Training Music Sequence Recognizers with Linear Dynamic Programming

Christopher Raphael craphael@indiana.edu
School of Informatics, Indiana University, Bloomington, IN 47408 USA

Eric Nichols epnichol@indiana.edu
Center for Research on Concepts and Cognition, Indiana University, Bloomington, IN 47408 USA

Keywords: machine learning, dynamic programming, harmonic analysis, trellis graph

Abstract

An obvious approach to the problem of harmonic analysis is to represent a musical score as a trellis graph, where layers in the graph correspond to musical slices of time, and each node corresponds to the harmonic analysis of the slice. If the transitions between nodes and the analyses at each node are assigned a score, the analysis problem reduces to finding the best-scoring path through the graph, which can be computed with standard dynamic programming (DP). But how can we set the parameters of the necessary score function? We demonstrate a novel algorithm, Linear Dynamic Programming (LDP), to solve the problem of *learning* the coefficients for a linear score function, based on a hand-labeled harmonic analysis.

1. Introduction

Many music recognition problems seek to explain some aspect of music data as a *sequence* of labels that collectively represent a time-varying analysis. Familiar examples of this view come from problems that deal with both symbolic music representations (e.g. MIDI) as well as audio. For instance, harmonic analysis of symbolic music data produces a label including chord, key, or other harmonic attributes, for each measure or beat of the piece under consideration (Temperley, 2001). Other examples of symbolic music sequence analysis include pitch spelling, instrument fingering, prosodic labeling of melody for musical expression, and decomposition of polyphonic music into voices. In the audio domain, the problem of beat tracking can be expressed as a labeling of each audio frame as either a "beat" or "non-beat." Similar views are often taken of other audio music processing examples such as the signal-to-score problem, computing harmony from audio, chorus detection, score alignment, etc. In fact, the sequence analysis view extends to other music data, such as optical music recognition.

The sequence estimation problem can be solved by expressing each possible label sequence as a path through a state graph. Arc costs in the graph correspond to *a priori* path plausibility, while data scores for nodes in the graph give the agreement between a label and an observation. Given this model, it is simple to compute the best-scoring path using DP. However, an oft-ignored problem faced in applying this methodology is the specification of arc and data scores — the training problem. Hand setting of parameters through trial and error is usually only feasible in small problems. Our research has produced a novel method for the *supervised* training of a DP-based sequence estimator we call *Linear Dynamic Programming*. The method assumes that the arc and data scores are known linear functions of some unknown parameter vector. In this context we seek the parameter value leading to the best performance on our labeled training set. An example is presented for harmonic analysis, though we believe the technique may be broadly applicable.

2. Application to Harmonic Analysis

The problem of *functional harmonic analysis* seeks to partition a piece of music into labeled sections, the labels giving the local *harmonic state*. The label usually consists of a key and a chord symbol such as I, ii, iii, IV, V, vi, vii. For instance, the label (A major, IV) corresponds to the triad (D,F♯,A) built on the fourth scale degree in the key of A major. Several efforts have addressed this problem of automatic harmonic analysis, including (Pardo & Birmingham, 2002), (Raphael

& Stoddard, 2003), and (Temperley, 2001).

We seek to label each measure (or beat) of music with a collection of predefined harmonic labels, appropriately chosen for the music in question. Our recognition approach uses DP to find the best scoring path through the lattice composed by an $S \times N$ array of states, where N is the number of measures or beats in the piece and S is the number of possible harmonic labels we consider. Our cost function consists of two components: a data cost and a path cost. The data cost encourages close agreement between each measure label and the pitches of that measure. The path cost rewards paths that are more musically plausible, independent of the data. Our recognized sequence is then computed as the lattice path that minimizes the sum of these costs.

