, finding that Lempel-Ziv is able to run in real time and find musically coherent motifs. However, it is also prone to replicating large sequences of the training data between joined with unexpected juxtapositions. Conversely, PST creates interesting transitions between replicated sequences, although may produce out of style notes and cannot be run in real time. Pachet (2003) presents an interactive music generation system built on prefix trees which learns all substrings of the training corpus and runs in real time. Musical generations are composed through a random walk method, falling back to shorter contexts in the prefix tree if the relevant context has not been seen in the training data. The system is reported to be able to produce fast tempo jazz improvisations which are stylistically indistinguishable from the user's input. Comparably, factor oracles (Assayag & Dubnov, 2004) are an acyclic automaton with a minimal number of states and a number of transitions linear to the length of the training sequence. They are capable of weak factor recognition, recognising all substrings in the training sequence, but also contain substrings not in the training data. Assayag and Dubnov (2004) note that factor oracles do not have a probability distribution over the alphabet at each state. However, long generated sequences should become asymptotically close to the observed training data using a generation algorithm which chooses stochastically between replicating a substring from the training corpus or jumping to a maximal suffix of the string sequence generated so far.

2.3 Information Dynamics of Music

The IDyOM model (Pearce, 2005) is a cognitive model of expectation for the perception of musical events and is applied in the current research as a statistical learning approach to modelling harmony. Expectation is an important aspect of music cognition (Meyer, 1956) and is quantified on an event by event basis by IDyOM with a probabilistic model. Events occupy a basic event space, ξ, roughly equivalent to the *musical surface* (Lerdahl & Jackendoff, 1983). These may be individual notes (Pearce, 2005), collections of simultaneous notes (Whorley, 2013), or, in the case of the current research, chord symbols. The multidimensional nature of music is captured with a multiple viewpoint system, with events comprising of multiple dimensions, or basic attributes. A multiple viewpoint system predicts one or more basic attributes, referred to as *target attributes*, from the event using a set of viewpoints. The advantage of such a system is that it combines the performance of individual expert models in a way which outperforms a single model by overcoming problems of sparsity and poor generalisation from the training data.

2.3.1 Viewpoints

Formally, a viewpoint signified by a type, τ, consists of a partial function, $\Psi_\tau : \xi^* \rightharpoonup [\tau]$, mapping sequences from the event space to symbols in the viewpoint alphabet, $[\tau]$, and a context model capable of calculating probabilities of sequences in $[\tau]^*$ (Conklin & Witten, 1995). The characteristics of the partial function provide a useful categorisation for viewpoints. For *basic viewpoints* the partial function acts simply as a selector function, selecting the relevant basic attribute element from the current event. *Derived viewpoints* apply some operation to the preceding sequence of basic attributes, commonly a difference function (e.g. `cpint` in Pearce, 2005) or a non-injective, surjective (many-to-one) function (e.g. `contour`) which groups similar elements into the same category. Derived viewpoints are defined as all non-basic viewpoints. *Threaded viewpoints* allow for patterns from non-adjacent events to be modelled, for example `thrbar` (Pearce, 2005) models pitch interval between the first notes of adjacent bars. Relevant events for threaded viewpoints are selected by *test viewpoints*, which simply return boolean values. Single viewpoints may be combined to form a *linked viewpoint* to explicitly model interactions between them. These are denoted by $\tau_1 \otimes ... \otimes \tau_n$, where n, the number of constituent viewpoints, is usually capped for a multiple viewpoint system. The viewpoint alphabet of the linked viewpoint is the cross product of its constituent viewpoints: $[\tau] = [\tau_1] \times ... \times [\tau_n]$.

2.3.2 Probability Calculations and Performance Metrics

The probabilistic model at the core of IDyOM takes a Markovian approach, with the goal being to estimate the probability function $p\left(e_i \mid e_1^{i-1}\right)$, in other words assigning a probability to every symbol in a sequence, given preceding symbols. A sequence of symbols from a viewpoint of type τ, running from indices i to j, is expressed as e_i^j, where symbols are drawn from a finite alphabet, $e \in [\tau]$, specific to that viewpoint. Assuming that the probability of an event is affected only by the previous $n-1$ events, these probabilities can be assigned with an n-gram model (also known as an $(n-1)^{th}$ order model) using the *maximum likelihood estimate* (Equation 1).[2] Trivially, the probability of a whole sequence is then estimated using the chain rule (Equation 2). In practice, IDyOM makes use of a collection of techniques known as *smoothing* which modify a standard Markov model (see §4.1).

$$p\left(e_i \mid e_{i-n+1}^{i-1}\right) = \frac{c\left(e_i \mid e_{i-n+1}^{i-1}\right)}{\sum_{e \in [\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)} \tag{1}$$

$$p\left(e_1^J\right) = \prod_{i=1}^{J} p\left(e_i \mid e_{i-n+1}^{i-1}\right) \tag{2}$$

Following a *natural language processing* approach (Manning & Schütze, 1999), the quality of a probabilistic model can be measured by the degree to which its probability function describes the data. *Information content* (Mackay, 2003; Shannon, 1948) is a useful performance metric, representing an estimate of the number of bits required to describe an event drawn from a discrete probability distribution. Mean information content can also be a measure of the *cross entropy* between two distributions (Manning & Schütze, 1999), even when one probability distribution is unknown. In other words, the degree of fit between the probability distribution of the model and the true probability distribution of the stochastic process that generated the training data.

[2] $c\left(e_i \mid e_{i-n+1}^{i-1}\right)$ is a count of the number of occurrences of the symbol e_i following the subsequence e_{i-n+1}^{i-1}.

The information content of a single event is given by Equation 3 and the mean information content of a sequence of length J by Equation 4. An information theoretically efficient model will return a low mean information content, resulting from relatively high probability estimates for events, suggesting a close fit between the model and statistical structure underlying the training data. Typically, the mean information content of a corpus is calculated with a k-fold cross validation (Conklin & Witten, 1995; Pearce & Wiggins, 2004), with the probability function $p\left(e_i \mid e_1^{i-1}\right)$ estimated from k training sets and the mean information content calculated from the corresponding testing sets.

$$h\big(e_i \mid e_{i-n+1}^{i-1}\big) = -log_2\, p\big(e_i \mid e_{i-n+1}^{i-1}\big) \tag{3}$$

$$\bar{h}\big(e_1^J\big) = -\frac{1}{J}\sum_{i=1}^{J} log_2\, p\big(e_i \mid e_{i-n+1}^{i-1}\big) \tag{4}$$

2.3.3 LTM and STM

Conklin and Witten (1995) introduce the notion of a long-term model (LTM) and short-term model (STM) to model global and local statistical structure respectively. The LTM is built from the held-out training set of the k-fold cross validation, whilst the STM model is built dynamically on an event-by-event basis for each composition in the test set and then discarded after the composition has been processed. A third type of model, LTM+, merges qualities from both models, building both from the training set and dynamically on an event-by-event basis. The current research follows both Conklin and Witten (1995) and Pearce, Conklin and Wiggins (2005) in combining predictions in two stages: firstly viewpoint predictions within the LTM(+) and STM, and secondly combining the predictions from the LTM(+) and STM themselves.

3 Corpora and Representation

To investigate methods for predicting multiple basic attributes and the effect of different smoothing techniques on a variety of data, five symbolic datasets across melodic and harmonic domains are selected (Table 1). Dataset 1 consists of the chord sequences of 348 jazz standards from the original *Real Book* (Leonard, 2012), compiled by Pachet, Suzda and Martín (2013). Compositions from the original source with ambiguous structure or section orders were not included in the experimental dataset. A second harmonic dataset comprises of the chord sequences from all 179 Beatles songs, compiled by Harte, Sandler, Abdallah and Gómez (2005). The three melodic datasets, all used by Pearce and Wiggins (2004), are a set of 185 Bach chorale melodies from Riemenschneider (1941), 556 German folksongs from the Essen Songbook Collection (Schaffrath, 1995), and 152 Canadian folksongs ballads from Nova Scotia (Creighton, 1966). Although the primary goal of the current research is to model harmonic sequences, the melodic datasets are included to check the generality of the techniques tested.

3.1 Basic Harmonic Attributes

Events in the harmonic domain are equivalent to the *chord symbols* of a *lead sheet* (e.g. G^7, $C\sharp^6$, $A\flat$), which can be represented fully by the basic attributes `Root`, `ChordType`, and `PosInBar`. A pre-processing step is required to reduce the alphabet size of the original `ChordType` attribute (67 for the *Real Book*) and incorporate slash chord notation (e.g G^7/C). Each chord is converted to a *pitch class set* (see Forte, 1973) and transposed so that the root (signified by the chord symbol prefix) is 0. In the case of slash chords, the bass note (the note name after the slash) is considered to be the root unless it is already present in the chord, or if it is a minor or major 7th (10 or 11 semitones) above the root of the chord. This implies that the bass note of a slash chord signifies only a change of inversion, but not function, if it is present in the original chord (Levine, 1995, Ch. 5). Finally, Algorithm 1 (see Appendix) assigns one of 13 symbols to a given pitch class set according to the combination of pitch classes in the set. Functionally equivalent chords with differing notation can, therefore, be represented by the same symbol. For example $C^{11}(no3rd)$ and G^7/C both represent 11th chords with an omitted major 3rd and a functional root of C. All 13 `ChordTypes`, including the special symbol *NC* (signifying that no chord is being played), are given in Table 2 alongside corresponding typical chords from the *Real Book* source.

`Root` has a full alphabet size of 13 and represents the pitch class (assuming enharmonic equivalence) of the functional root of the chord after pre-processing the `ChordType`. Again, the special symbol *NC* denotes that no chord is played, therefore, by definition when `Root` is *NC*, `ChordType` is also *NC* and vice versa.

`PosInBar` signifies the metrical position in the bar of the chord onset. A *timebase* (see Pearce, 2005, p. 63) of 2 allows all metrical positions to be expressed as an integer, where 0 represents the first beat of the bar and 2 is a crochet (quarter-note) into the bar. The full alphabet of `PosInBar` in dataset 1 is $\{0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$, and $\{0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$ for dataset 2. By definition, the start of every bar in the harmonic datasets has a chord onset, if a chord is repeated consecutively within a bar it is ignored. This gives the `PosInBar` alphabet a dynamic quality, as it shrinks throughout each bar. For example, if a chord on the final beat of the bar has been processed, the next chord cannot occur on a second beat, it must occur on the first beat of the next bar.

