# Quantifying the Contribution of the Jazz Grammar over a Pure Statistical Baseline

Mark Granroth-Wilding

5th April 2011

This is a brief summary of the experiments that I have run recently. They evaluate the success of different methods of recovering a tonal space interpretation of a chord sequence against a gold standard from my corpus.

## Contents

**1 Corpus** **1**

**2 Supertagging Model** **1**

**3 Grammarless Model** **2**

**4 Supertagging Model with Backoff** **2**

**5 Evaluation** **3**

**6 Results** **3**

## 1 Corpus

The corpus now contains 76 fully annotated chord sequences. The annotations assign a lexical entry from the latest version of the grammar (3.0) to each chord. They also provide the information about coordination structure necessary to build a full canonical derivation tree for each sequence and hence extract a gold standard tonal space path, with a chord function for each point.

Since the corpus is so small, all experiments are performed using 10-fold cross validation.

## 2 Supertagging Model

I am using an n-gram model of the chords of a sequence conditioned on the lexical grammatical category as a supertagging model. This assigns at least one category to each chord. A CKY parser attempts to parse these categories. Until an interpretation of the full sequence is found, the supertagger adds more lexical categories to the chart, getting progressively less probable.

Even with only a fairly small number of lexical categories, the chart can get full enough to make parsing infeasible. Improbable categories are pruned from populous edges of the chart, using the product of the probabilities of the lexical categories as an estimate of their relative probabilities.

If after a fixed length of time no result is found, the parser gives up. I am currently letting it run for $1\frac{1}{2}$ hours.

The n-gram model preprocesses the chord sequence to get its observations. It is generally accepted that perception of music depends much more on the relative pitch of notes (or chords) than on their absolute pitch. Instead of using the chord roots themselves, I treat the intervals between successive chords as observations. I add to this the chord type.

Evaluating on accuracy of category assignment along, I found little improvement from using a trigram model over a bigram model. For now I stick to using a bigram model, with Witten-Bell smoothing, backing off to a unigram model for unseen bigrams. I treat observations with counts less than 2 as unseen.

# 3 Grammarless Model

I use an n-gram model very similar to that used as a supertagging model to predict a tonal space path directly. Observations are constructed in the same way as for the supertagger and the same model architecture is used.

The model must predict a point in the infinite tonal space for each chord (potentially the same point as the previous or the next). State labels are made up of two vectors $(x, y)(X, Y)$. To turn the problem of choosing a point into the problem of recognising substitutions/inversions, we first consider the point nearest to the previous point on the path that is an instance of the played chord root. Our first vector, then, is the actual point, modulo enharmonic equivalence, relative to this initial guess. This will often be $(0, 0)$, but in the case of a tritone substitution, for example, it will be $(2, 1)$.

The second vector $(X, Y)$ tells us how many enharmonic blocks this point must be shifted by to get the actual point in the space. This will often be $(0, 0)$, since it is most common for a point to be the nearest instance to its predecessor. In some cases it is not.

The n-gram model is used to predict a state label for each chord in the sequence and these labels can then be transformed into a list of coordinates. This is then in the same form as the semantics of a parse result.

# 4 Supertagging Model with Backoff

The grammarless model will provide a state sequence, and hence a tonal space path, for any given chord sequence. The supertagger/parser model, however, will simply fail to produce an interpretation for some sequences, thanks to pruning either in the supertagger or in the chart. When it does produce an analysis, one would expect it to be of a higher quality than the pure n-gram model, since it is restricted to conform to the hand-written structures in the lexicon.

A simple extension to the supertagging model is to use the grammarless model as backoff. This model will first use the n-gram supertagger and attempt

to find a parse. If none is found, it simply uses the pure statistical model to get an analysis.

The two models are built in very similar ways and the abstraction to states described above for the grammarless model has the effect of condensing the training data into a limited number of states which are quite similar to those abstractions represented by the lexical entries. It is not unreasonable to think of the parsing model as filtering the predictions of an n-gram model to allow only those that abide by the structures in the lexicon. Backing off in this way, then, is rather like dropping the structural restrictions of the grammar and letting the n-gram model predict whatever path it wants.

# 5 Evaluation

Each model produces a tonal space path as a list of tonal space points. If more than one result is available, I look only at that which the model considers most probable. A gold standard path in the same form is also available for each sequence.

Before comparing the paths, I first transform them from a list of points to a list of intervals (vectors) between the points. This means (a) that the absolute pitch at which the path starts is ignored and (b) that if a path contains a mistake that puts it on an enharmonic equivalent of the correct point, only this mistake is penalised, not the whole of the rest of the path.

I then align the two paths optimally and evaluate how many of these intervals are aligned correctly, assigning a half point to alignments where the interval vector is correct but not the function (tonic, dominant, subdominant) or vice versa.

I compute an f-score for each alignment. Precision is defined as the proportion of the predicted points that align with a point in the gold standard path. Recall is the proportion of points in the gold standard path that are aligned with a point in the predicted path. F-score is the harmonic mean of these two metrics.

# 6 Results

| **Model** | **Precision (%)** | **Recall (%)** | **F-score (%)** |
|---|---|---|---|
| Supertagger/parser | 89.9 | 61.9 | 73.3 |
| Grammarless n-gram | 74.6 | 82.1 | 78.2 |
| Parser with backoff | 81.7 | 88.0 | 84.7 |

These results show nothing surprising at all, but confirm our expectations about the models.

The supertagger produces very high-precision results, because it only permits results that are permitted by the grammar. Its low recall is due to its low coverage of the test sequences. It only produced a result at all for about 75% of the sequences. All of these failed sequences serve to pull down the recall, since none of the points in the gold standard path for those sequences were recovered.

The grammarless model has a very much lower precision, since it does not have the handwritten restrictions of the grammar to filter out poor quality

interpretations predicted by the n-gram model. It produces some result for every sequence, though, so does not have its recall pulled down by uninterpreted sequences.

The higher recall than precision shows that it is more inclined to insert additional points than to miss points out. This is probably because it is often failing to recognise repeated points, which are qualitatively distinct in the supertagging model – certain choices of category identify precisely repeated points – whereas they fall out of this model merely by coincidence.

The combined model behaves much as expected. The precision is, of course, pulled down by the inclusion of the grammarless model's results for the failed 25% of sequences, but recall shoots up – the interpretations may be poor, but they are better than nothing. The recall is higher than the grammarless model, because the supertagger actually has a high recall if you consider only those sequences for which it produces an analysis: the low recall is due largely to the low coverage.

The f-score of the combined model shows (a) that this n-gram grammarless backoff is a reasonable method (as a first step, at least) for filling in where the parser is too strict or the supertagger overfit; and (b) that the structural direction provided by the grammar substantially improves over the the short-distance information captured by the model.