More explicitly, our cost function, $C_\theta(s)$, is composed as $C_\theta(s) = D_\theta(s) + P_\theta(s)$ where the path, s, is a sequence of labels, $s = s_1, \ldots, s_n$, one for each measure. The data score, $D_\theta(s) = \sum_{n=1}^{N} d_\theta(s_n, x_n)$, is represented as

$$d_\theta(s_n, x_n) = \sum_{i=1}^{3} \sum_{j=1}^{4} \theta_{ij}^d \delta_{ij}(s_n, x_n) \qquad (1)$$

where x_n is the collection of pitches in the nth measure and the counts, $\delta_{ij}(s_n, x_n)$, are as follows. Each harmonic label, s_n, divides the possible chromatic pitches into four categories: those that are 1) the root of the chord, 2) in the chord but not the root, 3) in the scale but not the chord, and 4) outside the scale. Similarly, the pitches in a measure are divided into those that begin 1) on the downbeat of the measure, 2) on a beat but not the downbeat, and 3) elsewhere in the measure. In Eqn. 1, $\delta_{ij}(s_n, x_n)$ counts the notes in the nth measure that are in position category i and of chromatic type j. To avoid degeneracies in which families of parameter assignments correspond to essentially identical choices, we further assume $\sum_j \theta_{ij}^d = 0$.

The path score $P_\theta(s) = \sum_{n=1}^{N-1} p_\theta(s_n, s_{n+1})$ is the sum of modulation and progression components:

$$p_\theta(s_n, s_{n+1}) = \theta^m M(s_n, s_{n+1}) + \sum_{i=1}^{3} \theta_i^h H_i(s_n, s_{n+1})$$

where $M(s_n, s_{n+1})$ is an indicator function for key change. If $M(s_n, s_{n+1}) = 1$, the $\{H_i\}$ terms act as indicators for various classes of harmonic motion such as progressive and regressive. As above, we assume $\sum_i \theta_i^h = 0$. In total, considering the linear constraints, our parameter θ has dimension 12.

The LDP algorithm divides parameter space into regions corresponding to distinct paths through the trellis. The assumption of linear path scores, along with the trellis structure, render this problem tractable — LDP uses DP itself to partition parameter space. Here, LDP yields a collection of distinct harmonic analyses along with regions in parameter space that produce each analysis. LDP chooses the analysis closest to the training data according to a harmonically motivated path quality measure. Then we select a parameter θ from the center of the associated region.

Using this procedure, we trained our algorithm on the *Grande Valse Brilliante* of Chopin using hand-labeled ground truth. LDP improved our path quality measure from 2203 to 173, starting with a random initial configuration for θ. The resulting configuration corresponded to a total summed symmetric difference (SSD) of 73 between the recognized chord pitch classes and the ground truth chord pitch classes, as well as an SSD of 100 between the two scale sequences. This corresponds to 0.24 chord errors and 0.33 scale errors per measure. We then applied this learned parameter to the Chopin "Minute Waltz", resulting in an SSD of 186 chord pitch class errors (1.33/measure) and 40 scale pitch class errors (0.29/measure). The analysis is available for download as a midi file at www.music.informatics.indiana.edu/papers/aaai08/.

Research is ongoing to improve the LDP algorithm and to test it on a larger collection of musical examples. Although our work to date has focused on harmonic analysis, LDP is applicable to other musical domains; indeed, the algorithm was inspired by difficulties encountered in choosing parameters for a piano fingering task (Kasimi et al., 2007). We hope that LDP will prove useful in many other domains where training data exists in the form of a known sequential labeling.

References

Kasimi, A., Nichols, E., & Raphael, C. (2007). A simple algorithm for automatic generation of polyphonic piano fingerings. *Proc. ISMIR*.

Pardo, B., & Birmingham, W. P. (2002). Algorithms for chordal analysis. *Computer Music Journal*, *26*, 27–49.

Raphael, C., & Stoddard, J. (2003). Harmonic analysis with probabilistic graphical models. *Proc. ISMIR*.

Temperley, D. (2001). *The cognition of basic musical structures*. Cambridge, MA: MIT Press.