3.2 Basic Melodic Attributes

`Pitch` and `Duration` are the basic attributes used to describe notes in the melodic datasets. `Pitch` (equivalent to `cpitch` in Pearce, 2005) can be thought of as the MIDI number of a note, in other words ignoring pitch class equivalence, but assuming enharmonic equivalence. The full alphabet sizes of `Pitch` for datasets 3, 4 and 5 are 21, 37 and 26 respectively. `Duration` is an integer signifying the duration in timebase units of a note, with the full alphabet sizes for datasets 3, 4 and 5 being 14, 17 and 14.

3.3 Derived Harmonic Viewpoints

A pool of derived viewpoints are defined to improve the modelling of chord sequences in the current research. Seven viewpoints are derived from `Root` and three from `ChordType`. `RootInt` is the root interval in semitones modulo-12 between two adjacent chords. If either `Root` symbol is a *NC* then the symbol -1 is returned. `MeeusInt` categorises root movement (`RootInt`) using root progression theories (Meeus, 2000; Rameau, 1971; Schoenberg, 1969), which describe harmonic progression exclusively through root transitions. Meeus (2000) simplifies all root progressions to a set of two: dominant progressions descend by a perfect fifth, descend by a third or ascend by a second, and subdominant progressions rise by a perfect fifth, ascend a third or descend a second. Conklin (2010) defines a similar viewpoint, `meeus`, although the current research uses a slightly different definition owing to the fact that enharmonic equivalence is assumed for `Root`. Dominant root progressions (where `RootInt` = 1,2,5,8 or 9) are given the symbol 1, subdominant progressions (`RootInt` = 3,4,7,10,11) the symbol -1, no root movement (`RootInt` = 0) the symbol 0, a `RootInt` of 6 (the diminished fifth/ augmented fourth) the symbol -2, and when either root is *NC* the symbol -3. A `ChromaDist` viewpoint aims to exploit the notion that tonal harmony progresses mainly in intervals of a perfect fifth (Chew, 2002; Longuet-Higgins, 1979; Piston, 1948; Rameau, 1971; Riemann, 1895). `ChromaDist` is simply the minimum number of perfect fifths required to get from one root pitch class to the next, or the smallest distance around a cycle of fifths. Again, if either `Root` is *NC*, then -1 is returned. All of these viewpoints return the undefined symbol, $\perp$, for the first event of a piece, when the previous event does not exist.

Three further viewpoints, `RootIntFiP, MeeusIntFiP` and `ChromaDistFiP`, apply `RootInt, MeeusInt` and `ChromaDist` to the current event and the first event of the piece instead of the previous event. This provides a crude measure of capturing some structure from non-adjacent events. As there is no reliable representation for key or tonal centre in the corpora (especially considering modulations are frequent) there is no way of directly representing the harmonic function or scale degree of a chord. `RootIntFiP` can provide a rough estimate of chromatic scale degree, given that pieces frequently begin on the tonic chord. Of course, there are many occurrences where this is not the case, so `RootIntFiP` should not be considered a pure representation of scale degree, merely that it is able to capture some of the statistical properties associated with it. Finally, a threaded viewpoint, `RootInt` $\ominus$ `FiB`, aims to capture further non-local structure by measuring the `RootInt` between chords on the first beats of successive bars.

Levine (1995) notes that the properties of the third and seventh in a chord are the most important indicators of its function. With this in mind, three viewpoints derived from `ChordType` group chords into different categories according to their pitch class set contents. `MajType` simply assigns a 1 to all chords where the third is major (4 is a member of the pitch class set), a 2 to all chords where the third is minor, and a 0 to all chords without a third. `7Type` assigns a 0 to all chords with a minor 7th and a 1 to all other chords (except a *NC* which is given a -1 symbol). Finally, `FunctionType` assigns all chords with a major third and minor seventh a 0, all other chords with a major third a 1, all chords with a minor third and minor seventh a 2, all other minor chords a 3, and *NC* a -1. Roughly, these four categories correspond to dominant, tonic major, pre-dominant,[3] and tonic minor chord functions. Table 3 summarises all of the harmonic viewpoints used in this research over a sample chord sequence.

4 Experiment 1: Optimal Smoothing Parameters for Harmonic and Melodic Domains

Of the statistical learning methods discussed in §2.2, PPM (used by IDyOM) offers a number of advantages over similar methods which are attractive for the current research. PPM is among the best performing lossless data compress algorithms available (Begleiter et al., 2004; Bunton, 1997; Shkarin, 2002), and whilst not the most efficient in terms of time and space complexity, this does not concern the current research which is not required to run in real time (c.f. Assayag and Dubnov, 2004; Pachet, 2003). The method does not require hand-tuned parameters (c.f. Bejerano and Yona, 2001; Ron et al., 1996) or bounded context lengths if the unbounded PPM* variant (Cleary & Teahan, 1997) is used. Finally, with the PPM framework various smoothing and escape methods can be easily implemented (Bunton, 1996, ch. 6) to optimise the algorithm for different domains corpora.

This experiment investigates the optimal smoothing parameters for linked viewpoints predicting separate attributes in a variety of harmonic and melodic datasets (Table 1). Pearce and Wiggins (2004) find an optimal set of smoothing parameters for a collection of monophonic melodic datasets, however, these are not guaranteed to apply to new datasets or domains. In the context of studying merged representations, it is necessary to first understand the effects of smoothing parameters on viewpoint models in different domains before comparisons are made across different forms of representation (§5).

4.1 Prediction by Partial Match and Smoothing Techniques

PPM uses a collection of techniques from data compression known as *smoothing* which improve the performance of Markov models, in particular tackling problems associated with zero-frequency counts (Witten & Bell, 1991) and fixed order models. In general, this is achieved by adjusting the maximum likelihood estimates to save probability mass for novel events and finding a way to combine models of different orders in a meaningful way.

Two frameworks exist to achieve this aim, commonly referred to as *backoff smoothing* (Kneser & Ney, 1995) and *interpolated smoothing* (Chen & Goodman, 1999; Jelinek & Mercer, 1980). Both frameworks utilise a global order bound, g, to recursively combine predictions from the g^{th} order down to the -1^{th} order. $\alpha(e_i \mid e_{i-n+1}^{i-1})$ represents the *prediction probability*, essentially an adjusted maximum likelihood estimate (Equation 1). $\gamma(e_{i-n+1}^{i-1})$ is the *escape probability*, or the amount of weight given to lower order models. $t(e_i^j)$ is the *type count* of a sequence and

[3]A chord which precedes the dominant, typically ii or IV.

returns the number of different symbol types seen after the sequence e_i^j. $t(\epsilon)$ is the type count of the empty sequence; in other words, the total number of symbol types already seen by the model. The fundamental difference between backoff and interpolated smoothing is that backoff smoothing (Equation 5) escapes to the next order only when it encounters a novel symbol for a given context, whilst interpolated smoothing (Equation 6) always escapes to the lower order, blending predictions from the $(n-1)^{th}$ and $(n-2)^{th}$ orders recursively until termination.

$$p\big(e_i \mid e_{i-n+1}^{i-1}\big) = \begin{cases} \dfrac{1}{\mid \xi \mid +1 - t(\epsilon)} & \text{if } n < 1 \\ \alpha\big(e_i \mid e_{i-n+1}^{i-1}\big) & \text{if } c\big(e_i \mid e_{i-n+1}^{i-1}\big) > 0 \\ \gamma\big(e_{i-n+1}^{i-1}\big) \cdot p\big(e_i \mid e_{i-n+2}^{i-1}\big) & \text{otherwise} \end{cases} \tag{5}$$

$$p\big(e_i \mid e_{i-n+1}^{i-1}\big) = \begin{cases} \dfrac{1}{\mid \xi \mid +1 - t(\epsilon)} & \text{if } n < 1 \\ \alpha\big(e_i \mid e_{i-n+1}^{i-1}\big) + \gamma\big(e_{i-n+1}^{i-1}\big) \cdot p\big(e_i \mid e_{i-n+2}^{i-1}\big) & \text{otherwise} \end{cases} \tag{6}$$

4.1.1 Unbounded Context Lengths

Many of these smoothing techniques were developed as variants of PPM, a leading data compression scheme. The original algorithm proposed by Cleary and Witten (1984) used backoff smoothing with a fixed order bound and made use of two escape methods: A and B (see §4.1.2). Variations of PPM relevant to the current research include additional escape methods (Howard, 1993; Moffat, Neal & Witten, 1998; Moffat, 1990), the use of interpolated smoothing (Bunton, 1997), and *unbounded length contexts* (Cleary & Teahan, 1997), also known as PPM*. Unbounded length contexts remove the need for a fixed order bound by exploiting the fact that novel symbols tend to occur less frequently after *deterministic* contexts (ones which are followed by only one symbol type, i.e. $t(e_i^j) = 1$) than *non-deterministic* contexts (where $t(e_i^j) > 1$) when compared to a uniform prior distribution (Cleary & Teahan, 1995). Cleary and Teahan (1997) propose that for unbounded length contexts an order bound can be found dynamically for each symbol in a sequence by selecting the shortest deterministic context as the order bound, or, if no such context exists, the longest matching context.

4.1.2 Escape Methods

Escape methods are different methods for calculating $\alpha\big(e_i \mid e_{i-n+1}^{i-1}\big)$ and $\gamma\big(e_{i-n+1}^{i-1}\big)$, determining the amount of weight assigned to novel events for a given context. Table 4 summarises five escape methods reviewed and empirically tested by Pearce and Wiggins (2004). Method A (Cleary & Witten, 1984) effectively assigns a count of one to all novel events given a context. Method B (Cleary & Witten, 1984) introduces the type count $t(e_{i-n+1}^{i-1})$ to the escape probability, so that more weight is given to novel symbols occurring after a context that is usually followed by more symbol types. Additionally, the effect of anomalies is reduced by subtracting one from the symbol count of the prediction probability so that novel symbols must occur twice before they are counted. Moffat (1990) proposed method C (also known as *Witten-Bell smoothing*) as a hybrid of the previous two methods, with the escape probability adjusted by the type count as in method B, but the symbol count of the prediction probability unaltered as in method A. Forcing symbols to occur twice before they are counted by subtracting one from the prediction probability is considered wasteful. Howard (1993) proposes method D as a compromise, which subtracts only half from the symbol counts of the prediction probability. Finally, method AX (Moffat et al., 1998) is a simplified and corrected version of method P (Witten & Bell, 1991), which assumes the occurrence of novel events follows a Poisson distribution. Note the special type count $t_1(e_{i-n+1}^{i-1})$ signifies the number of symbol types which have appeared only once after a given context. In data compression studies, methods similar to method AX tend to outperform methods C and D, with methods A and B performing worst (Bunton, 1997; Cleary & Teahan, 1997; Moffat, Sharman, Witten & Bell, 1994; Witten & Bell, 1991). Since various qualities of the training and test data such as alphabet size and skew (Moffat et al., 1994) have an impact on the performance of different escape methods, there is no informed way of selecting an escape method without *a priori* knowledge of the corpus (Witten & Bell, 1991). The optimal escape method can, therefore, only be found with an experimental approach.

4.1.3 Update Exclusion

Update exclusion is a method proposed by Cleary and Witten (1984) which aims to improve probability estimates with an altered counting scheme for *n*-grams. The rationale behind the method is that, when escaping down to lower orders, *n*-grams which would have already been seen at higher orders (therefore preventing escape in the case of backoff smoothing) are still included in calculating predictions for the lower order models. This wastes a portion of the probability mass which would otherwise be assigned to possible predictions. To borrow an example from Cleary and Teahan (1997), the task is to give a probability estimate to the symbol *'d'* following the sequence *'abracadabra'* with a backoff model, with an order bound of $g = 2$. A 3-gram model will escape to the lower order, since *'d'* does not occur in the context of *'ra'* in the sequence. Without update exclusion, a simple maximum likelihood estimate would assign a probability of $\frac{1}{4}$ for the 2-gram model, as the context *'a'* occurs four times, and is followed by a *'d'* on one of those occasions. If update exclusion is used, the 2-gram model will give a maximum likelihood estimate of $\frac{1}{3}$, since *'c'* has already been seen in the context of *'ra'* and is therefore removed from the predictions following the context *'a'*.

4.1.4 IDyOM Smoothing Parameters

Pearce and Wiggins (2004) established empirically the optimal smoothing parameters for IDyOM on a variety of monophonic melodic datasets. Interpolated smoothing consistently outperformed backoff smoothing across all datasets and for most other smoothing parameter combinations. Method C was the most consistently high performing escape method, although method AX also performed well for the STM. The effect of update exclusion was found to be sensitive to other smoothing parameters, the dataset and whether the LTM or STM was being used. Unbounded length contexts (PPM*) outperformed the best fixed-order models when combined with interpolated smoothing, but with backoff smoothing, improvements were inconsistent. The LTM+ was found to outperform the LTM when both were given the best smoothing parameters. Overall, the best LTM+ and STM were both unbounded, used interpolated smoothing and escape method C, but not update exclusion.

Model are given a shorthand notation (Pearce & Wiggins, 2004) indicating their set of smoothing parameters. The long- and short-term models are depicted by LTM and STM respectively, with LTM+ representing the hybrid model (see §2.3.3). Escape methods are indicated by 'A', 'B', 'C', 'D', and 'X' (for method AX). The order bound is given by an integer, or '*' if unbounded. If update-exclusion is used, a 'U' appears next in the shorthand string. An 'I' shows the model uses interpolated smoothing, otherwise backoff smoothing is used. For example, the best performing models found by Pearce and Wiggins (2004) were LTM+C*I, a hybrid long-term interpolated smoothing model, using escape method C and unbounded context lengths, and STMC*I, a short-term model otherwise with the same parameters.

4.2 Experimental Design

This experiment aims to find the optimal smoothing parameters for various basic attribute combinations across different datasets. Future experiments require the prediction of two attributes simultaneously, therefore, the linked viewpoint of the two basic attributes is chosen for optimisation, as opposed to the basic viewpoints individually. The harmonic datasets each have three basic attributes, giving three possible combinations of linked viewpoints to test. Model performance is assessed by mean information content, $\bar{h}$ (Equation 4) calculated by a 10-fold cross validation of the dataset being assessed. The mean is taken over all events, rather than over all pieces.

In theory, it is possible that each viewpoint in a multiple viewpoint system has different optimal smoothing parameters for every viewpoint predicting every target attribute. However, as the pool of possible viewpoints in a system is large[4] the current study and previous research (Conklin & Witten, 1995; Pearce & Wiggins, 2004; Whorley, 2013) avoids this approach. Instead, all viewpoints are given the same smoothing parameters, although the LTM(+) and STM are optimised separately. The parameters to be optimised are interpolated/backoff smoothing, the escape method and use of update exclusion. Different global order bounds will not be investigated since unbounded context lengths (PPM*) make minimal assumptions on the dataset and domain and were found to perform consistently well by Pearce and Wiggins (2004). Furthermore, the LTM will not be used, instead, experiments will be run using the LTM+ and STM which is found to be the best combined model in Pearce and Wiggins (2004). For a given dataset and linked viewpoint, mean information content is calculated for all possible parameter combinations, with

[4]An upper bound estimate would be $\sum_{l=1}^{L} \binom{v_n}{l}$ where v_n is the number of single viewpoints and L the maximum number of links permitted in a linked viewpoint.

the lowest value of $\bar{h}$ signifying the optimal model.

4.3 Hypothesis

It is predicted that there will be some subtle differences in optimal parameters across different domains, although some techniques are expected to be universally beneficial: in particular escape methods C, D and X. Models using interpolated smoothing are expected to outperform those that do not, and prediction models using update exclusion are predicted to be less effective than those that do not (Pearce & Wiggins, 2004).

4.4 Results

The results are summarised in Table 5, using mean information content to compare the optimal backoff and interpolated smoothing models, as well as the best models with and without update exclusion. For the LTM+ it is clear that escape methods C and D are consistently effective, however, this pattern does not extend to the STM where methods A, C, D, and X are all optimal for at least one dataset and viewpoint combination. The performance of interpolated over backoff smoothing across all datasets was assessed by taking the best performing interpolated and backoff models for each of the 18 model, dataset, and viewpoint combinations. A one-sided paired Wilcoxon signed-rank test over $\bar{h}$ values confirmed interpolated significantly outperformed backoff smoothing by 0.055 bits/event ($N = 18, W = 167, z = 3.549, p < 0.001$). Likewise, models that didn't use update-exclusion significantly outperformed those that did by 0.077 bits/event ($N = 18, W = 149, z = 2.765, p < 0.01$).

4.5 Conclusions and Discussion

The variety of optimal parameter combinations suggests that differing domains and viewpoints do have an impact on the effectiveness of different smoothing techniques. Therefore, when comparing models across domains and viewpoints it is necessary to optimise each first. Without optimal smoothing parameters it is difficult to attribute any results to genuine differences in statistical structure or simply the varying impact of non-optimal smoothing parameters.

In general, the relative performance of smoothing techniques established by Pearce and Wiggins (2004) was upheld. Escape methods A and B performed poorly overall, method C performed well, update exclusion was found to slightly damage model performance, and interpolated smoothing outperformed backoff smoothing. One notable difference is the high performance of escape method D over C when predicting `Root`$\otimes$`PosInBar` or `ChordType`$\otimes$`PosInBar` with the LTM+. It is also interesting to note that escape method A was optimal for four of the nine STM tests undertaken, as it has been found to perform poorly in data compression (Cleary & Witten, 1984; Moffat, 1990) and melodic prediction (Pearce & Wiggins, 2004). The instability of optimal parameters for the STMs could be attributed to the fact that the STMs themselves are highly dependant on local, dynamic statistical structure to make predictions. Therefore, the local effects of each dataset and viewpoint combination have a varying impact on the performance of different smoothing techniques. Optimal smoothing parameters for each dataset and viewpoint found in this experiment are retained for future experiments.

5 Experiment 2: Predicting Merged Attributes from a Linked Viewpoint

In past research (Conklin & Witten, 1995; Pearce, 2005), multiple viewpoint systems have primarily been used to predict a single basic attribute (although this is not exclusively the case, e.g. Pearce, Mullensiefen and Wiggins, 2010). The current research requires the prediction of chord symbols, comprising of two attributes: `Root` and `ChordType`. This experiment empirically tests two methods for predicting multiple attributes with viewpoint systems: one that predicts attribute symbols separately and a proposed alternative that predicts merged attribute symbols. The optimal smoothing parameters for predicting merged attributes are found and then the two methods are compared.

5.1 Merging Basic Attributes

Traditionally, multiple viewpoint systems follow Conklin (1990, p. 69) when calculating probabilities of multiple basic attribute predictions. It is assumed that the basic attributes are statistically independent, so the overall probability of multiple attributes co-occurring is simply the product of the individual probabilities. Suppose a multiple viewpoint system models two basic attributes, τ_x and τ_y, predicted by the linked viewpoint $\tau_x \otimes \tau_y$. At a given point in a sequence the system is required to predict an event represented by the tuple $\langle X, Y \rangle$ from a probability distribution over $[\tau_x] \times [\tau_y]$. Prediction is done in stages for each basic attribute to be predicted, including the matching of symbols and contexts in the PPM model. Probabilities for all symbols in $[\tau_x] \times [\tau_y]$ matching X and then matching Y are calculated. The total probability of X is the sum of all probabilities where X matches, with an identical case for the total probability of Y. Assuming statistical independence, the probability of $\langle X, Y \rangle$ is the product of the two probabilities. In other words, separate predictions are both marginalised over the other basic attribute before being combined:

$$p(\langle X, Y \rangle) = \sum_{y \in [\tau_y]} p(X, y) \cdot \sum_{x \in [\tau_x]} p(x, Y). \tag{7}$$

An alternative method is proposed which merges the basic attributes before prediction so that the probability of the merged symbol is matched and calculated directly. In this sense, X and Y are matched simultaneously and $p(\langle X, Y \rangle)$ is calculated directly by the PPM model. A merged attribute simply combines multiple basic attributes into a single representation and is modelled by a linked viewpoint. Any number of basic attributes, τ_b, may be linked to form a merged attribute which will, therefore, have a domain of $[\tau_{b_1}] \times ... \times [\tau_{b_n}]$. A merged attribute can be predicted by linked viewpoints that contain the merged attribute in their type sets. A type set is defined as the *"basic types the viewpoint is derived from and is, therefore, capable of predicting"* (Pearce, 2005, p. 59). Originally, the type set of a linked viewpoint would be $\langle \tau_1 \otimes ... \otimes \tau_n \rangle = \{\tau_{b_1}, ..., \tau_{b_n}\}$ where τ_{b_1} is a basic viewpoint predicted by τ_1. A small adjustment to this definition is required to enable the prediction of merged attributes. The type set of a linked viewpoint is now the power set of its constituent viewpoints, for example, the type set of $\tau_1 \otimes \tau_2$ would be $\{\tau_{b_1}, \tau_{b_2}, \tau_{b_1} \otimes \tau_{b_2}\}$. Note that merged attributes may be predicted by linked viewpoints comprising of derived viewpoints, providing the merged attribute is contained within the type set. For example, the linked viewpoint `RootInt`⊗`MajType` may predict `Root`⊗`Chordtype`, but not `Root`⊗`PosInBar`.

Using merged attributes can be viewed as partly addressing issues concerning appropriate levels of representation. When formulating a multiple viewpoint system the appropriate basic attributes, or input representation, must be defined. In some cases it is clear that different dimensions of music should be modelled separately, for example, pitch and duration. However, for others an appropriate representation is less clear, for example, pitch could be represented as a MIDI note (as in `cpitch`) or with two basic attributes representing pitch class and octave number. These two representations contain identical information about the musical surface, however, their statistical properties are likely to be very different. Similarly, in the current research, `Root` and `ChordType` can be considered as two basic attributes or as a single attribute: `Root`⊗`ChordType`.

5.2 Experimental Design

The experimental design broadly follows §4.2. For each merged attribute in the datasets the optimal smoothing parameters are found with an exhaustive search of the escape method, interpolated/backoff, and update exclusion smoothing parameters (§4.1). Model performance is assessed by mean information content, $\bar{h}$, calculated with a 10-fold cross validation of each dataset. An 'M' on the end of the shorthand model description (see §4.1.4) signifies the model predicts merged rather than separate attributes. By way of example, the shorthand notation for an unbounded STM using escape method AX, backoff smoothing, and update exclusion to predict a merged attribute is STMX*UM.

5.3 Hypothesis

As the predictive linked viewpoints are all the same as §4, the optimal smoothing parameters should remain similar. Interpolated smoothing is expected to outperform backoff, models without update exclusion should perform better than those with, and escape methods C and D should perform consistently well at least for the LTM+. The performance of models predicting merged attributes will be compared to predictions of separate basic attributes. When the basic attributes are statistically independent, modelling with separate basic attributes should give truer probability estimates than with merged attributes. However, when basic attributes are highly correlated resulting in small areas of high probability density in the prediction distribution, it is expected that predicting merged attributes will outperform predicting separate attributes. This is because the merged prediction is able to take advantage of the areas of high probability density by matching both symbols directly. On the other hand, the marginalisation process required to predict separate attributes dilutes these areas of high probability in order to match both symbols independently. Therefore, it is hypothesised that when basic attributes are more correlated predicting merged attributes will be more effective.

5.4 Results

The optimal smoothing parameters for predicting merged attributes broadly follow the precedents established for separate attribute predictions (§4.4). Table 6 shows escape methods C and D dominate the LTM+ results, while C, D, and AX are the optimal escape methods for the various STMs. Interpolated smoothing outperformed backoff smoothing by 0.058 bits/event ($N = 18, W = 171, z = 3.724, p < 0.001$), and non-update exclusion performed better than update exclusion models by 0.070 bits/event ($N = 18, W = 138, z = 2.286, p = 0.011$).

A way of quantifying correlation between basic attributes must be established in order to test the relationship between basic attribute correlation and the performance of merged attribute prediction. A chi-squared test gives a good indication of correlation between two basic attributes, however, the test statistics, χ^2, of experiments with different sample sizes cannot be compared meaningfully. Cramer's V, $\Phi_c = \sqrt{\frac{\chi^2}{N \cdot df}}$, is an effect size statistic where N is the sample size (number of events), and df the degrees of freedom.

Additionally, a metric to quantify the difference in performance between the merged and separate attribute prediction methods is presented. Paired t-tests over all pieces show that almost all differences in $\bar{h}$ are statistically significant, except for the STM of `Root`$\otimes$`PosInBar` and `ChordType`$\otimes$`PosInBar` for dataset 2 (Table 6). However, in this case t-tests are not necessarily meaningful because the sample sizes are large ($N > 150$) resulting in high t values.[5] Instead, performance difference is quantified by Cohen's d, an effect size calculated by $\frac{\bar{h_1} - \bar{h_2}}{\sigma_{pooled}}$ where σ_{pooled} is the pooled standard deviation of both populations.

Figure 1 plots the relationship between basic attribute correlation and performance difference, confirming the general trend that more highly correlated basic attributes are better predicted as merged attributes. A linear regression confirms this trend, returning a significant effect ($df = 16, F = 12.540, p < 0.01$) and an R^2 value of 0.439.

[5]Note that it is not possible to infer from this the magnitude of the difference, only that the null hypothesis (that there is no difference between the means) can be rejected.

5.5 Conclusions and Discussion

The main results of this experiment are that, in certain circumstances, predicting a merged attribute rather than separate attributes is more effective. Those circumstances are that the basic attributes themselves are correlated with each other, creating areas of high probability density in the distribution. A clear example of this is predicting `Root` and `ChordType` in both harmonic datasets, for both the STM and LTM+. On the other hand, when attributes are not correlated, such as `Pitch` and `Duration` in all melodic datasets, it is often more effective to predict attribute separately. This result calls into question that basic attributes can be assumed to be statistically independent (Conklin, 1990, p. 69), showing they can be strongly correlated. An argument can, therefore, be made that basic attribute correlation should be measured in order to determine whether merged or separate basic attributes should be predicted by a multiple viewpoint system.

Although the overall effect of smoothing techniques on the datasets was found to hold for merged attribute prediction, there were differences in optimal parameters found between corresponding separate and merged prediction models, even though they use the same linked viewpoint model to make predictions. This may be because different smoothing techniques have differing impacts on the sparsity of models. For example, escape method A punishes the event probability more when escaping to lower orders compared to method B, so could give a sparser distribution when predicting a symbol novel to a long context. As discussed in §5.1, the relative sparsity of models is likely to have an impact on how well merged attributes are predicted.

It is interesting to note that all of the STMs for the melodic datasets found update exclusion to be effective, going against the general trend found so far in the current research. This suggests that those datasets share a property that makes counting n-grams with the adjusted exclusion method more effective. Since the alphabet sizes for the melodic datasets are larger than harmonic ones and an STM model is unlikely to come close to seeing all of the alphabet, the rate at which new symbols are seen is high and consequently the model frequently escapes down to lower order models. In this case, using an excluded count method may be beneficial as probability mass is preserved in the lower orders by excluding symbols that would have been seen in higher order models.

6 Experiment 3: Predicting Merged Attributes with Multiple Viewpoint Systems

Full multiple viewpoint systems such as IDyOM are complex models with several components. It is not always clear how individual components of the model will interact to give a final prediction, and therefore it follows that an improvement in one component of the model does not necessarily imply an overall improvement in performance. This final experiment tests the prediction of merged and separate attributes with a full multiple viewpoint system including viewpoint selection on the primary domain of the current research, jazz chord sequences. A full multiple viewpoint system requires a method for combining viewpoints (§6.1) and a way of selecting a set of viewpoints which returns the lowest mean information content from the large number of possible sets (§6.2).

This experiment uses all of the basic and derived harmonic viewpoints defined in §3.1 and §3.3 to predict the two basic attributes `Root` and `ChordType`; in other words, the chord symbol itself. `PosInBar` is not predicted in the current research as it is not clear whether temporal structure follows the same Markovian properties as other dimensions of music (e.g. pitch) in music perception. However, it is likely that temporal structure does play a role in the statistical properties, so `PosInBar` can be linked with other viewpoints to make predictions. As `PosInBar` is not being predicted its value is assumed to be known at the point of a symbol prediction, as such it is considered to be a *given* attribute. Therefore, linked viewpoints containing `PosInBar` are constrained such that they match the `PosInBar` attribute for the predicted event.

A viewpoint pool is required which is large enough to perform well, but small enough to be computationally practical. For the current study, up to three viewpoints may be linked for a linked viewpoint, but with the condition that a link between three viewpoints must include `PosInBar`. For a system predicting `Root` and `ChordType` separately this gives a pool of 156 viewpoints and when predicting `Root` $\otimes$ `ChordType` a pool of 64. Note that the difference in viewpoint pool size is due to the fact that every viewpoint in the second system must predict both of the merged attributes together, whilst for separate predictions a viewpoint need only predict one so long as the system as a whole predicts both.

6.1 Viewpoint Combination

In a multiple viewpoint system predictions from various viewpoint models and STMs / LTMs must be combined in order to make a prediction over the basic event space, ξ, or more specifically, the attributes of the event space which are being predicted: $\{\tau_{b_1}, ..., \tau_{b_i}\}$. For derived viewpoints, sequences of the derived symbols are converted back into the basic event space with an inverse mapping function (see Pearce et al., 2005, p. 301). The combination function must be monotonic, give a probability between the minimum and maximum probabilities, and the relative weightings for each model must be dependant on their certainty (Conklin, 1990, p. 70-71). Let M be the set of all models at a stage of viewpoint combination, $p_m(t)$ be the probability assigned to the basic attribute $t \in [\tau_b]$ by model $m \in M$, w_m be the weight given to the model m, and R a normalisation constant to ensure the entire distribution over $[\tau_b]$ sums to one. Pearce et al. (2005) showed a weighted geometric combination function (Equation 8) to be optimal for both combining viewpoint predictions and STM and LTM predictions, therefore, this is upheld for the current research.

$$p\left(t\right) = \frac{1}{R}\left(\prod_{m \in M} p_m\left(t\right)^{w_m}\right)^{\dfrac{1}{\sum_{m \in M} w_m}} \tag{8}$$

The weights, w_m, are determined by the relative entropy (Equation 9), calculated using the *Shannon entropy* (Equation 10) and maximum entropy (Equation 11) of the distributions over $[\tau]$ arising from p_m. A maximally certain distribution (i.e. when one outcome has a probability of one and the rest zero) will have $H(p_m) = 0$. A maximally uncertain (i.e. uniform) distribution will have an entropy determined by its alphabet size. The weight for a model is then given by Equation 12 where $b \in \mathbb{Z}+$ is a bias parameter giving an exponential bias towards models with lower relative entropy as it decreases. The bias parameter must be found experimentally for each dataset and domain, however, a useful starting point is $b = 7$ for LTM-STM combination and $b = 2$ for viewpoint combination (Pearce et al., 2005). Following the precedent set by previous research (Conklin & Witten, 1995; Pearce, 2005), for each basic attribute[6] to be predicted viewpoint predictions are combined first, followed by LTM-STM combination.

$$H_{relative}(p_m) = \begin{cases} \frac{H(p_m)}{H_{max}(p_m)} & \text{if } H_{max}(p_m) > 0 \\ 1 & \text{otherwise} \end{cases} \tag{9}$$

$$H(p_m) = -\sum_{t \in [\tau_b]} p_m(t) \log_2 p_m(t) \tag{10}$$

$$H_{max}(p_m) = \log_2 |[\tau]| \tag{11}$$

$$w_m = H_{relative}(p_m)^{-b} \tag{12}$$

6.2 Viewpoint Selection

The space of possible multiple viewpoint models is too large to search exhaustively, since it is the power set of the viewpoint pool. The size of the viewpoint pool for merged attribute predictions is 64, giving $2^{64} = 1.8 \times 10^{19}$ possible viewpoint models, whilst for separate attribute predictions the number of possible viewpoint models is $2^{156} = 9.1 \times 10^{64}$. Pearce (2005, pp. 122) proposes a viewpoint selection algorithm which follows the principle of Ockham's razor in finding an acceptable, simple solution in the search space. The algorithm is referred to as *forward stepwise selection*, using $\bar{h}$ as a heuristic. Starting initially from the empty set, each iteration consists of two stages. During the deletion stage each viewpoint in the model is removed in turn, with the deletion that yields the largest drop in $\bar{h}$ selected for the next iteration. If no deletions improve the heuristic, an addition stage adds each viewpoint in the viewpoint pool to the model in turn, with the addition that gives the greatest improvement in performance selected for the next iteration.

Ordinarily, the algorithm terminates when no additions or deletions improve performance. However, Whorley (2013, pp. 189-191) notes that viewpoint selection typically results in a long tail, where the later iterations yield only small improvements in performance at the cost of time

[6]Note that a merged attribute is treated as a single basic attribute.

and memory complexity. Whorley (2013) curtails viewpoint selection *post hoc*; a large number of viewpoint models are processed only to be discarded. For practical purposes, a method for curtailing viewpoint selection at run time is proposed. Let $\bar{h}_i$ be the mean information content of the current iteration and $\bar{h}_{i+1}$ the mean information content of the proposed next iteration. Performance improvement is measured in terms of effect size by Cohen's $d = \frac{\bar{h'}_i - \bar{h'}_{i+1}}{\sigma_{pooled}}$ over events, with the algorithm terminating if $d < 0.005$. This is the equivalent to an improvement of less than 0.5% of a standard deviation over the population of all events.

6.3 Experimental Design

This experiment is carried out on the corpus of jazz lead sheets from the Real Book (dataset 1), predicting the attributes `Root` and `ChordType`. To build a full viewpoint system, viewpoint selection is undertaken first using bias weights of $b = 7$ and $b = 2$ (established by Pearce et al., 2005) for LTM-STM and viewpoint combination respectively. Afterwards, the best bias values are found for each multiple viewpoint model with an exhaustive search where $b \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 16, 32\}$. As in previous experiments, $\bar{h}$ is calculated with a 10-fold cross validation of the dataset. Throughout the experiment the optimum smoothing parameters found in Experiments 1 (§4) and 2 (§5) are retained. A model predicting separate attributes will, therefore, use STMD*IU and LTM+C*I, whilst a model predicting merged attributes will use STMC*IUM and LTM+C*IM.

6.4 Hypothesis

The hypothesis tested is that when using the full viewpoint system the prediction of merged attributes will outperform the prediction of separate attributes, as `Root` and `ChordType` are highly correlated in this corpus. Since the initial estimated bias weights are optimal for similar data (Pearce et al., 2005), it is likely that this difference in performance will be observed both after viewpoint selection alone and after bias optimization.

6.5 Results

The viewpoint selection results for predicting separate and merged attributes are described in Figure 2. Both follow strikingly similar patterns, with five addition steps before curtailed termination on the sixth iteration. For this initial result, predicting merged attributes ($\bar{h} = 3.037$) outperformed separate attributes ($\bar{h} = 3.425$) by 0.389 bits/event. A paired t-test over all pieces proved this to be statistically significant ($df = 347, t = 20.287, p < 0.001$) with an effect size of $d = 0.320$. However, these results are calculated using preliminary bias parameters from melodic datasets (Pearce et al., 2005). In order to make a proper comparison between the two methods, the bias parameters must be optimal for the harmonic corpus at hand.

Model performance across all bias parameters for separate prediction is shown on Table 7 and for merged prediction on Table 8. A lowest $\bar{h}$ of 3.393 for separate attribute prediction was found with the LTM-STM+ and viewpoint biases both set to 2. The merged attribute prediction improved further with a lowest $\bar{h}$ of 2.963 with the LTM-STM+ and viewpoint biases set to 2 and 1 respectively, reinforcing it as the best prediction method. An improvement in performance of 0.430 bits/event, with an effect size of $d = 0.378$, was again found to be statistically significant ($df = 347, t = 35.586, p < 0.001$).

6.6 Conclusions and Discussion

The main result of this experiment is that the prediction of merged attributes outperforms predicting the same attributes separately in a dataset where those attributes are highly correlated. The difference in performance of 0.430 bits/event is substantial, equivalent to 37.8.4% ($d = 0.378$) of the pooled standard deviations of the population of events and warrants consideration in future research using multiple viewpoint models. Interestingly, this is greater than both of the improvements in performance for the STM and LTM+ found in Experiment 2. It seems that in this instance the improvements over separate prediction seen in the individual models are exaggerated by the extra components of the full multiple viewpoint system, rather than diminished.

Both methods resulted in similar multiple viewpoint models through viewpoint selection, presumably because PPM components of both merged and separate methods are identical. `Root`, `RootInt`, and `RootIntFiP` were present in both viewpoint selections and in both cases were selected in that order, giving a strong indication of importance (since the algorithm greedily selects viewpoints). In contrast to viewpoint selection of melodic data (Pearce, 2005, pp. 127-128), the root interval viewpoints (`RootInt, RootIntFiP`) are poorer predictors than `Root` itself. Pearce (2005) found that `cpint` was an important predictor of `cpitch` as it generalised statistical structure by allowing transpositionally equivalent sequences to be considered the same. The fact that this does not appear to translate to the harmonic domain could be attributed to a mixture of three factors. Firstly, the alphabet size of `Root` is 13, considerably smaller than the typical alphabet size of `cpitch` (21 to 37 for the datasets in the current study). Large alphabets have a problem of sparsity, partially solved by generalising with interval viewpoints, however, this need not be a problem for the small alphabet of `Root` symbols. Secondly, given that `Root` models are not particularly sparse, it may be the case that most of the common harmonic progressions occur in almost (or all) transpositions in the dataset. As this happens, there will be enough statistical structure in the model for any given transposition without having to revert to a derived viewpoint such as `RootInt` to describe it successfully. Finally, the occurrence of the special *NC* root symbol slightly damages the predictive power of the `RootInt` viewpoint. Since it does not have a proper pitch class value, `RootInt` must divide the prediction probability over the whole alphabet (Pearce, 2005, p. 115). This final effect is unlikely to be particularly large; for dataset 1 only 0.863% of chords (131 of 15,197) are *NC*.

None of the viewpoints derived from `ChordType`, namely `MajType`, `7Type`, and `FunctionType`, were selected for either model. These viewpoints simplified sequences by categorising `ChordType` into smaller categories. Viewpoints such as these are only successful if the information gained by generalising sequences of sparse data outweighs the information lost when converting from the derived sequences back to the event space, ξ. It appears this was not the case for the current study, suggesting that `ChordType` is not particularly sparsely distributed in the jazz dataset.

The appearance of `PosInBar` twice in each model suggests that temporal structure is correlated to a degree with harmonic structure for the corpus. This implies that certain harmonic functions are more likely to occur on different beats of the bar, for example, a tonic may be likely to occur on the first beat of a bar.

The bias combination results contrast strikingly with those established for melodic corpora. Notably, Pearce et al. (2005) establishes a high LTM-STM bias of 7 to predict `cpitch`, whilst the current study finds an LTM-STM bias of 2 predicts `Root` and `ChordType` best (both merged and separately). The suggestion that the high LTM-STM bias helps to weight away from cases where the STM produces high entropy predictions (because of a lack of context) does not appear to apply to the current domain and dataset. It is possible that the STM in general performs well for the jazz dataset as there is often a large amount of repetition within a lead sheet.

7 Summary and Discussion

Contrasting methods for the prediction of multiple attributes of a musical surface have been presented and tested with multiple viewpoint systems across three experiments. One method predicts basic attributes separately, whilst another forms a merged representation so that simultaneous predictions can be made. As hypothesised, it was found that when the basic attributes are highly correlated in a corpus they are better predicted by the merged method, whereas if they are relatively uncorrelated a separate model is best (§5). With a full multiple viewpoint system predicting the primary domain of the study, jazz chord sequences, the merged method statistically significantly outperformed predicting separate attributes by 0.430 bits/event (§6).

A secondary contribution of this work was to explore the optimal smoothing techniques for melodic and harmonic domains, predicting separate and merged attributes. Experiment 1 (§4) reinforced that techniques found to be effective for monophonic melodic prediction (Pearce & Wiggins, 2004) performed well when predicting chord sequences. Interpolated smoothing statistically significantly outperformed backoff smoothing. The use of update exclusion was in general found to damage model performance. One notable difference was that escape method D performed well for certain harmonic attribute combinations involving temporal information (`PosInBar`). Escape method C predicted the melodic datasets and chord symbols (consisting of `Root` and `ChordType`) effectively, confirming the results established by Pearce and Wiggins (2004). These results held for the prediction of merged attributes in Experiment 2 (§6) with only minor discrepancies. A useful avenue for future research might be to investigate whether the optimal smoothing techniques for a dataset can be ascertained from it's qualities. The chief predictors might be the alphabet size, a measure of its distribution (Shannon Entropy), and the rate at which new symbols are seen. The problem could be solved as a supervised machine learning task, with those predictors as variables, and the optimal set of smoothing parameters found with exhaustive search.

The effectiveness of predicting merged attributes for highly correlated basic attributes hints at some interesting implications for cognitive representation. In hand constructed multiple viewpoint systems (Conklin & Witten, 1995), as well as for viewpoint selection (Pearce, 2005, pp. 127-128), it has been speculated that linked viewpoints are effective predictors when their constituent viewpoints are correlated. The current research supports this idea but goes further, suggesting that correlated attributes are merged not only at the prediction level (the linked viewpoint), but also on the surface level. How this translates onto a cognitive system is not clear, as it would be naive to directly map processes within IDyOM onto human cognition. Tentatively, it could be hypothesised that representations which are found to be correlated are merged into a single representation. Certainly, from a computational perspective this study has shown that this gives a more compact representation in terms of information theoretic properties; lower mean information contents imply closer fits between the model and training data. A merged representation contains identical absolute information about the surface compared to separate representations; no information is lost since a merged representation is simply a Cartesian product of its constituent attributes. However, the representation is more compact owing to a lower mean information content, therefore, it is estimated that less bits are required to represent each event. This gain in representational efficiency does not increase time or space complexity since the predictive part of the model (the viewpoints) are identical for both methods, only the surface representation is different.

References

Assayag, G. & Dubnov, S. (2004). Using Factor Oracles for Machine Improvisation. *Soft Computing*, *8*(9), 1–7.

Begleiter, R., El-Yaniv, R. & Yona, G. (2004). On prediction using variable order Markov models. *Journal of Artificial Intelligence Research*, *22*, 385–421.

Bejerano, G. & Yona, G. (2001). Variations on probabilistic suffix trees: statistical modeling and prediction of protein families. *Bioinformatics*, *17*(1), 23–43.

Bunton, S. (1996). *On-line Stochastic Processes in Data Compression* (Doctoral dissertation, University of Washington, Seattle, WA).

Bunton, S. (1997). Semantically motivated improvements for PPM variants. *The Computer Journal*, *40*(2, 3), 76–93.

Chen, S. F. & Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, *13*(4), 359–394.

Chew, E. (2002). The spiral array: An algorithm for determining key boundaries. In *Music and Artificial Intelligence* (pp. 18–31). Berlin, Heidelberg: Springer Berlin Heidelberg.

Cleary, J. G. & Teahan, W. J. (1995). Experiments on the zero frequency problem. In *Proceedings of Data Compression Conference 2002* (p. 480). Snowbird, UT: IEEE.

Cleary, J. G. & Teahan, W. J. (1997). Unbounded length contexts for PPM. *The Computer Journal*, *40*(2 and 3), 67–75.

Cleary, J. G. & Witten, I. (1984). Data compression using adaptive coding and partial string matching. *Communications, IEEE Transactions on*, *32*(4), 396–402.

Conklin, D. (1990). *Prediction and Entropy of Music* (Master's thesis, Department of Computer Science, University of Calgary).

Conklin, D. (2002). Representation and discovery of vertical patterns in music. In *Music and Artificial Intelligence* (pp. 32–42). Berlin, Heidelberg: Springer.

Conklin, D. (2010). Discovery of distinctive patterns in music. *Intelligent Data Analysis*, *14*(5), 547–554.

Conklin, D. & Witten, I. (1995). Multiple viewpoint systems for music prediction. *Journal of New Music Research*, *24*(1), 51–73.

Creighton, H. (1966). *Songs and Ballads from Nova Scotia*. New York, NY: Dover.

Dubnov, S., Assayag, G., Lartillot, O. & Bejerano, G. (2003). Using machine-learning methods for musical style modeling. *Computer*, *36*(10), 73–80.

Ebcïoğlu, K. (1986). An Expert System for Chorale Harmonization. In *Proceedings of the 5th National Conference on Artificial Intelligence* (pp. 784–788). Menlo Park, CA: AAAI Press.

Forte, A. (1973). *The Structure of Atonal Music*. Yale University Press.

Granroth-Wilding, M. & Steedman, M. (2014). A Robust Parser-Interpreter for Jazz Chord Sequences. *Journal of New Music Research*, *43*(4), 355–374.

Harte, C., Sandler, M. B., Abdallah, S. A. & Gómez, E. (2005). Symbolic representation of musical chords: A proposed syntax for text annotations. In *6th International Society for Music Information Retrieval Conference* (pp. 66–71). London, UK.

Howard, P. G. (1993). *The Design and Analysis of Efficient Lossless Data Compression Systems* (Doctoral dissertation, Providence, RI).

Huron, D. (2006). *Sweet Anticipation: Music and the Psychology of Expectation.* Cambridge, MA: MIT Press.

Jelinek, F. & Mercer, R. L. (1980). Interpolated estimation of Markov source parameters from sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice* (pp. 381–397). Amsterdam, The Netherlands: North-Holland.

Kneser, R. & Ney, H. (1995). Improved backing-off for m-gram language modeling. In *International Conference on Acoustics, Speech and Signal Processing* (pp. 181–184). Detroit, MI.

Krichevsky, R. E. & Trofimov, V. K. (1981). The performance of universal encoding. *Information Theory, IEEE Transactions on*, *27*(2), 199–207.

Leonard, H. (2012). *The Real Book: Volume I, II, III, IV and V.* Winoa, MN: Hal Leonard.

Lerdahl, F. & Jackendoff, R. (1983). *A Generative Theory of Tonal Music.* Cambridge, MA: MIT Press.

Levine, M. (1995). *The Jazz Theory Book.* Petaluma, CA: Sher Music Co.

Longuet-Higgins, H. C. (1979). Review Lecture: The perception of music. In *Proceedings of the Royal Society* (pp. 307–322). London, UK.

Mackay, D. (2003). *Information Theory, Inference and Learning Algorithms.* Cambridge, UK: Cambridge University Press.

Manning, C. & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing.* MIT Press.

Meeus, N. (2000). Toward a post-Schoenbergian grammar of tonal and pre-tonal harmonic progressions. *Music Theory Online*, *6*(1), 1–8.

Meyer, L. (1956). *Emotion and Meaning in Music.* London, UK: University of Chicago Press.

Moffat, A., Neal, R. M. & Witten, I. (1998). Arithmetic coding revisited. *ACM Transactions on Information Systems*, *16*(3), 256–294.

Moffat, A., Sharman, N., Witten, I. H. & Bell, T. C. (1994). An empirical evaluation of coding methods for multi-symbol alphabets. *Information Processing and Management*, *30*(6), 791–804.

Moffat, A. (1990). Implementing the PPM data compression scheme. *Communications, IEEE Transactions on*, *38*(11), 1917–1921.

Narmour, E. (1990). *The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model.* Chicago, IL: University of Chicago Press.

Pachet, F. & Roy, P. (2014). Non-conformant harmonization: the Real Book in the style of Take 6. In *The 5th International Conference on Computational Creativity.* Ljubljana, Slovenia.

Pachet, F., Suzda, J. & Martín, D. (2013). A comprehensive online database of machine-readable leadsheets for jazz standards. In *14th International Society for Music Information Retrieval Conference* (pp. 275–280). Curitiba, Brazil.

Pachet, F. (2000). Computer analysis of jazz chord sequences: Is Solar a blues? In *Readings in Music and Artificial Intelligence*. Harwood Academic Publishers.

Pachet, F. (2003). The Continuator: Musical Interaction With Style. *Journal of New Music Research*, *32*(3), 333–341.

Pachet, F. & Roy, P. (2011). Markov constraints: steerable generation of Markov sequences. *Constraints*, *16*(2), 148–172.

Paiement, J. F., Eck, D. & Bengio, S. (2005). A probabilistic model for chord progressions. In *6th International Conference of Music Information Retrieval* (pp. 312–319). London, UK.

Pearce, M. (2005). *The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Composition* (Doctoral dissertation, City University, London).

Pearce, M., Conklin, D. & Wiggins, G. A. (2005). Methods for combining statistical models of music. In *CMMR'04: Proceedings of the 2nd International Conference on Computer Music Modeling and Retrieval* (pp. 295–312). Springer-Verlag.

Pearce, M., Mullensiefen, D. & Wiggins, G. A. (2010). The role of expectation and probabilistic learning in auditory boundary perception: A model comparison. *Perception*, *39*(10), 1365–1389.

Pearce, M. & Wiggins, G. A. (2004). Improved Methods for Statistical Modelling of Monophonic Music. *Journal of New Music Research*, *33*(4), 367–385.

Perez-Sancho, C., Rizo, D. & Inesta, J. M. (2009). Genre classification using chords and stochastic language models. *Connection Science*, *21*(2), 145–159.

Piston, W. (1948). *Harmony*. New York: W. W. Norton and Company.

Ponsford, D., Wiggins, G. A. & Mellish, C. (1999). Statistical learning of harmonic movement. *Journal of New Music Research*, *28*(2), 150–177.

Rameau, J. P. (1971). *Treatise on Harmony*. New York: Dover Publications.

Riemann, H. (1895). *Harmony Simplified*. Or, the Theory of the Tonal Functions of Chords. London: Augener.

Riemenschneider, A. (1941). *371 Harmonised Chorales and 69 Chorale Melodies with Figured Bass*. New York, NY: G. Schirmer Inc.

Rissanen, J. (1983). A universal data compression system. *Information Theory, IEEE Transactions on*, *29*(5), 656–664.

Rohrmeier, M. (2011). Towards a generative syntax of tonal harmony. *Journal of Mathematics and Music*, *5*(1), 35–53.

Rohrmeier, M. & Graepel, T. (2012). Comparing feature-based models of harmony. In *Proceedings of the 9th International Symposium on Computer Music Modeling and Retrieval*. London, UK.

Ron, D., Singer, Y. & Tishby, N. (1996). The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, *25*(2-3), 117–149.

Schaffrath, H. (1995). The Essen Folksong Collection. In D. Huron (Ed.), *Database containing folksong transcriptions in the Kern format and a 34-page research guide.* Menlo Park, CA.

Schoenberg, A. (1969). *Structural Functions of Harmony* (L. Stein, Ed.). New York: W. W. Norton and Company.

Shannon, C. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, *27*(3), 379–423.

Shkarin, D. (2002). PPM: one step to practicality. In *Data Compression Conference, Proceedings* (pp. 202–211).

Steedman, M. (1984). A Generative Grammar for Jazz Chord Sequences. *Music Perception: An Interdisciplinary Journal*, *2*(1), 52–77.

Temperley, D. (2001). *The Cognition of Basic Musical Structures.* MIT Press.

Ukkonen, E. (1995). On-line construction of suffix trees. *Algorithmica*, *14*(3), 249–260.

Ulrich, J. (1977). The Analysis and synthesis of jazz by computer. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence* (pp. 865–872). San Francisco, CA: 5th International Joint Conference on Artificial Intelligence.

Volf, P. (2002). *Weighting Techniques in Data Compression: Theory and Algorithms* (Doctoral dissertation, Eindhoven, Netherlands).

Whorley, R. (2013). *The Construction and Evaluation of Statistical Models of Melody and Harmony* (Doctoral dissertation, Goldsmiths, University of London, London).

Whorley, R., Rhodes, C., Wiggins, G. A. & Pearce, M. (2013). Harmonising Melodies: Why Do We Add the Bass Line First? In *Proceedings of the 4th International Conference on Computational Creativity* (pp. 79–86). Sydney, Australia.

Willems, F. M. J., Shtarkov, Y. M. & Tjalkens, T. J. (1995). The context-tree weighting method: basic properties. *Information Theory, IEEE Transactions on*, *41*(3), 653–664.

Witten, I. & Bell, T. C. (1991). The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *Information Theory, IEEE Transactions on*, *37*(4), 1085–1094.

Ziv, J. & Lempel, A. (1978). Compression of individual sequences via variable-rate coding. *Information Theory, IEEE Transactions on*, *24*(5), 530–536.

Appendix

Algorithm 1 Pitch class set Categorisation Algorithm.

```
Require: pcset = $\{x \mid x \in \mathbb{Z}, x \geq 0, x < 11\}$
  function CATEGORISE(pcset)
      if $4 \in pcset$ then
          if $10 \in pcset$ then
              if $8 \in pcset$ then
                  return alt
              else
                  return 7
          else
              if $9 \in pcset$ then
                  return 6
              else if $8 \in pcset$ then
                  return aug
              else
                  return maj
      else if $3 \in pcset$ then
          if $10 \in pcset$ then
              if $6 \in pcset$ then
                  return halfdim
              else
                  return min7
          else
              if $6 \in pcset$ then
                  return dim
              else
                  if $8 \in pcset$ then
                      return min$\sharp$5
                  else
                      return min
      else if $\mid pcset \mid > 0$ then
          if $7 \in pcset$ then
              return sus
          else
              return special
      else
          return NC
```

Tables

ID	Description	Compositions	Events	Basic Attributes
1	Real Book Vol. 1	348	15,197	`Root` `ChordType` `PosInBar`
2	Complete Beatles	179	17,557	`Root` `ChordType` `PosInBar`
3	Bach chorales	185	9,227	`Pitch` `Duration`
4	German folksongs	566	33,087	`Pitch` `Duration`
5	Canadian folksongs	152	8,552	`Pitch` `Duration`

Table 1: Two harmonic and three melodic datasets used in the current research.

ChordType	Chord types from the *Real Book*
7^{th}	7, 9, 13, 7♯9, 7♭9♭5
$maj.$	M, M^7, M^9, $M^{7\flat 9}$, $M^{7\flat 5}$
$maj.6^{th}$	6, 6♯11, 69
$min.7^{th}$	m^7, m^9, m^{13}, $m^{7\sharp 5}$, m^{7add4}
$min.$	m, m^6, m^{69}, m^{M7}, m^{add9}
$min.\sharp 5$	$m^{\flat 6}$, $m^{\sharp 5}$
$dim.$	dim, $dim7$, $m^{\flat 5\flat 13}$
$halfdim.$	$halfdim7, m^{7\flat 5\flat 13}, m^{7(\flat 5\flat 2)}$, $m^{7\flat 5\sharp 5}$
$aug.$	$+$, $aug\sharp 4$
$alt.$	7♯5, 9♯5, 13♯9♯5, 7♯5♭5
sus	$7sus$, $sus2$, $6sus4$, $13\flat 9sus$, $Phrygian$
$special$	various unclassified slash chords e.g. $FM^7/\ E\flat$
NC	NC

Table 2: The complete alphabet of ChordType with typical corresponding chords mapped from the *Real Book* using Algorithm 1.

	BM^7	D^7	GM^7	$B\flat^7$	$E\flat M^7$	NC	Am^7
`Root`	11	2	7	10	3	-1	9
`ChordType`	maj	7	maj	7	maj	NC	m7
`PosInBar`	0	4	0	4	0	4	0
`RootInt`	$\perp$	3	5	3	5	-1	-1
`MeeusInt`	$\perp$	-1	1	-1	1	-3	-3
`ChromaDist`	$\perp$	3	1	3	1	-1	-1
`RootIntFiP`	$\perp$	3	8	11	4	-1	10
`MeeusIntFiP`	$\perp$	-1	1	-1	-1	-3	-1
`ChromaDistFiP`	$\perp$	3	4	5	4	-1	2
`RootInt` $\ominus$ `FiB`	$\perp$	$\perp$	8	$\perp$	8	$\perp$	6
`MajType`	1	1	1	1	1	0	2
`7Type`	0	1	0	1	0	-1	0
`FunctionType`	0	1	0	1	0	-1	2

Table 3: Sample chord sequence with basic and derived viewpoints. A timebase of 2 is used.

Escape Method	Prediction Probability $\alpha\left(e_i \mid e_{i-n+1}^{i-1}\right)$	Escape Probability $\gamma\left(e_{i-n+1}^{i-1}\right)$
A	$\frac{c\left(e_i \mid e_{i-n+1}^{i-1}\right)}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)+1}$	$\frac{1}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)+1}$
B	$\frac{c\left(e_i \mid e_{i-n+1}^{i-1}\right)-1}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)}$	$\frac{t\left(e_{i-n+1}^{i-1}\right)}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)}$
C	$\frac{c\left(e_i \mid e_{i-n+1}^{i-1}\right)}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)+t\left(e_{i-n+1}^{i-1}\right)}$	$\frac{t\left(e_{i-n+1}^{i-1}\right)}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)+t\left(e_{i-n+1}^{i-1}\right)}$
D	$\frac{c\left(e_i \mid e_{i-n+1}^{i-1}\right)-0.5}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)}$	$\frac{0.5\cdot t\left(e_{i-n+1}^{i-1}\right)}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)}$
AX	$\frac{c\left(e_i \mid e_{i-n+1}^{i-1}\right)}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)+t_1\left(e_{i-n+1}^{i-1}\right)+1}$	$\frac{t_1\left(e_{i-n+1}^{i-1}\right)+1}{\sum_{e\in[\tau]} c\left(e \mid e_{i-n+1}^{i-1}\right)+t_1\left(e_{i-n+1}^{i-1}\right)+1}$

Table 4: Prediction and escape probabilities of five escape methods empirically tested by Pearce and Wiggins (2004).

	Dataset	Viewpoint	$\lvert[\tau_a] \times [\tau_b]\rvert$	Backoff smoothing		Interpolated smoothing		No Update Exclusion		Update Exclusion	
				Model	$\bar{h}$	Model	$\bar{h}$	Model	$\bar{h}$	Model	$\bar{h}$
STM	1	`Root`⊗`ChordType`	145	STMD*U	4.372	**STMD*IU**	**4.337**	STMC*I	4.351	**STMD*IU**	**4.337**
	1	`Root`⊗`PosInBar`	143	STMC*	3.348	**STMC*I**	**3.312**	**STMC*I**	**3.312**	STMD*IU	3.352
	1	`ChordType`⊗`PosInBar`	143	STMA*U	2.657	**STMA*IU**	**2.625**	STMA*I	2.644	**STMA*IU**	**2.625**
	2	`Root`⊗`ChordType`	121	STMA*U	2.350	**STMD*I**	**2.263**	**STMD*I**	**2.263**	STMA*IU	2.292
	2	`Root`⊗`PosInBar`	195	STMA*U	2.013	**STMD*I**	**1.934**	**STMD*I**	**1.934**	STMA*IU	1.960
	2	`ChordType`⊗`PosInBar`	165	STMA*	1.691	**STMA*I**	**1.561**	**STMA*I**	**1.561**	STMA*IU	1.695
	3	`Pitch`⊗`Duration`	294	STMA*U	4.677	**STMX*I**	**4.668**	**STMX*I**	**4.668**	STMA*U	4.677
	4	`Pitch`⊗`Duration`	629	**STMA*U**	**4.897**	STMA*IU	4.898	STMX*I	4.950	**STMA*U**	**4.897**
	5	`Pitch`⊗`Duration`	364	**STMA*U**	**4.673**	STMA*IU	4.685	STMX*I	4.702	**STMA*U**	**4.673**
LTM+	1	`Root`⊗`ChordType`	145	LTM+C*	4.147	**LTM+C*I**	**4.073**	**LTM+C*I**	**4.073**	LTM+D*IU	4.266
	1	`Root`⊗`PosInBar`	143	LTM+C*	3.045	**LTM+D*I**	**2.985**	**LTM+D*I**	**2.985**	LTM+D*IU	3.156
	1	`ChordType`⊗`PosInBar`	143	LTM+C*	2.519	**LTM+D*I**	**2.427**	**LTM+D*I**	**2.427**	LTM+A*IU	2.714
	2	`Root`⊗`ChordType`	121	LTM+C*	2.694	**LTM+C*I**	**2.627**	**LTM+C*I**	**2.627**	LTM+A*IU	2.705
	2	`Root`⊗`PosInBar`	195	LTM+C*	2.298	**LTM+C*I**	**2.252**	**LTM+C*I**	**2.252**	LTM+A*IU	2.298
	2	`ChordType`⊗`PosInBar`	165	LTM+C*	1.756	**LTM+D*I**	**1.695**	**LTM+D*I**	**1.695**	LTM+A*IU	1.966
	3	`Pitch`⊗`Duration`	294	LTM+C*	3.628	**LTM+C*I**	**3.541**	**LTM+C*I**	**3.541**	LTM+D*IU	3.654
	4	`Pitch`⊗`Duration`	629	LTM+C*	4.305	**LTM+C*I**	**4.254**	**LTM+C*I**	**4.254**	LTM+D*IU	4.299
	5	`Pitch`⊗`Duration`	364	LTM+C*	4.350	**LTM+C*I**	**4.296**	**LTM+C*I**	**4.296**	LTM+D*IU	4.350

Table 5: Mean information content ($\bar{h}$) for the optimal backoff smoothing, interpolated smoothing, with update exclusion, and without update exclusion STM/LTM+ models. The best model for each row is underlined and in boldface.

	Dataset	Viewpoint	$\lvert[\tau_a] \times [\tau_b]\rvert$	Separate		Merged		Correlation (Φ_c)	Difference (d)
				Model	$\bar{h}_1$	Model	$\bar{h}_2$		
STM	1	`Root`⊗`ChordType`	145	STMD*IU	4.337	**STMC*IUM**	**4.089**	0.325	0.176*
	1	`Root`⊗`PosInBar`	143	**STMC*I**	**3.312**	STMC*IUM	3.502	0.069	-0.235*
	1	`ChordType`⊗`PosInBar`	143	**STMA*IU**	**2.625**	STMC*IM	2.702	0.208	-0.104*
	2	`Root`⊗`ChordType`	121	STMD*I	2.263	**STMC*IM**	**1.988**	0.359	0.407*
	2	`Root`⊗`PosInBar`	195	STMD*I	1.934	**STMC*IM**	**1.928**	0.078	-0.006
	2	`ChordType`⊗`PosInBar`	165	STMA*I	1.561	**STMD*IM**	**1.529**	0.130	0.040
	3	`Pitch`⊗`Duration`	294	**STMX*I**	**4.668**	STMX*IUM	5.348	0.075	-0.904*
	4	`Pitch`⊗`Duration`	629	**STMA*U**	**4.897**	STMX*IUM	5.678	0.051	-0.736*
	5	`Pitch`⊗`Duration`	364	**STMA*U**	**4.673**	STMX*IUM	5.352	0.074	-0.822*
LTM+	1	`Root`⊗`ChordType`	145	LTM+C*I	4.073	**LTM+C*IM**	**3.678**	0.325	0.313*
	1	`Root`⊗`PosInBar`	143	LTM+D*I	2.985	**LTM+C*IM**	**2.940**	0.069	0.067*
	1	`ChordType`⊗`PosInBar`	143	LTM+D*I	2.427	**LTM+D*IM**	**2.361**	0.208	0.060*
	2	`Root`⊗`ChordType`	121	LTM+C*I	2.627	**LTM+C*IM**	**2.312**	0.359	0.409*
	2	`Root`⊗`PosInBar`	195	LTM+D*I	2.252	**LTM+C*IM**	**2.190**	0.078	0.170*
	2	`ChordType`⊗`PosInBar`	165	LTM+D*I	1.695	**LTM+D*IM**	**1.657**	0.130	0.049*
	3	`Pitch`⊗`Duration`	294	**LTM+C*I**	**3.541**	LTM+C*IM	3.647	0.075	-0.121*
	4	`Pitch`⊗`Duration`	629	**LTM+C*I**	**4.254**	LTM+C*IM	4.402	0.051	-0.152*
	5	`Pitch`⊗`Duration`	364	**LTM+C*I**	**4.296**	LTM+C*IM	4.487	0.074	-0.172*

Table 6: Mean information content ($\bar{h}$) for optimal models predicting basic and merged attributes. Correlation is measured by Cramer's V $\Phi_c = \sqrt{\frac{\chi^2}{n \cdot df}}$. Performance difference is measured by Cohen's $d = \frac{\bar{h_1} - \bar{h_2}}{\sigma_{pooled}}$. Statistically significant differences in performance measured with two-sided paired t-tests over all pieces at the $p < 0.001$ level are marked with *.

		Viewpoint bias										
		0	1	2	3	4	5	6	7	8	16	32
LTM-STM+ bias	0	3.551	3.492	3.456	3.436	3.426	3.421	3.419	3.418	3.419	3.432	3.450
	1	3.482	3.426	**3.402**	**3.394**	**3.394**	**3.397**	**3.400**	3.404	3.408	3.428	3.446
	2	3.450	3.404	$\boxed{\mathbf{3.393}}$	**3.398**	3.409	3.420	3.429	3.437	3.444	3.474	3.493
	3	3.436	**3.398**	**3.396**	3.411	3.429	3.445	3.459	3.470	3.480	3.519	3.541
	4	3.431	**3.398**	3.403	3.424	3.446	3.467	3.484	3.497	3.509	3.556	3.581
	5	3.430	**3.402**	3.411	3.435	3.461	3.484	3.503	3.519	3.531	3.584	3.613
	6	3.432	3.406	3.418	3.445	3.473	3.498	3.518	3.535	3.549	3.607	3.638
	7	3.435	3.411	3.425	3.454	3.484	3.509	3.531	3.548	3.563	3.625	3.658
	8	3.438	3.416	3.432	3.462	3.492	3.519	3.541	3.559	3.574	3.639	3.675
	16	3.459	3.442	3.463	3.496	3.531	3.560	3.584	3.604	3.621	3.697	3.741
	32	3.476	3.461	3.485	3.521	3.558	3.589	3.613	3.634	3.652	3.734	3.780

Table 7: Mean information content, $\bar{h}$, (bits/event) of a multiple viewpoint model predicting `Root` and `ChordType` separately with `Root`⊗`ChordType`⊗`PosInBar`, `RootInt`⊗`ChordType`⊗`PosInBar`, `RootIntFiP`⊗`ChordType`, `ChordType`⊗`PosInBar`, and `RootInt`⊗`RootIntFiP`⊗`PosInBar` using a range of model combination bias parameters. The lowest ten $\bar{h}$ values are in bold and the lowest boxed.

		Viewpoint bias										
		0	1	2	3	4	5	6	7	8	16	32
LTM-STM+ bias	0	3.084	3.050	3.034	3.029	3.032	3.037	3.043	3.050	3.056	3.090	3.122
	1	3.000	**2.973**	**2.969**	**2.978**	2.992	3.007	3.020	3.032	3.042	3.087	3.119
	2	**2.980**	$\boxed{\mathbf{2.963}}$	**2.971**	2.993	3.018	3.042	3.062	3.080	3.094	3.153	3.188
	3	**2.980**	**2.970**	**2.986**	3.015	3.047	3.077	3.102	3.124	3.142	3.213	3.252
	4	2.988	**2.981**	3.001	3.035	3.071	3.105	3.134	3.158	3.178	3.259	3.303
	5	2.996	2.992	3.015	3.052	3.091	3.126	3.157	3.184	3.206	3.294	3.341
	6	3.005	3.002	3.027	3.065	3.106	3.144	3.176	3.204	3.227	3.321	3.371
	7	3.013	3.011	3.037	3.077	3.119	3.157	3.191	3.219	3.244	3.342	3.394
	8	3.020	3.019	3.045	3.086	3.129	3.168	3.203	3.232	3.257	3.359	3.413
	16	3.054	3.054	3.082	3.127	3.173	3.215	3.252	3.284	3.313	3.429	3.491
	32	3.078	3.078	3.107	3.154	3.201	3.245	3.283	3.316	3.347	3.471	3.537

Table 8: Mean information content, $\bar{h}$, (bits/event) of a multiple viewpoint model predicting `Root⊗ChordType` with `Root⊗ChordType⊗PosInBar`, `RootInt⊗ChordType`, `RootIntFiP⊗ChordType⊗PosInBar`, `Root⊗ChordType`, `RootInt⊗ChordType⊗PosInBar` using a range of model combination bias parameters. The lowest ten $\bar{h}$ values are in bold and the lowest boxed.

Figure Captions

Figure 1: Relationship between correlation of basic attributes, and relative performance of merged vs. separate attribute predictions. Merged attribute prediction outperforms separate when $d > 0$.

Viewpoint selection predicting `Root` and `ChordType` separately (dotted line):
1 + `Root`⊗`ChordType`⊗`PosInBar`
2 + `RootInt`⊗`ChordType`⊗`PosInBar`
3 + `RootIntFiP`⊗`ChordType`
4 + `ChordType`⊗`PosInBar`
5 + `RootInt`⊗`RootIntFiP`⊗`PosInBar`

Viewpoint selection predicting `Root`⊗`ChordType` as a merged attribute (dashed line):
1 + `Root`⊗`ChordType`⊗`PosInBar`
2 + `RootInt`⊗`ChordType`
3 + `RootIntFiP`⊗`ChordType`⊗`PosInBar`
4 + `Root`⊗`ChordType`
5 + `RootInt`⊗`ChordType`⊗`PosInBar`

Figure 2: Viewpoint selection for multiple viewpoint model systems predicting `Root` and `ChordType` as separate and merged attributes. Viewpoints added at each iteration are shown below the graph.